

Towards Diverse Generation and Reliable Classification using Neural Networks



Viveka Kulharia
St Catherine's College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy
Hilary 2022

To this wonderful world

Declaration

I hereby declare that this thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work based on publications in collaboration with the co-authors where due acknowledgement is made.

Viveka Kulharia
St Catherine's College

Acknowledgements

First of all, I would like to thank my supervisor, Prof. Philip H.S. Torr, for allowing me to pursue DPhil on which this thesis is based. Phil has been very patient as I tried to venture into different projects. He made me realize the importance of questioning every assumption and his stress on the clarity of thoughts led to a great foundation of the work that followed. He helped me see research as an act of exploration and inspired me to probe every tiny detail. His radically positive-mindedness and bustling nature has been contagious and pushed me through the ups and downs during this research journey. I cherish the experience of working with him and it will be guiding me in the future.

I am also highly grateful to Dr Puneet K. Dokania who was my co-supervisor. The discussions I had with Puneet have been very insightful, from that of equations on board to the general life discussion in pubs. He helped me understand the importance of rigour in not only experiments and theory but also in writing research papers. He has been very tolerant as I needed more time to pursue different projects. He has also been very supportive outside the lab by being there for help in any obstacle life offered.

I am not sure I would have pursued computer vision had I not interacted with Dr Arnab Ghosh, Prof. Vinay P. Namboodiri, and Prof. Amitabha Mukerjee during my undergrad. While Arnab has been a very creative collaborator, Prof. Vinay provided the space, flexibility, and support we needed and ignited our minds with the latest research ideas, and Prof. Amitabha introduced me to the general concepts in AI.

I am also very fortunate to have been invited to work outside Oxford. Prof. Matthias Nießner supported my research visit to TUM and working with him helped me learn how to design the experiments from the very basics. I also got the opportunity to do an industrial internship at Amazon with a very supportive mentor Dr Siddhartha Chandra, always insightful Dr Amit Agrawal, and a great manager Dr Ambrish Tyagi. At the Niantic internship, I got the first-hand experience of working on 3D datasets. I am thankful to Prof. Gabriel J. Brostow for this fantastic opportunity along with encouraging mentors Dr Eric Brachmann, Dr Aron Monszpart, Dr Sara Vicente, and Dr. Guillermo Garcia-Hernando. I also got the wonderful opportunity to collaborate with Dr Edward Ayers, Dr Romain Mueller, Dr Sina Samangooei, and John Redford at Five AI, an emerging autonomous car company.

I am also grateful to Prof. Ingmar Posner and Prof. M. Pawan Kumar who examined my transfer of status and confirmation of status reports and provided very constructive feedback which shaped this thesis. I am also thankful to Prof. Ingmar Posner and Prof. Jinwoo Shin who examined my DPhil Viva.

Collaboration with Mohammad Najafi, AJ, Daniela, Sid, Csabi, Adnane, Jishnu, Amartya, and Stuart (in chronological order) helped me experience diverse perspectives and some of the most efficient ways of doing research, from proposing a project to writing it up. I am indebted for this experience.

I am also thankful to everyone in the vibrant lab. Saumya has always been very welcoming and she organized several unforgettable trips. AJ inspired me not only with how rigorously he drives research but also cars. Sid was always there for any technical and non-technical help besides organizing informative seminars. Lunches with Stuart, AJ, Namhoon, Anurag, Nantas, Luca, Jack, Tommaso, and Nick led to some great discussions and ideas. Puneet, Daniela, and Arslan inspired me to take up long-distance running. It was always fun to talk to Harkirat, Bradley, Rodrigo, Christian, Qizhu, Li, Aravindh, Oscar, Alessio, Song Bai, Xiaojuan, and others. More recently, I am thankful for my interactions with Naeem, Tom, Jimmy, Andrew, Francesco, Adel, Atılım, Ameya, Pau, and Kevin.

I am also thankful to Joanna Zapisek, the project manager, and Cassandra Warren, the group administrator for all the administrative help. Thanks are also due to Kit and Adam who helped manage the robots account, Jeremy who manages our clusters, and the ARC team for supporting the computing needs of our group.

Outside lab, I am fortunate to have had friends in Natasha, Shivohum, Shree, Hemani, Varnika, Aditi, Aditya, Mayur, Sai, Nitin, Angira, Suvajit, Mohita, Samarth, Akash, among others. I had a burst of emotions as I wrote their names, indicating how much I enjoyed non-work time with them, helping make this DPhil journey as delightful as possible.

I owe my deepest gratitude to my parents and my sister Vrindaa. I can't use words to express how much they support and believe in me. Great research requires free thinking and focus, and they did more than I could ever wish in allowing me the priceless opportunity for that.

Finally, I would like to thank the Toyota Research Institute for providing me with generous funding for my DPhil.

Abstract

Humans can easily understand their surroundings, analyze the situation, and even figure out the intentions of others, all in a few seconds. However, computers see the surroundings as an array of numbers and we need to write algorithms for them to make a sense of it. It is a huge challenge but significant strides have been made toward the scene understanding problem over the last decade, thanks to an increase in computing infrastructure and datasets. However, several challenges remain in making computers more ‘intelligent’, some of which are tackled in this thesis. This thesis develops different techniques to create more reliable systems and proposes innovations in loss functions by analyzing the system’s prediction distributions.

First, we try to address the well-known problem of mode collapse in Generative Adversarial Networks (GANs) and propose MAD-GAN, an intuitive generalization of the GANs. MAD-GAN is a multi-agent GAN architecture incorporating multiple generators and one discriminator. To enforce that different generators capture diverse high probability modes, the objective function of the discriminator is designed such that along with identifying the real and fake samples, it is also required to identify the generator that generated the sample if it is fake. Intuitively, to succeed in this task, the discriminator must learn to push different generators toward different identifiable modes. Extensive experiments on synthetic and real datasets demonstrate the effectiveness of the MAD-GAN approach.

We then consider the task of semantic segmentation using weak supervision with bounding box annotations. Bounding boxes are noisy labels for the foreground objects. To focus on foreground objects, our approach predicts a per-class attention map that saliently guides the per-pixel cross-entropy loss and refines the segmentation boundaries. This avoids propagating erroneous gradients due to the incorrect foreground labels. Additionally, our approach learns pixel embeddings to simultaneously optimize for high intra-class feature affinity while increasing discrimination between features across different classes. This helps in capturing global context via long-range pairwise interactions. Qualitative and quantitative results along with ablation studies show the benefit of different loss terms on the overall performance.

The widespread usage of neural networks is particularly dependent on their reliability and trustworthiness. We therefore also consider the important task of calibrating deep neural networks – where a model’s confidence should be the same as its correctness. Ideally, we

want networks to be accurate, calibrated, and confident. We provide a thorough analysis of the factors causing miscalibration and use the obtained insights to justify the empirically excellent performance of focal loss. We show that, as opposed to the standard cross-entropy loss, focal loss allows us to learn models that are already very well-calibrated. When combined with temperature scaling, whilst preserving accuracy, focal loss yields state-of-the-art calibrated models. To facilitate its use in practice, we also provide a principled approach to automatically select the hyperparameter involved in the loss function. Extensive experiments on a variety of computer vision and NLP datasets, and with a wide variety of network architectures, justify that this approach achieves state-of-the-art calibration without compromising accuracy in almost all cases.

One-shot video object segmentation is an important problem in computer vision with many interesting real-world applications. To tackle this, we finally introduce a similarity learning approach, that can learn to perform dense label transfer from one image to the other. More specifically, the objective is to learn a similarity metric for dense pixel-wise correspondence between two images. This learned model can then be used in a label transfer framework to propagate object annotation from the reference frame to all the subsequent frames in a video. Unlike previous methods, our similarity learning approach works fairly well across various domains, even when no domain adaptation is involved. We demonstrate the effectiveness of our method on two standard datasets with favourable results. This approach not only provides good object segmentation but is also time-efficient. Therefore, using it we also achieved second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

Contents

List of Figures	v
List of Tables	xi
1 Introduction	1
1.1 A very brief history of Deep Learning	1
1.2 Challenges in deep learning	5
1.3 Approach	6
1.4 Thesis outline	8
1.5 Contributions	9
1.6 Publications	11
2 Background	13
2.1 Generative Adversarial Networks	13
2.1.1 Mode-collapse problem	14
2.2 Weakly supervised semantic segmentation	15
2.3 Classifier calibration	18
2.3.1 Calibrating classical models	18
2.3.2 Calibrating deep learning	19
3 MADGAN: Multi-Agent Diverse Generative Adversarial Networks	22
3.1 Introduction	24
3.2 Related Work	26
3.3 Preliminaries	28
3.4 Multi-Agent Diverse GAN	29
3.4.1 Multi-Agent GAN Architecture	29
3.4.2 Enforcing Diverse Modes	30
3.5 Experiments	33
3.5.1 Non-Parametric Density Estimation	36
3.5.2 Stacked and Compositional MNIST	37

3.5.3	Diverse Samples for Image-to-Image Translation and Comparison to InfoGAN	39
3.5.4	Diverse-Class Data Generation	41
3.5.5	Diverse Face Generation	42
3.5.6	Unsupervised Representation Learning	42
3.6	Conclusion	42
4	Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation	43
4.1	Introduction	45
4.2	Related Work	47
4.3	Proposed Approach	50
4.3.1	Feed Forward Network Architecture	50
4.3.2	Box And GrabCut Based Losses	51
4.3.3	Attention Weighted Segmentation Loss	52
4.3.4	Discriminative Feature Learning	54
4.3.5	Training Box2Seg	55
4.4	Experiments and Results	55
4.4.1	Implementation Details	55
4.4.2	Quantitative and Qualitative Evaluation	56
4.4.3	Ablation studies	59
4.4.4	Attention Weighted Loss in the Fully-Supervised Setting	59
4.4.5	Semi-supervised Semantic Segmentation	60
4.5	Conclusions	61
5	Calibrating Deep Neural Networks using Focal Loss	62
5.1	Introduction	64
5.2	Problem Formulation	66
5.3	What Causes Miscalibration?	67
5.4	Improving Calibration using Focal Loss	69
5.5	Experiments	74
5.6	Conclusion	78

6	Similarity Learning for One-Shot Video Object Segmentation	79
6.1	Introduction	81
6.2	Related Work	83
6.3	Method	84
6.3.1	Siamese Networks	84
6.3.2	Object Segmentation using Siamese Networks	85
6.3.2.1	Stochastic Pooling	86
6.3.2.2	Label Transfer	87
6.3.3	Domain Adaptation	88
6.4	Experiments	89
6.4.1	Datasets	89
6.4.2	Experiment Setup	89
6.4.3	Results	92
6.5	Conclusion	96
7	Conclusion	97
7.1	Discussion of contributions	98
7.2	Remaining challenges and future directions	102
7.3	Concluding remarks	104
	Appendices	105
A	MADGAN: Multi-Agent Diverse Generative Adversarial Networks	106
A.1	Insights for Diversity in MAD-GAN	106
A.2	Similarity-based competing objective	107
A.2.1	Approach	107
A.2.2	Experiments	109
A.2.3	Diverse Samples for Image-to-Image Translation	109
A.2.4	Unsupervised Representation Learning	110
A.3	Network Architectures and Parameters	110
A.3.1	Non-Parametric Density Estimation	112
A.3.2	Stacked and compositional MNIST Experiments	113
A.3.3	Image-to-Image Translation	113
A.3.3.1	MAD-GAN/MAD-GAN-Sim	113
A.3.3.2	InfoGAN	115
A.3.4	Diverse-Class Data Generation	116
A.3.5	Diverse Face Generations with DCGAN	117
A.3.6	Unsupervised Representation Learning	118

B	Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation	120
B.1	Qualitative results	120
B.2	GrabCut Baselines: Implementation Details and Quantitative Results	122
B.3	Erroneous Annotations in the OpenImages Dataset	123
C	Calibrating Deep Neural Networks using Focal Loss	126
C.1	Reliability Plots	127
C.2	Relation between Focal Loss and Entropy Regularised KL Divergence	128
C.3	Focal Loss and Cross-Entropy on a Linear Model	129
C.4	Proofs	131
C.5	Dataset Description and Implementation Details	133
C.6	Additional Results	135
C.7	Bar plots	139
C.8	Focal Loss is Confident and Calibrated	141
C.9	Ordering of Feature Norms	143
C.10	Early stopping	144
C.11	Machine Translation: A Downstream Task	145
D	Publications	147
D.1	Multi-Agent Diverse Generative Adversarial Networks	147
D.2	Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation	157
D.3	Calibrating Deep Neural Networks using Focal Loss	177
D.4	Similarity Learning for Dense Label Transfer	190
	Bibliography	195

List of Figures

1.2.1 Semantic segmentation performance improvement on PASCAL VOC 2012 (Everingham et al. 2010) over the years. Image source: https://paperswithcode.com/ .	6
3.1.1 Diverse-class data generation using MAD-GAN. The diverse-class dataset contains images from different classes/modalities (in this case, forests, icebergs, and bedrooms). Each row represents generations by a particular generator and each column represents generations for a given random noise input z . As shown, once trained using this dataset, generators of MAD-GAN can disentangle different modalities, hence, each generator can generate images from a particular modality.	24
3.1.2 Multi-Agent Diverse GAN (MAD-GAN). The discriminator outputs $k + 1$ softmax scores signifying the probability of its input sample being from either one of the k generators or the real distribution.	27
3.4.1 Visualization of different generators getting pushed towards different modes. Here, M_1 and M_2 could be a cluster of modes where each cluster itself contains many modes. The arrows abstractly represent generator-specific gradients for the purpose of building intuition.	30
3.5.1 A toy example to understand the behaviour of different GAN variants in order to compare with MAD-GAN (each method was trained for 198000 iterations). The orange bars show the density estimate of the training data and the blue ones for the generated data points. After careful cross-validation, we chose the bin size of 0.1 to get smooth plots based on the number of samples. Changing the bin size does not change the order of the GAN variants.	34
3.5.2 A toy example to understand the behaviour of MAD-GAN with a different number of generators (each method was trained for 198000 iterations). The orange bars show the density estimate of the training data and the blue ones for the generated data points. After careful cross-validation, we chose the bin size of 0.1.	34

3.5.4	Diverse generations for <i>edges-to-handbags</i> generation task. In each sub-figure, the first column is the input, columns 2-4 are generations by MAD-GAN (using three generators), and columns 5-7 are generations by InfoGAN (using three categorical codes). Clearly different generators of MAD-GAN are producing diverse results capturing different colours, textures, design patterns, <i>etc.</i> However, InfoGAN generations are visually almost the same, indicating mode collapse.	38
3.5.6	InfoGAN for <i>edges-to-handbags</i> task by sharing the discriminator and Q Network. In each sub-figure, the first column is the input, columns 2 – 4 are generations when input is categorical code besides conditioning image, and columns 5 – 7 are generations with noise as an additional input. The generations for both architectures are visually the same irrespective of the categorical code value, which clearly indicates that it is not able to capture diverse modes.	38
3.5.8	Diverse generations for <i>night-to-day</i> image generation task. The first column in each sub-figure represents the input. The remaining three columns show the diverse generations of three different generators of MAD-GAN (Our). .	39
3.5.9	Face generations using MAD-GAN. Each generator employed is DCGAN. Each row represents a generator. Each column represents generations for a given random noise input z . Note that, the first generator is generating faces pointing to the left. The second generator is generating female faces with long hair, while the third generator generates images with light backgrounds.	41
3.5.11	Face generations using MAD-GAN. Each sub-figure shows generations by a single generator. The first generator is generating faces with very dark backgrounds. The second one is generating female faces with long hair in light backgrounds, while the third one is generating faces with coloured backgrounds and casual looks (based on facial direction and expression). .	41
4.1.1	Our Box2Seg model is able to produce high-quality semantic segmentation using only bounding box annotations. Each image pair shows ground-truth (left) and predicted segmentation (right).	45
4.2.1	Distribution of filling rate in PASCAL VOC 2012 training dataset. The per-class filling rates vary widely, especially for the categories such as sofa, dining table and bicycle. Unlike Song et al. (2019) that proposes using 1-3 filling rates per class, we use a per-image class-specific filling rate.	49

4.3.1 Overview of our weakly-supervised Box2Seg training pipeline. Our segmentation model is a three-branch CNN which outputs segmentation probabilities y , per-class spatial attention maps α , and pixel embeddings β . B denotes segmentation masks defined using bounding box annotations. GrabCut is applied to B to obtain refined segmentation masks M . We use B and M to supervise the three outputs of the model. Pixel embeddings capture long-range pairwise relationships and the attention map refines the segmentation output y by reducing the effect of label noise. At test time we only use the branch producing the segmentation probabilities y , and discard the other branches that output α, β	50
4.3.2 Visualizing attention maps produced by our network. (a) Input images with overlaid ground truth masks. (b) Grabcut output with ground-truth bounding box. (c) Predicted attention map. Note that while Grabcut output is erroneous and often includes background pixels, our attention maps are concentrated on the objects of interest.	53
4.3.3 Visualization of affinities produced by our network. (a) Input image with overlaid ground truth masks and two pixels marked with purple and green + signs. (b,c) Affinity heatmaps (Eqn. 4.9) w.r.t. the purple and green pixels, resp.	54
4.4.1 Comparison of our segmentation results with those of BCM (Song et al. 2019).	56
4.4.2 (Left) Input images with overlaid bounding box labels. (Middle) Missing pixel-level annotations on foreground objects such as <i>Aeroplane</i> and <i>Person</i> . Note that the bounding box annotations are correct. (Right) Predicted attention map with boxes. Please refer to Sect. 4.4.4 for a discussion on how our AWL can help in cases with conflicting ground truth during fully-supervised training.	58
4.4.3 Failure Cases. (Left) Incorrect label assignment on box annotations B can happen when the smaller bounding box is physically behind the larger bounding box. (Right) Predicted segmentation bleeds into the background.	61
5.3.1 Metrics related to calibration plotted whilst training a ResNet-50 network on CIFAR-10.	68
5.4.1 How metrics related to model calibration change whilst training several ResNet-50 networks on CIFAR-10, using either cross-entropy loss, or focal loss with γ set to 1, 2 or 3.	70
5.4.2 (a): $g(p, \gamma)$ vs. p and (b-d): histograms of the gradient norms of the last linear layer for both cross-entropy and focal loss.	72

5.5.1 Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE, computed for ResNet-50 (first 3 figures) and ResNet-110 (last 3 figures) on CIFAR-10. For each loss, the calibration metric before temperature scaling is represented by the dark colour and after temperature scaling is represented by the light colour. Lower is better. Clearly, our proposed FLSD-53 provides better results both before and after temperature scaling.	75
5.5.2 ROC plots obtained from ResNet-110 and Wide-ResNet-26-10 architectures trained on CIFAR-10 (in-distribution) and tested on SVHN (OoD), both pre and post-temperature scaling.	76
6.1.1 One-shot segmentation outputs for sample videos from DAVIS-2016, obtained using our model without any test domain adaptation, online learning or interaction. Each row illustrates outputs for different time frames of the video.	81
6.3.1 The Siamese architecture used for learning dense correspondence. The model takes in a pair of images and outputs the similarity of their pixels by computing the inner product of their subsampled pixel embeddings ($\langle \Phi_x^K, \Phi_y^K \rangle$). During the evaluation, stochastic pooling is not applied on the target image y and the inner-product output is $\langle \Phi_x^K, \Phi_y \rangle$	86
6.3.2 a) Left image: pixel embeddings are highly correlated and selecting more than $k = 100$ pixels via Stochastic Pooling does not affect the performance (The results are obtained without adaptation.) b) Right image: segmentation performance improves by increasing the number of fine-tuning iterations on the first frame of the test video.	87
6.4.1 Qualitative segmentation outputs for some sample videos from DAVIS-2016 (each row), obtained using our model after fine-tuning.	91
6.4.2 Some examples where domain adaptation (second row) has improved the results over the pre-trained model (first row).	94
6.4.3 The interactive task performance of our method on DAVIS-2017 validation split (left) with $AUC = 70.2\%$ and $\mathcal{J}@60 = 54.8\%$, and Test-Dev split (right) with $AUC = 54.5\%$ and $\mathcal{J}@60 = 39.4\%$. Note that in the Test-Dev set, the 60s time-point occurs before the second interaction because more complex videos with a larger number of objects are present and processed in this data split.	94
6.4.4 The impact of successive interactions on the quality of the segmentation outputs. The top row illustrates the input images and the segmentation results for consecutive interactions are depicted in the 2nd to 8th rows.	95

A.2.1	MAD-GAN-Sim compared with MAD-GAN. All the generators share parameters of all the layers except the last one. Two proposed diversity enforcing objectives, ‘competing’ (MAD-GAN-Sim) and ‘generator identification’ (MAD-GAN), are shown at the end of the discriminator	108
A.2.3	MAD-GAN-Sim: Diverse generations for ‘edges-to-handbags’ image generation task. The first column in each sub-figure represents the input. The remaining three columns show the diverse outputs of different generators. It is evident that different generators are able to produce very diverse results capturing colour (brown, pink, black), texture, design pattern, and shininess, among others.	111
A.2.5	MAD-GAN-Sim: Diverse generations for ‘night to day’ image generation task. The first column in each sub-figure represents the input. The remaining three columns show the diverse outputs of different generators. It is evident that different generators are able to produce very diverse results capturing different lighting conditions, different sky patterns (cloudy vs clear sky), different weather conditions (winter, summer, rain), and different landscapes, among many other minute yet useful cues.	112
B.1.1	Additional qualitative results: (a) Original Image with overlaid ground-truth (b) Results with Box baseline (c) Results with GrabCut baseline (d) Box2Seg (with CRF). Please refer to Table 4.4.3 in the main chapter for a description of the baselines.	121
B.3.1	Erroneous bounding box ground-truth in the OpenImages dataset: Col. 1 shows ground-truth bounding boxes that contain multiple instances inside. Col. 2 shows missing bounding boxes for the person class. Bounding-box annotations are not exhaustive in OpenImages.	124
B.3.2	Erroneous bounding box ground-truth in the OpenImages dataset: representative examples from the dataset where bounding boxes are not aligned with the object boundaries. Eg. the bounding box of the bicycle in row 2 col. 1 overestimates the object boundary on the upper-right corner, and the bounding box underestimates the legs of the person in row 3 col. 1.	125
C.1.1	The confidence values for training samples at different epochs during the NLL training of a ResNet-50 on CIFAR-10 (see §5.3). Top row: reliability plots using 25 confidence bins; bottom row: % of samples in each bin. As training progresses, the model gradually shifts all training samples to the highest confidence bin. Notably, it continues to do so even after achieving 100% training accuracy by the 300 epoch point.	127

C.2.1 Optimal $\hat{p}$ for various values of q	128
C.3.1 (a): Norm of logits (b): Norm of weights.	129
C.3.2(a): Confidence of mis-classifications (b): Decision boundary of a linear classifier trained using cross entropy (c): Decision boundary of a linear classifier trained using focal loss	130
C.7.1 Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for ResNet-50 (first 3 figures) and ResNet-110 (last 3 figures) on CIFAR-100.	140
C.7.2 Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for Wide-ResNet-26-10 (first 3 figures) and DenseNet-121 (last 3 figures) on CIFAR-100.	140
C.7.3 Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for Wide-ResNet-26-10 (first 3 figures) and DenseNet-121 (last 3 figures) on CIFAR-10.	141
C.7.4 Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for ResNet-50 on Tiny-ImageNet.	141
C.8.1 Qualitative results showing the performance of Cross-Entropy, Brier loss, MMCE, and Focal Loss (sample-dependent γ 5,3) for a ResNet-50 trained on CIFAR-10. The first row of images has been correctly classified by networks trained on all of the four loss functions and the second row of images has all been incorrectly classified. For each image, we present the actual label, the predicted label, and the confidence of the prediction both before and after temperature scaling.	142
C.9.1 L_2 norm of features obtained from the last ResNet block (before the linear layer) of ResNet-50 averaged over the entire training dataset of CIFAR-10 using a batch size of 128.	144
C.11. Changes in <i>test set</i> BLEU score and <i>validation set</i> ECE with temperature, for models trained using (a) cross-entropy with hard targets (CE) (b) cross-entropy with label smoothing (LS) ($\alpha = 0.1$), and (c) focal loss (FL) ($\gamma = 1$).	146

List of Tables

3.5.1 Synthetic experiment on 1D GMM (Fig. 3.5.1). This table does not report an in-depth statistical analysis using different training runs, however, MAD-GAN was consistently superior to the baselines in all the iterations we tried during model development for this experiment.	35
3.5.2 Synthetic experiment with different number of MAD-GAN generators (same setup as in Fig. 3.5.1). This table does not report an in-depth statistical analysis using different training runs, however, MAD-GAN with four generators was consistently superior to MAD-GAN with the other number of generators in all the iterations we tried during model development for this experiment.	35
3.5.3 Stacked-MNIST experiments and comparisons. Note that three generators are used for MAD-GAN.	37
3.5.4 Compositional-MNIST experiments and comparisons. Note that three generators are used for MAD-GAN.	37
4.4.1 Comparison of Box2Seg to previous weakly supervised semantic segmentation methods on PASCAL VOC validation set. C=dense-CRF post-processing.	57
4.4.2 Per-class results on PASCAL VOC 2012 Validation set. We compare our <i>Box2Seg</i> (with and without dense CRF) with those of the previous state-of-the-art methods. Please note that Li <i>et al.</i> (CRF) results require COCO annotations. Per-class results from Song et al. (2019) for their ResNet-101 model are not available.	57
4.4.3 Ablation study showing the effect of our loss functions in improving the performance over baseline methods.	59
4.4.4 Improvements in segmentation accuracy on the PASCAL VOC 2012 validation set in the fully supervised setting using AWL.	60
4.4.5 Semi-supervised segmentation using Box2Seg model as initialization. ($S=\tau\%$) implies $\tau\%$ images with pixel annotations are used for semi-supervised fine-tuning. C=dense-CRF post-processing. †: ResNet-101 backbone (refer Song et al. (2019)).	60

5.5.1 ECE (%) computed for different approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets. $T \approx 1$ indicates an innately calibrated model.	75
5.5.2 Test set error (%) computed for different approaches.	75
5.5.3 AUROC (%) computed for models trained on CIFAR-10 (in-distribution), and using SVHN and CIFAR-10-C (Gaussian Noise corruption with severity level 5) respectively as the OoD datasets.	76
6.4.1 Results on DAVIS-2016 (left) and DAVIS-2017 (right) validation Dataset. The reported numbers are in (%) units. D.A.: Domain Adaptation. Note that in the case of the DAVIS-2017 dataset, we did not have access to the results of other methods without fine-tuning, so all the results in this table are with domain adaptation. Statistical test to get confidence significance was not done using different training runs.	93
A.3.1 Non-Parametric density estimation architecture for discriminators (Disc), encoders (Enc), decoders (Dec), and Q Network (QNet). nGen is a number of generators, fc is fully connected layer.	111
A.3.2 Generator architecture for 1000 class stacked-MNIST experiment. For MAD-GAN, all the layers except those mentioned in the last two rows are shared.	113
A.3.3 Discriminator architecture for 1000 class stacked-MNIST experiment. For MAD-GAN, with k generators, it is adapted to have $k + 1$ dimensional last layer output. For InfoGAN, with 156 dimensional salient variables and 100-dimensional incompressible noise, it is adapted to have 156 dimensional output for the Q network.	114
A.3.4 Discriminator architecture for 1000 class compositional-MNIST experiment. For MAD-GAN, with k generators, it is adapted to have $k + 1$ dimensional last layer output. For InfoGAN, with 156 dimensional salient variables and 100-dimensional incompressible noise, it is adapted to have 156 dimensional output for the Q network.	114
A.3.5 DCGAN Generator: All the layers except the last one are shared among all the three generators.	117
A.3.6 Discriminator architecture for diverse-class data generation and diverse face generation: The last layer output is $k + 1$ dimensional for MAD-GAN with k generators (normalizer is softmax). ‘RESIDUAL’ layer is elaborated in Table A.3.7.	118

A.3.7 Residual layer description for Tab. A.3.6.	118
A.3.8 DCGAN Discriminator: It is adapted to have $k + 1$ dimensional last layer output for MAD-GAN with k generators. (normalizer is softmax).	119
A.3.9 The misclassification error of MAD-GAN and MAD-GAN-Sim on SVHN while varying the number of generators.	119
 B.2.1 Comparison of different methods using GrabCut pseudo-ground-truths obtained with our Algorithm. We consider two <i>evaluation-only</i> baselines, <i>GrabCut-NoTrain-GT</i> and <i>GrabCut-NoTrain-Det</i> , where we evaluate the accuracy of Grabcut-based Segmentation (refer to Algorithm 1) on Ground-Truth and Detected boxes respectively. We also consider a learning-based baseline, <i>Grabcut</i> which is a CNN baseline (using our ResNet-101 architecture described in Sec. 4.1 of the main chapter) trained with our loss $\mathcal{L}_{GC}$ described in Eqn. 4.2 in the main chapter.	123
 C.6.1 Adaptive ECE (%) computed for different approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.	136
C.6.2 Classwise-ECE (%) computed for different approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.	136
C.6.3 ECE (%) computed for different focal loss approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.	137
C.6.4 AdaECE (%) computed for different focal loss approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.	137
C.6.5 Classwise-ECE (%) computed for different focal loss approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.	138
C.6.6 Error (%) computed for different focal loss approaches.	138
C.6.7 Top-1 and Top-5 accuracies computed for different approaches.	138
C.6.8 NLL (%) computed for different approaches pre temperature scaling.	139
C.6.9 NLL (%) computed for different approaches post temperature scaling (cross-validating T on ECE).	139
C.6.10 MCE (%) computed for different approaches pre temperature scaling.	139

C.6.1 MCE (%) computed for different approaches post temperature scaling (cross-validating T on ECE).	140
C.8.1 Percentage of test samples predicted with confidence higher than 99% and the corresponding accuracy for Cross Entropy, MMCE and Focal loss computed both pre and post-temperature scaling (represented in the table as pre-T and post-T respectively).	141
C.10. Classification errors and ECE scores obtained from ResNet-50 models trained using cross-entropy and focal loss with different early stopping criteria (best in hindsight ECE, NLL and classification error on the validation set) applied during training. In the table CE and FL stand for cross-entropy and focal loss respectively and the Full Criterion indicates models where early stopping has not been applied.	144
C.11. Test set ECE and BLEU score both pre and post-temperature scaling for cross-entropy (CE) with hard targets, cross-entropy with label smoothing (LS) ($\alpha = 0.1$) and focal loss (FL) ($\gamma = 1$).	145

1

Introduction

This is an integrated thesis where published work in peer-reviewed conferences and workshops is reformatted as different chapters. Section 1.1 briefly provides the journey taken to reach the current state of deep learning. Section 1.2 then discusses some of the challenges faced by deep learning which lead to the work presented in this thesis. The approach taken to address these challenges is briefly presented in Section 1.3. Section 1.4 provides the outline of the remaining chapters of the thesis while Section 1.5 discusses the main contributions of the thesis. Finally, Section 1.6 lists the publications which form the main chapters of this thesis.

1.1 A very brief history of Deep Learning

Humans can easily carry out a wide variety of tasks even when they may not have much prior experience. For example, it is very easy to identify and locate a chair in an image even if such a chair has not been seen before. Such accomplishment usually involves skills like visual scene recognition, *e.g.*, having an idea of objects, and having some form of imagination that adds to their experience.

For centuries, machines have been used to help humans in a lot of repetitive and laborious tasks. However, it is only recently that they became computationally strong enough such that they could be used to learn and observe non-trivial but useful patterns, so much so that the machine learning field has also been called ‘Artificial Intelligence’.

While work on machine learning has been done for decades, it used to require careful design of domain-dependent feature extractors whose output could then be fed into classifiers, *e.g.*, SIFT (Lowe 2004) and HOG (Dalal and Triggs 2005). It all changed with representation learning where these feature extractors can adapt themselves based on the data using backpropagation (LeCun et al. 1989). Deep learning involves several representation learning layers which are trained using back-propagation. Due to the need for data to train these feature extractors, the superiority of deep learning became more evident with access to large-scale datasets like ImageNet (Deng et al. 2009).

Machine learning is now the basis of several aspects of our society. There are several applications like recommendation systems for e-commerce or songs, image searches, chat-bots, driving assistants, *etc.*

A lot of work has been done to get computers to understand the dynamically changing environment, just like humans do. Deep learning has led to a lot of breakthroughs in many computer vision tasks, some of which are mentioned below.

Classification The task of object recognition is an important aspect of the human visual system, so this task has been considered one of the important challenges to conquer in the field of computer vision. Image classification involves categorizing an image based on the objects contained in it. It has been part of the famous image recognition challenge based on the ImageNet dataset: ILSVRC (Russakovsky et al. 2015) which in 2012 was won by AlexNet (Krizhevsky et al. 2012) by a large margin, a significant breakthrough. This model was based on a convolutional neural network and it was for the first time that a deep learning-based model won this competition. It led to a lot of significant work in deep learning. There was a trend to make the models deeper to get better performance. VGG-nets (Simonyan and Zisserman 2014b) used a simple strategy of stacking convolutional layers of the same shape to construct very deep networks. However, it seemed that after a certain depth the model accuracy would start decreasing (He et al. 2016). To counter this ResNets (He et al. 2016) introduced deep residual learning which helps it further lower the error rate on the ImageNet dataset while also being able to generalize this learned model to other tasks.

Segmentation Humans can not only recognize the objects present in an image but can also provide fine detail about the location of each object. As compared to the classification task which involves extracting image-level information, image segmentation involves extracting pixel-level information: classifying every pixel into a category. It has several applications in

driverless cars, augmented reality, medical image analysis, *etc.* Long et al. (2015) showed that the segmentation problem can be approached using the fully convolutional network. For this, they supplement the usual contracting convolutional network with successive layers with upsampling operators. It led to the possibility of transfer learning from classification to segmentation. It became common to use ImageNet pre-trained networks whose weights could be finetuned for segmentation. There have been several other advances in architecture. U-Net (Ronneberger et al. 2015) introduced a U-shaped network where contracting layers capture context and symmetric expanding layers propagate the context information to higher resolution layers for localization. Atrous convolutions (Chen et al. 2014) or dilated convolutions (Yu and Koltun 2016) have been widely used to increase the field of view of convolutional filters without increasing the number of parameters. This is useful for getting a larger context. Besides, spatial pyramid pooling (He et al. 2014) has been useful in capturing objects of different scales.

Generative models Humans have very good imagination skills which they can use to recreate experiences, *e.g.*, in dreams. They can also use these imagination skills to create artworks and stories. By generative models, we refer to the models that can learn to represent an estimate of the distribution from which the samples of the data are drawn (Goodfellow 2016). The applications of generative models include image super-resolution, interactive image generation, and image translation among others. As the data distribution is unknown, variational autoencoder (VAE) (Kingma and Welling 2014) does a deterministic and tractable approximation of the unknown posterior. In this process, it maximizes evidence lower bound (ELBO) which helps it maximize the likelihood of the data and thus learn to generate the data. However, it is dependent on the quality of the posterior approximation. Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) do not explicitly define the data density function but train a generator network by simultaneously training a discriminator network. The generator and discriminator are made to play a game where the discriminator has to identify if a given sample is real or generated while the generator has to fool the discriminator. When the Nash equilibrium is reached, the generator can sample from the data distribution. However, finding the Nash equilibrium is not easy and the network pair can empirically get stuck in a local optimum. Normalizing Flows (Rezende and Mohamed 2015; Kingma and Dhariwal 2018) is another popular generative modelling approach that optimizes the exact log-likelihood by optimizing an invertible function from data distribution

to a latent distribution. The optimization is over the jacobian matrix and prior over latent. However, the requirements of invertibility and Jacobian computations bring some limitations.

Videos Our surrounding is dynamic and always changing with time. Besides a two-dimensional image, it, therefore, becomes important to also consider time as another dimension of input to have a video input. This additional temporal cue is useful for better scene understanding. It has applications in action recognition, video summarization, human intention prediction, *etc.* Simonyan and Zisserman (2014a) used two separate networks: one to capture the spatial information on appearance from still images and another to capture temporal information on motion between frames using optical flow input. They called it two-stream architecture. This soon became quite a popular approach as it achieved similar performance to the previous best hand-crafted features. It used a late fusion of spatial-temporal stream at the softmax layers. Feichtenhofer et al. (2016) investigates the different ways of fusing the two streams. It proposes fusing at the last convolutional layer rather than the softmax layer to substantially save on parameters while maintaining good performance. While the above networks are very popular, they required pre-computing optical flow which requires offline storage. Carreira and Zisserman (2017) adopt 3D convolutional layers to directly learn about temporal patterns from an RGB stream. They bootstrap the parameters of the 3D filter from the 2D pre-trained ImageNet parameters by considering video as a sequence of the same 2D image. However, it still finds optical flow as part of the two-stream configuration useful in getting better results.

Recently deep learning has had several impactful work with transformer architecture (Kolesnikov et al. 2021) showing promising results over convolutional architecture when data is abundantly available. Generative adversarial networks have produced great visual results with work like StyleGAN (Karras et al. 2019; Karras et al. 2020). Several companies like Tesla (Tesla 2021) and GMCruise (Cruise 2020) have been compelled to use deep learning for futuristic autonomous cars. However, there remain several challenges.

1.2 Challenges in deep learning

*If a machine is expected to be infallible,
it cannot also be intelligent.*

Alan Turing

Deep learning outperforms humans on several tasks, *e.g.*, Go game (Silver et al. 2016) and ImageNet classification challenge (Russakovsky et al. 2015). However, several challenges still remain.

Mode collapse While generative adversarial networks have been there for a while, they faced a major problem of mode-collapse. It was not easy to generate samples from all the different modes of data. Though, theoretically, convergence guarantees the generator learning the true data distribution. However, practically, reaching the true equilibrium was difficult and not guaranteed, which potentially lead to the aforementioned problem of mode collapse. The limitations were mostly due to the network getting stuck in the local optimum. The standard objective for the generator was to just ‘fool’ the discriminator which requires the generated sample to look realistic, not diverse.

Human labeling The accuracy of the semantic segmentation approaches has improved significantly in recent years as evident from Figure 1.2.1. This is mostly due to the availability of datasets with human-labelled per-pixel image annotations. These annotations are expensive and slow to obtain even for large corporations. It also limits the variety of object categories labelled. Approaches using synthetic datasets can be data-efficient, but they do not always generalize well to the real world due to the domain gap.

Confidence calibration Even though convolutional neural networks can outperform humans on the image classification task, their confidence in the predictions can not be measured. If we consider the probability value that they assign to the predicted class labels as confidence, then they are usually overconfident. This is a major problem, since if networks are routinely overconfident, then downstream components cannot trust their predictions. This is a bottleneck for their widespread deployment, especially in critical environments like healthcare and autonomous cars.

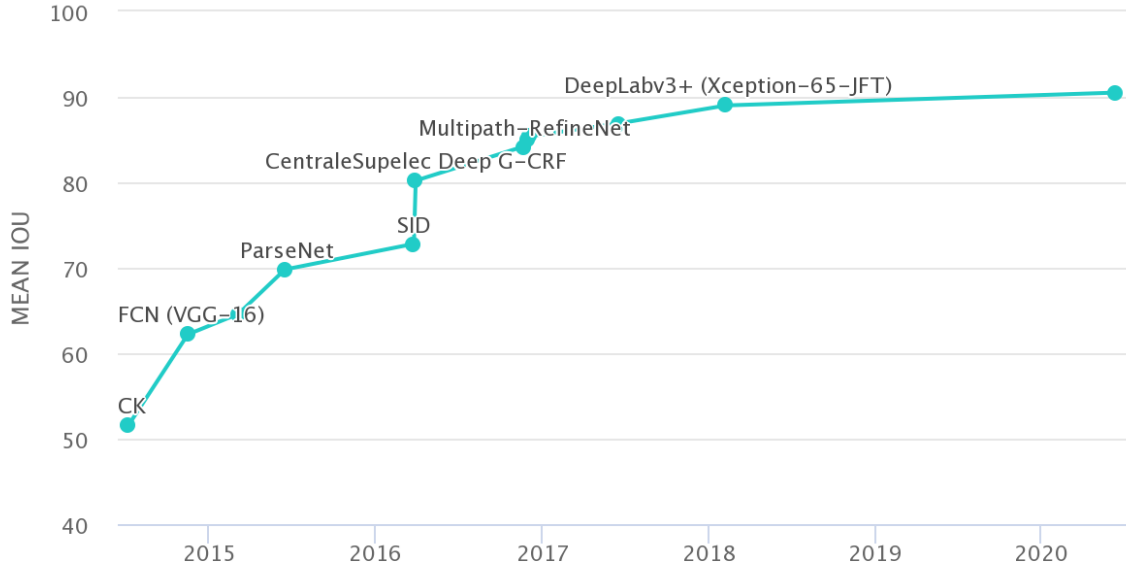


Figure 1.2.1: Semantic segmentation performance improvement on PASCAL VOC 2012 (Everingham et al. 2010) over the years. Image source: <https://paperswithcode.com/>.

Real-time inference Object segmentation in a video is an important problem in computer vision with many interesting real-world applications in autonomous driving, surveillance, and video editing. Given one or more target objects, the task is to track and segment those objects in every frame of the video. However, it is not easy to do this in real-time while maintaining good performance. This is mostly due to the object variation over time in a video as well as the possibility of a domain gap in every new video which requires heavy matching computations. The little work in this regard may be due to the lack of relevant benchmarks. While most of the benchmarks track the network accuracy, there are very few which track the inference speed.

1.3 Approach

To address the aforementioned issues, this thesis mostly discusses using prediction distribution along with differently weighted cross-entropy loss functions.

Generative adversarial networks (GANs) are an avenue where we found a differently weighted cross-entropy loss function to be useful. GANs have been popular due to the quality of images they can generate, which look realistic (Isola et al. 2017; Karras et al. 2018). However, they would face the problem of mode-collapse, a phenomenon where a trained

network would only be able to sample from a few modes of the training data distribution. In other words, the support of the generated data distribution or ‘prediction distribution’ would only be a subset of the real data distribution. This would happen due to the local optima reached by the networks in the training. To explicitly avoid this, we considered a framework with an ensemble of generators such that each generator would be expected to generate samples from different modes of the data. In this regard, we made a simple modification to the objective of the discriminator as compared to the standard GANs (Goodfellow et al. 2014). We proposed the generator identification-based objective, which is a cross-entropy loss where terms are weighted according to the generator generating a sample. So just by considering the generator ‘distribution’ in the loss, this simple generator-based weighting modification pushes each generator of the multi-agent model to capture different modes of the data distribution.

The attention-weighted cross-entropy loss also has a fascinating effect in the case of weakly supervised segmentation. The weakly-supervised segmentation is an important problem as getting densely segmented annotations are very time-consuming and costly (Everingham et al. 2010). We consider weak supervision in the form of bounding boxes where we treat each bounding box as a noisy label for a foreground object. The attention-weighted loss tries to deal with the label noise in the bounding box annotations. We modify the standard Cross-Entropy loss function by introducing a multiplicative attention score per pixel. We predict the per-class attention map, using an extra head of the network, that saliently guides the per-pixel cross-entropy loss to focus on foreground pixels and refines the segmentation boundaries. This avoids propagating erroneous gradients due to incorrect foreground labels on the background. Thus, this method could attend to segmentation in the bounding boxes and remove background noise. Hence we can improve segmentation accuracy using predicted attention distribution in the loss.

Next, we observe the possible factors of miscalibration, *e.g.*, the network getting overconfident on wrong predictions at test time and the test NLL gradually increasing after a certain point of training. We hypothesize that weight magnification is the possible cause of this phenomenon. To address this issue, we propose using focal loss which is like cross-entropy loss where each term is weighted by the incorrectness of a prediction, *i.e.*, $(1 - \text{predicted probability})^\gamma$. We show that it inherently tries to regularize the prediction, by maximizing entropy while minimizing the KL divergence between the predicted and the target distribution. We also show that the focal loss naturally regularizes the weights of the network during training. Capitalizing on this observation, we propose a sample-dependent variant of focal loss where the γ is decided by the predicted probability for a sample. We empirically

show its performance over existing methods on a wide array of network architectures and datasets. Thus we can improve the model performance and calibration just by the term-wise weighting of the cross-entropy loss using the predicted distribution. This work essentially has a consequence on the way neural networks are trained for them to be more reliable.

For the video object segmentation, we specifically consider the one-shot case. We use a similarity learning approach, that can learn to perform dense label transfer from one image to the other. This learned model can then be used in a label transfer framework to propagate object annotation from a reference frame to all the subsequent frames in a video. Using stochastic pooling, we can utilize per-image feature distribution to propagate labels in a computationally efficient way. Unlike previous methods, this similarity learning approach works fairly well across various domains, even when no domain adaptation is involved. It is also time-efficient during inference for interactive video object segmentation.

1.4 Thesis outline

The remaining chapters of this thesis are outlined as mentioned here. Please note that each of the chapters 3–6 is based on a different published work at a leading peer-reviewed conference in machine learning and computer vision. These published papers have been reformatted with some additional experiments for Chapters 3 and 6, and the supplementary material has been included as part of the Appendices. Each of these chapters is self-contained with its own related work section centred around the contribution of the paper. These contributions are now described next:

Chapter 2 This chapter provides background on generative adversarial networks along with the mode collapse issue, semantic segmentation with different image labelling strategies, and classifier calibration.

Chapter 3 This chapter presents simple modifications to GANs objective such that an ensemble of generators can learn to capture diverse modes of the dataset distribution. It shows interesting results on the problems of image-to-image translation tasks as well as image clustering tasks among others.

Chapter 4 This chapter looks beyond per-pixel image annotations for semantic segmentation and seeks to do semantic segmentation when only bounding box annotations are available at training time. It uses attention-weighted loss along with an affinity-based loss to achieve visual results which rival even those of the models trained using dense image annotations.

Chapter 5 This chapter tries to observe the possible causes of miscalibration in classifier neural networks from both theoretical and empirical perspectives. It finds focal loss to be a good alternative to existing loss functions to get better calibrations. It also presents a possible way to adapt focal loss such that its hyper-parameter need not be tuned for every new dataset and network architecture.

Chapter 6 This chapter looks at the one-shot video object segmentation task and introduces a simple, flexible, and fast similarity learning-based method. The proposed method is simple enough to work across various domains, even when no domain adaptation is applied. The method is also flexible and fast enough to be used for the interactive video object segmentation task where users may specify an object to segment just using scribbles.

Chapter 7 We conclude this thesis in this chapter, by summarizing the contributions presented in each of the chapters. We also discuss the remaining challenges for future work.

1.5 Contributions

The main contribution of this thesis is on analyzing and improving deep network performance by simple modifications to existing loss functions which also help get useful prediction distribution. These are summarized in the following:

Chapter 3: Multi-Agent Diverse Generative Adversarial Networks This chapter proposes a simple generalization to GANs objective which can incorporate an ensemble of generators while keeping a single discriminator. We name this approach multi-agent diverse generative adversarial networks (MAD-GAN). It enforces that multiple generators capture diverse modes of data distribution which tries to avoid the mode-collapse problem in standard GANs. To do this, the discriminator objective of MAD-GAN is designed such that for any given sample it has to identify if it is real or the generator that generated the sample. This helps the discriminator push different generators towards different modes of data distribution.

It shows interesting results on a synthetic dataset, and image clustering tasks among others. In Appendix, we also introduce a similarity-based competing objective (MAD-GAN-Sim) which considers a user-defined similarity metric to encourage different generators to generate diverse samples. It works quite well in generating diverse image-to-image translations and is also effective on the unsupervised feature representation task.

Chapter 4: Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation

This chapter considers the task of semantic segmentation where only weak supervision in the form of bounding boxes is available. The proposed approach is called Box2Seg. It treats the bounding boxes as noisy labels for the foreground objects as a bounding box may also overlap a background area beyond foreground pixels. During training time the network has three branches for segmentation, attention, and discriminative feature prediction. The attention branch predicts a per-class attention map which saliently guides the per-pixel cross-entropy loss to focus on foreground pixels. It avoids backpropagating erroneous gradients due to incorrect foreground labels on the background and refines the segmentation boundaries. The discriminative feature prediction branch outputs pixel embeddings to simultaneously optimize for high intra-class feature affinity while increasing discrimination between features across different classes. At the test time, only the segmentation branch is used. Box2Seg outperformed the contemporary methods in the weakly supervised segmentation task. Interestingly, fully supervised methods can also benefit from attention-weighted loss in the presence of exhaustive bounding box annotations but missing pixel-level annotations.

Chapter 5: Calibrating Deep Neural Networks using Focal Loss

This chapter attempts to make deep neural networks more trustworthy, especially in the classification domain. If the prediction probability of a classifier neural network is considered as the network's confidence in its prediction, then modern classifier neural networks are usually overconfident and not calibrated. For deep neural networks to have widespread adoption, they need to be accurate and calibrated. In this chapter, we first analyze the possible causes of miscalibration. We then use these insights to show and justify the strength of focal loss over cross-entropy loss in getting better calibration. We also provide a principled approach to automatically select the hyperparameter involved in the focal loss to facilitate its use in practice. Extensive experiments on a variety of computer vision and NLP datasets and with a wide variety of network architectures justify the proposed approach.

Chapter 6: Similarity Learning for Dense Label Transfer This chapter looks at the one-shot video object segmentation task where the task is to separate an object from the background in a video, given the object mask of the first frame. This chapter introduces a simple, flexible, and fast similarity learning-based method that can perform dense label transfer from one image to the other. More specifically, we use Siamese neural network to learn a similarity metric for dense pixel-wise correspondence between a pair of images. We make this process memory-efficient using stochastic pooling. This approach works fairly well across various domains even without using domain adaptation. This approach also works for interactive video object segmentation where a user provides scribbles instead of dense object masks to initiate and improve object segmentation and thus achieved the second position in the corresponding DAVIS challenge in terms of both quality and speed.

1.6 Publications

In this integrated thesis, the contributions in each of the chapters, as described in the previous section, are based on different publications as indicated below (*: Equal contribution). These publications are also presented in Appendix D exactly as published:

Chapter 3

(Ghosh et al. 2018): **Viveka Kulharia***, Arnab Ghosh*, Vinay P. Namboodiri, Philip HS Torr, and Puneet K. Dokania. “Multi-agent diverse generative adversarial networks.” *Computer Vision and Pattern Recognition (CVPR)*, 2018.

Chapter 4

(Kulharia et al. 2020): **Viveka Kulharia***, Siddhartha Chandra*, Amit Agrawal, Philip Torr, and Ambrish Tyagi. “Box2seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation.” *European Conference on Computer Vision (ECCV)*, 2020.

Chapter 5

(Mukhoti et al. 2020): **Viveka Kulharia***, Jishnu Mukhoti*, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. “Calibrating deep neural networks using focal loss.” *Neural Information Processing Systems (NeurIPS)*, 2020.

Chapter 6

(Najafi et al. 2018): **Viveka Kulharia***, Mohammad Najafi*, Thalaiyasingam Ajanthan, and Philip HS Torr. “Similarity learning for dense label transfer.” *The 2018 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2018.

2

Background

Here we first review generative adversarial networks (GANs) in Section 2.1. We then look at the literature review of semantic segmentation with different forms of weak supervision in Section 2.2. Next in Section 2.3 we review the background of classifier calibration.

2.1 Generative Adversarial Networks

Here we present a brief review of standard GANs (Goodfellow et al. 2014). Given a set of samples $\mathcal{D} = (x_i)_{i=1}^n$ from the true data distribution p_d , the GAN learning problem is to obtain the optimal parameters θ_g of a generator $G(z; \theta_g)$ that can sample from an approximate data distribution p_g , where $z \sim p_z$ is the prior input noise (e.g., samples from a normal distribution). In order to learn the optimal θ_g , the GAN objective (Eq. (2.1)) employs a discriminator $D(x; \theta_d)$ that learns to differentiate between a *real* (from p_d) and a *fake* (from p_g) sample x . The overall GAN objective is:

$$\min_{\theta_g} \max_{\theta_d} V(\theta_d, \theta_g) := \mathbb{E}_{x \sim p_d} \log D(x; \theta_d) + \mathbb{E}_{z \sim p_z} \log (1 - D(G(z; \theta_g); \theta_d)) \quad (2.1)$$

The above objective is optimized in a block-wise manner where θ_d and θ_g are optimized one at a time while fixing the other. For a given sample x (either from p_d or p_g) and the parameter θ_d , the function $D(x; \theta_d) \in [0, 1]$ produces a score that represents the probability

of x belonging to the true data distribution p_d (or probability of it being real). The objective of the discriminator is to learn parameters θ_d that maximize this score for the true samples (from p_d) while minimizing it for the fake ones $\tilde{x} = D(z; \theta_g)$ (from p_g). In the case of a generator, the objective is to minimize $\mathbb{E}_{z \sim p_z} \log(1 - D(G(z; \theta_g); \theta_d))$, equivalently maximize $\mathbb{E}_{z \sim p_z} \log D(G(z; \theta_g); \theta_d)$. Thus, the generator learns to maximize the scores for the fake samples (from p_g), which is exactly the opposite of what the discriminator is trying to achieve. In this manner, the generator and the discriminator are involved in a minimax game where the task of the generator is to maximize the mistakes of the discriminator. Theoretically, at equilibrium, the generator learns to generate real samples, which means $p_g = p_d$.

2.1.1 Mode-collapse problem

Despite the remarkable success of GAN, it suffered from the major problem of *mode collapse* (Che et al. 2017; Chen et al. 2016; Metz et al. 2017; Arjovsky and Bottou 2017; Salimans et al. 2016). Though, theoretically, convergence guarantees the generator learning the true data distribution. However, practically, reaching the true equilibrium was difficult and not guaranteed, which potentially lead to the aforementioned problem of mode collapse. Broadly speaking, there were two schools of thought to address the issue: (1) improving the learning of GANs to reach better optima (Metz et al. 2017; Arjovsky et al. 2017; Berthelot et al. 2017; Juefei-Xu et al. 2017); and (2) explicitly enforcing GANs to capture diverse modes (Che et al. 2017; Chen et al. 2016; Liu and Tuzel 2016).

Metz et al. (2017) proposed a surrogate objective for the update of the generator with respect to the unrolled optimization of the discriminator (UnrolledGAN) to address the issue of convergence of the training process of GANs. This improves the training process of the generator which in turn allows the generator to explore better coverage to true data distribution.

W-GAN (Arjovsky et al. 2017) employs integral probability metrics based on the earth mover's distance rather than the JS-divergences that the original GAN uses. BEGAN (Berthelot et al. 2017) builds upon W-GAN using an autoencoder-based equilibrium enforcing technique alongside the Wasserstein distance. GoGAN (Juefei-Xu et al. 2017) introduced a training procedure for the training of the discriminator using a maximum margin formulation alongside the earth mover distance based on the Wasserstein-1 metric.

Che et al. (2017) proposed a mode regularized GAN (ModeGAN) which uses an encoder-decoder paradigm. The basic idea behind ModeGAN is that if a sample from the true data

distribution p_d belongs to a particular mode, then the sample generated by the generator (fake sample) when the true sample is passed through the encoder-decoder is likely to belong to the same mode. ModeGAN assumes that there exist enough true samples from a mode for the generator to be able to capture it.

The recent work called InfoGAN (Chen et al. 2016) proposed an information-theoretic extension to GANs to address the problem of mode collapse. Briefly, InfoGAN disentangles the latent representation by assuming a factored representation of the latent variables. To enforce that the generator learns factor-specific generations, InfoGAN maximizes the mutual information between the factored latent and the generator distribution.

Liu and Tuzel (2016) presented Coupled GAN, a method for training two generators with shared parameters to learn the joint distribution of the data. The shared parameters guide both generators towards similar subspaces but since they are trained independently on two domains, they promote diverse generations.

2.2 Weakly supervised semantic segmentation

The accuracy of semantic segmentation approaches has improved significantly in recent years (Chen et al. 2017a; Chen et al. 2018b; Long et al. 2015; Noh et al. 2015; Zhang et al. 2018a; Zhang et al. 2019; Zheng et al. 2015). The mean Intersection-over-Union (mIoU) metric on the PASCAL VOC semantic segmentation benchmark has improved by over 20% in the last five years. The success of these efforts can be broadly attributed to (i) advancements in deep neural network architectures and loss functions, (ii) efficient processing (better GPUs), and (iii) the availability of large datasets of images with human-labelled per-pixel annotations (Everingham et al. 2010; Lin et al. 2014b). Improvements in network architectures and hardware capabilities benefit all deep learning tasks. However, large datasets with per-pixel semantic labels are both expensive and slow to obtain (Bearman et al. 2016; Everingham et al. 2010) (typically 4-10 minutes per image), making it challenging to scale to a large number of object categories. Consequently, even the largest semantic segmentation datasets (Lin et al. 2014b; Zhou et al. 2017b) include less than a couple of hundred object categories.

To address the scarcity of labelled data, some previous works have used synthetic datasets (Richter et al. 2017). While labelling synthetically generated datasets involve little annotation effort, models trained on them do not always generalize well to the real world due to the domain gap between the real and the synthetic images. Alternative training

strategies such as semi-, self-, or weak-supervision that require simpler/fewer labels have also been proposed. Previous works on weakly-supervised semantic segmentation have used image-level annotations (Hou et al. 2017; Huang et al. 2018; Kwak et al. 2017; Lee et al. 2019; Redondo-Cabrera et al. 2019; Wang et al. 2018b; Zeng et al. 2019), points/clicks (Bearman et al. 2016), scribbles (Lin et al. 2016; Tang et al. 2018a; Tang et al. 2018b; Wang et al. 2019), bounding box annotations (Dai et al. 2015; Hu et al. 2018; Papandreou et al. 2015; Rajchl et al. 2016; Song et al. 2019; Zhao et al. 2018) and adversarial training (Arandjelović and Zisserman 2019; Hung et al. 2018). We take a closer look at some of these methods and categorize them based on the labels required and their methodology.

Image-level labels Deep learning approaches that use image-level labels typically train a classification model first to recover coarse class activation maps (Zhou et al. 2016), which describe class-discriminative image regions. The predicted class activation maps are then used as ‘seeds’ for optimization methods that grow the coarse activation maps to larger pseudo-segmentation maps. A number of optimization methods such as superpixelization (Kwak et al. 2017), deep-seeded region growing (Huang et al. 2018), conditional random fields (Redondo-Cabrera et al. 2019; Zeng et al. 2019), and combinatorial grouping (Pont-Tuset et al. 2016) have been proposed. Finally, the pseudo segmentation maps are used as ground truth to train the segmentation model (Li et al. 2018). Some approaches additionally employ a class-agnostic saliency estimation model (Chaudhry et al. 2017; Wang et al. 2018b; Zeng et al. 2019) to capture *objectness* of pixels, and others employ Expectation-Maximization (EM) (Hou et al. 2017) to iteratively refine the pseudo ground truth and the parameters of the segmentation model.

Scribble annotations While per-pixel image annotation can take several minutes, scribble annotation can take just around 8 seconds (Caelles et al. 2018). Scribble annotation has also been used for interactive image segmentation (Rother et al. 2004). Lin et al. (2016) generates dense segmentation first using a CRF model (Krähenbühl and Koltun 2011) on the weakly annotated scribbles, which is then further refined by alternately optimizing using a neural network and CRF energy function. BPG (Wang et al. 2019) tried to utilize more information from the encoder using the perception refinement network. It also uses extra edge information to refine the edge details by considering an auxiliary network. Motivated by the general ideas in semi-supervised learning, Tang et al. (2018a) combined partial cross-entropy loss on scribbles with the normalized cut. Tang et al. (2018b) extends this further

by evaluating several other regularized losses based on Potts/CRF (Krähenbühl and Koltun 2011), normalized cut (Shi and Malik 2000) and KernelCut regularizer (Tang et al. 2016).

Bounding box annotations Bounding box annotations yield high-quality ground truth at a small cost. According to Bearman et al. (2016) and Everingham et al. (2010), per-pixel labelling takes over 4 minutes per image compared to ~ 7 seconds (35x faster) for annotating bounding boxes (Papadopoulos et al. 2017). Furthermore, large datasets with bounding box annotations containing over 9 million images are publicly available (Kuznetsova et al. 2018). Weakly supervised approaches using bounding box annotations are more accurate when compared to methods that use only image-level labels. SOTA segmentation result on VOC using bounding box annotations (Song et al. 2019) outperforms the SOTA methods using image-level labels (Lee et al. 2019) by $\sim 4\%$.

The availability of bounding box annotations also alleviates the need to estimate class activation maps for localizing objects of interest in the case of image-level labels. The bounding boxes serve as crude segmentation masks which are refined using heuristic cues (Khoreva et al. 2017a) and graph-based optimization algorithms such as GrabCut (Rother et al. 2004) or mean-field inference (Krähenbühl and Koltun 2011) on a densely-connected conditional random field (Papandreou et al. 2015; Rajchl et al. 2016; Song et al. 2019). Previous works have used the refined masks for training the segmentation model (Li et al. 2018). Some of these approaches additionally use EM (Dai et al. 2015; Khoreva et al. 2017a; Papandreou et al. 2015; Zhao et al. 2018) for iterative refinement of the ground truth and model parameters. However, training models with EM is time-consuming and is prone to the propagation of errors over iterations.

Multi-task learning Hu et al. (2018) train an object detector and segmentation model simultaneously using the Mask-RCNN framework (He et al. 2017) assuming a closed-form relationship between parameters of the detection and segmentation branches. While Hu et al. (2018) can benefit from the advantages of multi-tasking, it requires (a) (limited) training data with per-pixel annotations, (b) a region of interest (RoI) proposal method, and (c) uses the RoI warping module which captures local context alone.

2.3 Classifier calibration

Deep neural networks have dominated computer vision and machine learning in recent years, and this has led to their widespread deployment in real-world systems (Cao et al. 2018; Chen et al. 2018a; Kamilaris and Prenafeta-Boldú 2018; Ker et al. 2018; Wang et al. 2018a). However, many current multi-class classification networks in particular are poorly calibrated, in the sense that the probability values that they associate with the class labels they predict overestimate the likelihood of those class labels being correct in the real world. This is a major problem, since if networks are routinely overconfident, then downstream components cannot trust their predictions. The underlying cause is hypothesized to be that these networks' high capacity leaves them vulnerable to overfitting on the negative log-likelihood (NLL) loss they conventionally use during training (Guo et al. 2017). Given the importance of this problem, numerous work trying to address it have been proposed.

2.3.1 Calibrating classical models

Much of the recent work has been inspired by approaches that were not originally formulated in a deep learning context but for SVMs, decision trees, naive Bayesian classifiers, *etc.* Platt scaling (Platt 1999) proposed a post-hoc calibration method by training a parametrized sigmoid function on top of the logit outputs. Zadrozny and Elkan (2001) proposed using the histogram method to obtain calibrated probability estimates from a naive Bayesian classifier by binning the output scores. It also smoothed the decision tree outputs using m-estimation to make them calibrated. The issue with the binning method is that it is not easy to decide the number of bins across datasets, especially if the dataset is unbalanced (Zadrozny and Elkan 2002). Isotonic regression (Zadrozny and Elkan 2002) is a non-parametric form of regression in which the function is chosen from the class of all isotonic (*i.e.*, non-decreasing) functions. For this to be effective for calibration, they assume that a classifier ranks the samples for a class correctly, and thus the mapping from scores into probabilities would be non-decreasing, which they can learn using isotonic regression. It can be viewed as a binning algorithm where the position of the boundaries and the size of the bins are chosen according to how well the classifier ranks the examples. Bayesian binning and averaging (Naeini et al. 2015; Naeini and Cooper 2016) refines the histogram-binning calibration method by using multiple different binnings and their combination to yield more robust calibrated predictions.

2.3.2 Calibrating deep learning

As deep learning has become more dominant, however, various work have begun to directly target the calibration of deep networks. In this thesis, we specifically focused on loss functions to improve calibration but there have been several other work using orthogonal approaches, like using data augmentation to improve robustness or estimating the uncertainty of deep neural networks, specifically using bayesian methods or deep ensembles. We briefly discuss all of them here.

Temperature scaling Guo et al. (2017) have popularised a modern variant of Platt scaling (Platt 1999) known as *temperature scaling*, which works by dividing a network’s logits by a scalar $T > 0$ (learnt on a validation subset) prior to performing softmax. Temperature scaling has the desirable property that it can improve the calibration of a network without in any way affecting its accuracy. However, whilst its simplicity and effectiveness have made it a popular network calibration method, it does have downsides. For example, whilst it scales the logits to reduce the network’s confidence in incorrect predictions, this also slightly reduces the network’s confidence in predictions that were correct (Kumar et al. 2018). Moreover, it is known that temperature scaling does not calibrate a model under data distribution shift (Ovadia et al. 2019).

Different loss functions Kumar et al. (2018) initially eschew temperature scaling in favour of minimising a differentiable proxy for calibration error at training time, called Maximum Mean Calibration Error (MMCE), although they do later also use temperature scaling as a post-processing step to obtain better results than cross-entropy followed by temperature scaling (Guo et al. 2017). Another loss function is brier loss (Brier 1950) which is a squared error between the predicted softmax vector and the one-hot ground truth encoding. It can be decomposed into calibration and refinement terms (DeGroot and Fienberg 1983)) which makes it quite effective in calibration. While, Müller et al. (2019) propose training models on cross-entropy loss with smoothed target labels $\mathbf{s}$ instead of one-hot labels $\mathbf{q}$, and show that label smoothing has a very favourable effect on model calibration. Given the one-hot ground-truth distribution $\mathbf{q}$ and a smoothing factor α (hyperparameter), the smoothed vector $\mathbf{s}$ is obtained as $\mathbf{s} = (1 - \alpha)\mathbf{q} + \alpha(1 - \mathbf{q})/(K - 1)$, where K is the number of classes.

Data augmentation Thulasidasan et al. (2019) further shows that label smoothing as a result of the data augmentation technique, mixup (Zhang et al. 2018b), results in significantly better calibrated deep neural networks than those trained in a regular fashion. It simply involves using a convex combination of a pair of data samples along with their labels for training the networks. Hendrycks et al. (2020) proposes AugMix which creates multiple augmentations of the same image and then mixes them, like mixup. The augmented images thus obtained are used with Jensen-Shannon Divergence consistency loss to smoothen the network’s output and make it consistent across a diverse range of inputs. It, therefore, achieves good calibration even as the data distribution shifts.

Bayesian methods There have been several work in the uncertainty estimation of classifier neural networks. Bayesian neural networks usually require a prior distribution on the weights from which the posterior distribution is estimated given the training data (Lampinen and Vehtari 2001). Thus it brings some form of stochasticity to the prediction, which can help estimate some form of uncertainty in the process. Further, taking an expectation under the posterior distribution on weights is equivalent to using an ensemble of an uncountably infinite number of neural networks (Blundell et al. 2015). However, exact Bayesian neural network inference is computationally intractable as the functional form of a neural network does not allow exact integration, therefore several approximations have been developed. Blundell et al. (2015) uses re-parametrisation trick (Kingma and Welling 2014) to compute variational free energy (Neal and Hinton 1998) and considers monte carlo sampling to evaluate the expectations to get a backpropagation-like algorithm. However, it still has a huge computational cost and requires double the number of parameters (μ and σ). Gal and Ghahramani (2016) proposed MC-Dropout where dropout (Srivastava et al. 2014) is used in the network at test time. They show that this network is mathematically equivalent to approximate probabilistic deep Gaussian process (Damianou and Lawrence 2013). For inference, it simply does several forward passes and their outputs are averaged. This makes the uncertainty estimation computationally efficient. Mandt et al. (2017) showed that under several simplifying assumptions running SGD with a constant learning rate is equivalent to sampling from a Gaussian distribution centred at the minimum of the loss. Stochastic Weight Averaging (Izmailov et al. 2018) simply averages the network parameters obtained along the trajectory of SGD to get a better-generalized model. SWA-Gaussian (SWAG) (Maddox et al. 2019) defines a Gaussian posterior approximation over neural network weights. It fits the Gaussian using Stochastic Weight Averaging (Izmailov et al. 2018) as the mean

and a low rank plus diagonal approximation to the covariance from the SGD iterate as the covariance. They show that SWAG can provide well-calibrated uncertainty estimates for neural networks. Ashukha et al. (2020) points out the pitfalls of existing metrics for uncertainty estimation and proposes using calibrated log-likelihood which is a log-likelihood after temperature scaling for fairer comparison across models. Under this metric, however, conventional non-Bayesian networks seem to match the performance of the Bayesian ones, *e.g.*, Variational inference (Blundell et al. 2015), MC-Dropout (Gal and Ghahramani 2016), *etc.* It also shows that most of the above-mentioned bayesian techniques are far less efficient than deep ensembles during inference and fail to produce the same level of performance as deep ensembles, which we discuss next.

Deep ensembles Lakshminarayanan et al. (2017) proposed using an ensemble of deep neural networks as an alternative to Bayesian NNs. Optimizing networks with different initializations and then averaging their predictions make the ensemble less prone to committing mistakes. It yields high-quality predictive uncertainty estimates and also produces well-calibrated uncertainty estimates which are better than approximate Bayesian NNs (Lakshminarayanan et al. 2017; Ovadia et al. 2019; Ashukha et al. 2020). It may be because most Bayesian NNs are ensembling techniques that are bounded to a single mode, or provide positively correlated samples, while Deep ensemble provides independent samples from different modes of the loss landscape, which, intuitively, should result in a more robust ensemble (Fort et al. 2019; Ashukha et al. 2020). However, when calibrating individual networks of the ensemble, specifically using mixup or label smoothing, Rahaman and Thiery (2021) finds contradictory evidence that it may result in less calibrated ensembles as averaging the networks’ predictions generally leads to less confident predictions. It shows that post-processing using temperature scaling on the ensemble output can help. A similar observation is made by Wen et al. (2021) and they propose using mixup only for classes where an individual network is overconfident before using that network in an ensemble.

3

MADGAN: Multi-Agent Diverse Generative Adversarial Networks

Abstract

We propose MAD-GAN, an intuitive generalization to the Generative Adversarial Networks (GANs) and its *conditional variants* to address the well-known problem of *mode collapse*. First, MAD-GAN is a multi-agent GAN architecture incorporating multiple generators and one discriminator. Second, to enforce that different generators capture diverse high probability modes, the discriminator of MAD-GAN is designed such that along with finding the real and fake samples, it is also required to identify the generator that generated the given fake sample. Intuitively, to succeed in this task, the discriminator must learn to push different generators towards different identifiable modes. We perform extensive experiments on synthetic and real datasets and compare MAD-GAN with different variants of GAN. We show high-quality diverse sample generations for challenging tasks such as image-to-image translation and face generation. In addition, we also show that MAD-GAN can disentangle different modalities when trained using a highly challenging diverse-class dataset (for example, the dataset with images of forests, icebergs, and bedrooms). In the end, we show its efficacy on the unsupervised feature representation task. In Appendix A.2, we introduce a similarity-based competing objective (MAD-GAN-Sim) which encourages different generators to generate diverse samples based on a user-defined similarity metric. We show its performance on the image-to-image translation, and also show its effectiveness on the unsupervised feature representation task.



Figure 3.1.1: Diverse-class data generation using MAD-GAN. The diverse-class dataset contains images from different classes/modalities (in this case, forests, icebergs, and bedrooms). Each row represents generations by a particular generator and each column represents generations for a given random noise input z . As shown, once trained using this dataset, generators of MAD-GAN can disentangle different modalities, hence, each generator can generate images from a particular modality.

3.1 Introduction

Generative models have attracted considerable attention recently. The underlying idea behind such models is to attempt to capture the distribution of high-dimensional data such as images and texts. Though these models are highly useful in various applications, it is computationally expensive to train them as they require intractable integration in a very high-dimensional space. This drastically limits their applicability. However, recently there has been considerable progress in *deep generative models* – a conglomerate of deep neural networks and generative models – as they do not explicitly require intractable integration and can be efficiently trained using a back-propagation algorithm. Two such famous examples are Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) and Variational Autoencoders (Kingma and Welling 2014).

In this work, we focus on GANs as they are known to produce sharp and plausible images. Briefly, GANs employ a generator and a discriminator where both are involved in a minimax game. The task of the discriminator is to learn the difference between *real* samples (from true data distribution p_d) and *fake* samples (from generator distribution p_g). Whereas, the task of the generator is to maximize the mistakes of the discriminator. At convergence, the generator learns to produce real-looking images. A few successful applications of GANs are video generation (Vondrick et al. 2016), image inpainting (Pathak et al. 2016), image manipulation (Zhu et al. 2016), 3D object generation (Wu et al. 2016), interactive image generation using few brush strokes (Zhu et al. 2016), image super-resolution (Ledig et al. 2017), diagrammatic abstract reasoning (Kulharia et al. 2017) and conditional GANs (Mirza

and Osindero 2014; Reed et al. 2016).

Despite the remarkable success of GAN, it suffers from the major problem of *mode collapse* (Arjovsky and Bottou 2017; Che et al. 2017; Chen et al. 2016; Metz et al. 2017; Salimans et al. 2016). Recent GAN literature has assigned *mode collapse* phenomena to inability to produce diverse samples (Lin et al. 2018), *e.g.*, GANs may miss some of the modes when trained on multimodal distributions. For instance, if trained on hand-written digits with ten modes, the mode-collapsed generator might fail to produce some of the digits. Though, theoretically, convergence guarantees the generator learning the true data distribution. However, practically, reaching the true equilibrium is difficult and not guaranteed, which potentially leads to the aforementioned problem of mode collapse. Broadly speaking, there are two schools of thought to address the issue: (1) improving the learning of GANs to reach better optima (Arjovsky and Bottou 2017; Metz et al. 2017; Salimans et al. 2016); and (2) explicitly enforcing GANs to capture diverse modes (Che et al. 2017; Chen et al. 2016; Liu and Tuzel 2016). Here we focus on the latter.

Borrowing from the multi-agent algorithm (Abadi and Andersen 2016) and coupled GAN (Liu and Tuzel 2016), we propose to use multiple generators with one discriminator. We call this framework the Multi-Agent GAN architecture, as shown in Fig. 3.1.2. In detail, similar to the standard GAN, the objective of each generator here is to maximize the mistakes of the *common* discriminator. Depending on the task, it might be useful for different generators to share information. This is done using the initial layer parameters of generators. Another reason behind sharing these parameters is the fact that initial layers capture low-frequency structures that are almost the same for a particular type of dataset (for example, faces), therefore, sharing them reduces redundant computations. However, when the dataset contains images from completely different modalities, one can avoid sharing these parameters. Naively using multiple generators may lead to the *trivial solution* where all the generators learn to generate *similar* samples. Further, the winner-takes-all approach is prone to making just one of the generators get trained while leaving the remaining generators poorly trained, which is similar to using a single generator framework. To resolve this issue and generate different visually plausible samples capturing diverse high-probability modes, we propose to modify the objective function of the discriminator. In the modified objective, along with finding the real and the fake samples, the discriminator also has to correctly identify the generator that generated the given fake sample. Intuitively, to succeed in this task, the discriminator must learn to push generations corresponding to different generators towards different identifiable modes. Combining the Multi-Agent GAN architecture with

the diversity enforcing term allows us to generate diverse plausible samples, thus the name Multi-Agent Diverse GAN (MAD-GAN).

As an example, an intuitive setting where mode collapse occurs is when a GAN is trained on a dataset containing images from different modalities/classes. For example, a diverse-class dataset containing images such as forests, icebergs, and bedrooms. This is of particular interest as it not only requires the model to disentangle intra-class variations, but it also requires inter-class disentanglement. Fig. 3.1.1 demonstrates the surprising effectiveness of MAD-GAN in this challenging setting. Generators among themselves can disentangle inter-class variations, and each generator is also able to capture intra-class variations.

In addition, we analyze MAD-GAN through extensive experiments and compare it with several variants of GAN. First, for the proof of concept, we perform experiments in controlled settings using a synthetic dataset (mixture of Gaussians), and complicated Stacked/ Compositional MNIST datasets with hand-engineered modes. In these settings, we empirically show that our approach outperforms all other GAN variants we compare with and can generate high-quality samples while capturing a large number of modes. In a more realistic setting, we show high-quality diverse sample generations for the challenging tasks of *image-to-image translation* (Isola et al. 2017) (conditional GAN) and *face generation* (Chen et al. 2016; Radford et al. 2015). Using the SVHN dataset (Netzer et al. 2011), we also show the efficacy of our framework for learning the feature representation in an unsupervised setting.

We also provide a theoretical analysis of this approach and show that the proposed modification in the objective of the discriminator allows generators to learn together as a mixture model where each generator represents a mixture component. We show that at convergence, the global optimum value of $-(k + 1) \log(k + 1) + k \log k$ is achieved, where k is the number of generators.

3.2 Related Work

The recent work called InfoGAN (Chen et al. 2016) proposed an information-theoretic extension to GANs to address the problem of mode collapse. Briefly, InfoGAN disentangles the latent representation by assuming a factored representation of the latent variables. To enforce that the generator learns factor-specific generations, InfoGAN maximizes the mutual information between the factored latent and the generator distribution. Che et al. (2017) proposed a mode regularized GAN (ModeGAN) which uses an encoder-decoder paradigm. The basic idea behind ModeGAN is that if a sample from the true data distribution p_d belongs

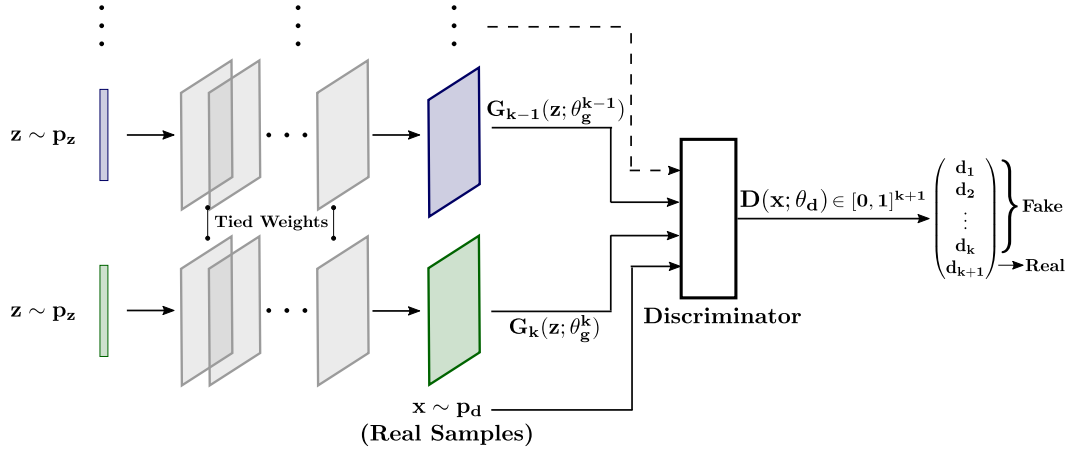


Figure 3.1.2: Multi-Agent Diverse GAN (MAD-GAN). The discriminator outputs $k + 1$ softmax scores signifying the probability of its input sample being from either one of the k generators or the real distribution.

to a particular mode, then the sample generated by the generator (fake sample) when the true sample is passed through the encoder-decoder is likely to belong to the same mode. ModeGAN assumes that there exist enough true samples from a mode for the generator to be able to capture it. Another work by Metz et al. (2017) proposed a surrogate objective for the update of the generator with respect to the unrolled optimization of the discriminator (UnrolledGAN) to address the issue of convergence of the training process of GANs. This improves the training process of the generator which in turn allows the generator to explore better coverage to true data distribution.

Liu and Tuzel (2016) presented Coupled GAN, a method for training two generators with shared parameters to learn the joint distribution of the data. The shared parameters guide both generators towards similar subspaces but since they are trained independently on two domains, they promote diverse generations. Durugkar et al. (2017) proposed a model with multiple discriminators whereby an ensemble of multiple discriminators has been shown to stabilize the training of the generator by guiding it to produce better samples.

W-GAN (Arjovsky et al. 2017) is a recent technique that employs integral probability metrics based on the earth mover distance rather than the JS-divergences that the original GAN uses. BEGAN (Berthelot et al. 2017) builds upon W-GAN using an autoencoder-based equilibrium enforcing technique alongside the Wasserstein distance. DCGAN (Radford et al. 2015) was a seminal technique that used a fully convolutional generator and discriminator for the first time along with the introduction of batch normalization thus stabilizing the training procedure, and was able to generate compelling generations. GoGAN (Juefei-Xu et al. 2017)

introduced a training procedure for the training of the discriminator using a maximum margin formulation alongside the earth mover distance based on the Wasserstein-1 metric.

Concurrent to our work, Lin et al. (2018) propose using packed or concatenated images of the same type (real or generated) as input to the discriminator to alleviate the problem of mode-collapse. Intuitively, packing helps the discriminator detect mode collapse because a lack of diversity is more obvious in a set of samples than in a single sample. The authors also provide an analytical definition of mode-collapse, and they show that mode-collapse leads to a heavily penalized generator due to larger loss using a 0-1 loss assumption. Although their empirical results are promising and they are able to get diverse generations, it is not easy to use their method to get diverse generations in the image-to-image translation task which follows a delta distribution.

Arora et al. (2017) introduced a technique and theoretical formulation stating the importance of multiple generators and discriminators to completely model the data distribution. In terms of employing multiple generators, our work is closest to Arora et al. (2017), Liu and Tuzel (2016), and Ghosh et al. (2016). However, while using multiple generators, our method explicitly enforces them to capture diverse modes.

3.3 Preliminaries

Here we present a brief review of GANs (Goodfellow et al. 2014). Given a set of samples $\mathcal{D} = (x_i)_{i=1}^n$ from the true data distribution p_d , the GAN learning problem is to obtain the optimal parameters θ_g of a generator $G(z; \theta_g)$ that can sample from an approximate data distribution p_g , where $z \sim p_z$ is the prior input noise (*e.g.*, samples from a normal distribution). In order to learn the optimal θ_g , the GAN objective (Eq. (3.1)) employs a discriminator $D(x; \theta_d)$ that learns to differentiate between a *real* (from p_d) and a *fake* (from p_g) sample x . The overall GAN objective is:

$$\min_{\theta_g} \max_{\theta_d} V(\theta_d, \theta_g) := \mathbb{E}_{x \sim p_d} \log D(x; \theta_d) + \mathbb{E}_{z \sim p_z} \log (1 - D(G(z; \theta_g); \theta_d)) \quad (3.1)$$

The above objective is optimized in a block-wise manner where θ_d and θ_g are optimized one at a time while fixing the other. For a given sample x (either from p_d or p_g) and the parameter θ_d , the function $D(x; \theta_d) \in [0, 1]$ produces a score that represents the probability of x belonging to the true data distribution p_d (or probability of it being real). The objective

of the discriminator is to learn parameters θ_d that maximize this score for the true samples (from p_d) while minimizing it for the fake ones $\tilde{x} = D(z; \theta_g)$ (from p_g). In the case of a generator, the objective is to minimize $\mathbb{E}_{z \sim p_z} \log(1 - D(G(z; \theta_g); \theta_d))$, equivalently maximize $\mathbb{E}_{z \sim p_z} \log D(G(z; \theta_g); \theta_d)$. Thus, the generator learns to maximize the scores for the fake samples (from p_g), which is exactly the opposite of what the discriminator is trying to achieve. In this manner, the generator and the discriminator are involved in a minimax game where the task of the generator is to maximize the mistakes of the discriminator. Theoretically, at equilibrium, the generator learns to generate real samples, which means $p_g = p_d$.

3.4 Multi-Agent Diverse GAN

In the GAN objective, one can argue that the task of a generator is much harder than that of the discriminator as it has to produce real-looking images to maximize the mistakes of the discriminator. This, along with the minimax nature of the objective raises several challenges for GANs (Arjovsky and Bottou 2017; Che et al. 2017; Chen et al. 2016; Metz et al. 2017; Salimans et al. 2016): (1) mode collapse; (2) difficult optimization; and (3) trivial solution. In this work, we propose a new framework to address the first challenge of *mode collapse* by increasing the capacity of the generator while using well-known tricks to partially avoid other challenges (Arjovsky and Bottou 2017).

Briefly, we propose a *Multi-Agent GAN architecture* that employs multiple generators and one discriminator to generate different samples from high-probability regions of the true data distribution. In addition, theoretically, we show that our formulation allows generators to act as a mixture model with each generator capturing one component.

3.4.1 Multi-Agent GAN Architecture

Here we describe our proposed architecture (Fig. 3.1.2). It involves k generators and one discriminator. In the case of homogeneous data (all the images belong to the same class, *e.g.*, faces or birds), we allow all the generators to share information by tying most of the initial layer parameters. This is essential to avoid redundant computations as initial layers of a generator capture low-frequency structures which are almost the same for a particular type of dataset. This also allows different generators to converge faster. However, in the case of *diverse-class* data (*e.g.*, dataset with a mixture of different classes such as forests, icebergs, *etc.*), it is necessary to avoid sharing these parameters to allow each generator

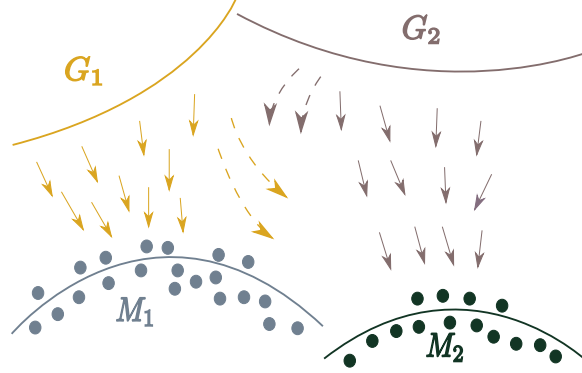


Figure 3.4.1: Visualization of different generators getting pushed towards different modes. Here, M_1 and M_2 could be a cluster of modes where each cluster itself contains many modes. The arrows abstractly represent generator-specific gradients for the purpose of building intuition.

to capture content-specific structures. Thus, the extent to which one should share these parameters depends on the task at hand.

More specifically, given $z \sim p_z$ for the i -th generator, similar to the standard GAN, the first step involves generating a sample (for example, an image) $\tilde{x}_i$. Since each generator receives the same latent input sampled from the same distribution, naively using this simple approach may lead to the *trivial solution* where all the generators learn to generate similar samples. In what follows, we propose an intuitive solution to avoid this issue and allow the generators to capture diverse modes.

3.4.2 Enforcing Diverse Modes

Inspired by the discriminator formulation for the semi-supervised learning (Salimans et al. 2016), we use a generator identification-based objective function that, along with minimizing the score $D(\tilde{x}; \theta_d)$, requires the discriminator to identify the generator that generated the given fake sample $\tilde{x}$. In order to do so, as opposed to the standard GAN objective function where the discriminator outputs a scalar value, we modify it to output $k + 1$ soft-max scores. In more detail, given the set of k generators, the discriminator produces a soft-max probability distribution over $k + 1$ classes. The score at $(k + 1)$ -th index ($D_{k+1}(\cdot)$) represents the probability that the sample belongs to the true data distribution and the score at $j \in \{1, \dots, k\}$ -th index represents the probability of it being generated by the j -th generator. Under this setting, while learning θ_d , we optimize the cross-entropy between the soft-max output of the discriminator and the Dirac delta distribution $\delta \in \{0, 1\}^{k+1}$, where for $j \in \{1, \dots, k\}$, $\delta(j) = 1$ if the sample belongs to the j -th generator, otherwise

$\delta(k+1) = 1$. Thus, the objective of the discriminator, which is optimizing θ_d while keeping θ_g constant (refer Eq. (3.1)), is modified to:

$$\max_{\theta_d} \mathbb{E}_{x \sim p} H(\delta, D(x; \theta_d))$$

where, $Supp(p) = \cup_{i=1}^k Supp(p_{g_i}) \cup Supp(p_d)$ and $H(\cdot, \cdot)$ is the negative of the cross entropy function. Intuitively, in order to correctly identify the generator that produced a given fake sample, the discriminator must learn to push different generators towards different identifiable modes. However, the objective of each generator remains the same as in the standard GAN. Thus, for the i -th generator, the objective is to minimize the following:

$$\mathbb{E}_{x \sim p_d} \log D_{k+1}(x; \theta_d) + \mathbb{E}_{z \sim p_z} \log(1 - D_{k+1}(G_i(z; \theta_g^i); \theta_d))$$

To update the parameters, the gradient for each generator is simply computed as $\nabla_{\theta_g^i} \log(1 - D_{k+1}(G_i(z; \theta_g^i); \theta_d))$. Please note that here every generator's objective is to fool the discriminator by generating realistic images, they have no incentive to fool the discriminator by generating images like each other. Notice that all the generators in this case can be updated in parallel. For the discriminator, given $x \sim p$ (can be real or fake) and corresponding δ , the gradient is $\nabla_{\theta_d} \log D_j(x; \theta_d)$, where $D_j(x; \theta_d)$ is the j -th index of $D(x; \theta_d)$ for which $\delta(j) = 1$. Therefore, using this approach requires *very minor modifications to the standard GAN optimization algorithm* and can be easily used with different variants of GAN. An intuitive visualization is shown in Fig. 3.4.1.

Theorem 1 shows that the above objective function actually allows generators to form a mixture model where each generator represents a mixture component and the global optimum of $-(k+1) \log(k+1) + k \log k$ is achieved when $p_d = \frac{1}{k} \sum_{i=1}^k p_{g_i}$. Notice that, at $k = 1$, which is the case with one generator, we obtain exactly the same Jensen-Shannon divergence-based objective function as shown in Goodfellow et al. (2014) with the optimal value of $-\log 4$.

Theorem 1. *Given the optimal discriminator, the objective for training the generators boils down to minimizing*

$$\begin{aligned} & KL\left(p_d(x) || p_{avg}(x)\right) + k KL\left(\frac{1}{k} \sum_{i=1}^k p_{g_i}(x) || p_{avg}(x)\right) \\ & - (k+1) \log(k+1) + k \log k \end{aligned} \quad (3.2)$$

where, $p_{avg}(x) = \frac{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}{k+1}$. The above objective function obtains its global minimum if $p_d = \frac{1}{k} \sum_{i=1}^k p_{g_i}$ with the objective value of $-(k+1) \log(k+1) + k \log k$.

Proof. The joint objective of all the generators is to minimize the following:

$$\mathbb{E}_{x \sim p_d} \log D_{k+1}(x) + \sum_{i=1}^k \mathbb{E}_{x \sim p_{g_i}} \log(1 - D_{k+1}(x))$$

Using Corollary 1, we substitute the optimal discriminator in the above equation and obtain:

$$\begin{aligned} & \mathbb{E}_{x \sim p_d} \log \left[\frac{p_d(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)} \right] + \\ & \sum_{i=1}^k \mathbb{E}_{x \sim p_{g_i}} \log \left[\frac{\sum_{i=1}^k p_{g_i}(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)} \right] \\ & = \mathbb{E}_{x \sim p_d} \log \left[\frac{p_d(x)}{p_{avg}(x)} \right] + k \mathbb{E}_{x \sim p_g} \log \left[\frac{p_g(x)}{p_{avg}(x)} \right] \\ & - (k+1) \log(k+1) + k \log k \end{aligned} \quad (3.3)$$

where, $p_g = \frac{\sum_{i=1}^k p_{g_i}}{k}$ and $p_{avg}(x) = \frac{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}{k+1}$. Note that, Eq. (3.3) is exactly the same as Eq. (3.2). When $p_d = \frac{\sum_{i=1}^k p_{g_i}}{k}$, both the KL terms become zero and the global minimum is achieved. $\square$

Corollary 1. *For fixed generators, the optimal distribution learned by the discriminator D has the following form:*

$$\begin{aligned} D_{k+1}(x) &= \frac{p_d(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}, \\ D_i(x) &= \frac{p_{g_i}(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}, \forall i \in \{1, \dots, k\}. \end{aligned}$$

where, $D_i(x)$ represents the i -th index of $D(x; \theta_d)$, p_d the true data distribution, and p_{g_i} the distribution learned by the i -th generator.

Proof. For fixed generators, the objective function of the discriminator is to maximize

$$\mathbb{E}_{x \sim p_d} \log D_{k+1}(x) + \sum_{i=1}^k \mathbb{E}_{x_i \sim p_{g_i}} \log D_i(x_i)$$

where, $\sum_{i=1}^{k+1} D_i(x) = 1$ and $D_i(x) \in [0, 1], \forall i$. The above equation can be written as:

$$\begin{aligned} & \int_x p_d(x) \log D_{k+1}(x) dx + \sum_{i=1}^k \int_x p_{g_i}(x) \log D_i(x) dx \\ & = \int_{x \in p} \sum_{i=1}^{k+1} p_i(x) \log D_i(x) dx \end{aligned} \quad (3.4)$$

where, $p_{k+1}(x) := p_d(x)$, $p_i(x) := p_{g_i}(x)$, $\forall i \in \{1, \dots, k\}$, and $Supp(p) = \bigcup_{i=1}^{k+1} Supp(p_i)$. Therefore, for a given x , the optimum of the objective function defined in Eq. (3.4) with constraints defined above can be obtained using Proposition 1. $\square$

Proposition 1. *Given $\mathbf{y} = (y_1, \dots, y_n)$, $y_i \geq 0$, and $a_i \in \mathbb{R}$, the optimal solution for the objective function defined below is achieved at $y_i^* = \frac{a_i}{\sum_{i=1}^n a_i}$, $\forall i$*

$$\max_{\mathbf{y}} \sum_{i=1}^n a_i \log y_i, \quad s.t. \quad \sum_{i=1}^n y_i = 1$$

Proof. The Lagrangian of the above problem is:

$$L(\mathbf{y}, \lambda) = \sum_{i=1}^n a_i \log y_i + \lambda \left(\sum_{i=1}^n y_i - 1 \right)$$

Differentiating w.r.t y_i and λ , and equating to zero,

$$\frac{a_i}{y_i} + \lambda = 0, \quad \sum_{i=1}^n y_i - 1 = 0$$

Solving the above two equations, we obtain $y_i^* = \frac{a_i}{\sum_{i=1}^n a_i}$. $\square$

3.5 Experiments

We present an extensive quantitative and qualitative analysis of MAD-GAN on various synthetic and real-world datasets. First, we use a simple 1D mixture of Gaussians and also Stacked/Compositional MNIST dataset (1000 modes) to compare MAD-GAN with several known variants of GANs, such as DCGAN (Radford et al. 2015), WGAN (Arjovsky et al. 2017), BEGAN (Berthelot et al. 2017), GoGAN (Juefei-Xu et al. 2017), Unrolled GAN (Metz et al. 2017), Mode-Reg GAN (Che et al. 2017) and InfoGAN (Chen et al. 2016). Furthermore, we created another baseline, called *MA-GAN (Multi-Agent GAN)*, which is a trivial extension of GAN with multiple generators and one discriminator. As opposed to MAD-GAN, MA-GAN has a simple Multi-Agent architecture without modifications to the objective of the discriminator. This comparison allows us to understand the effect of explicitly enforcing diversity in the objective of the MAD-GAN. We use KL-divergence (Kullback and Leibler 1951) and the number of modes recovered (Che et al. 2017) as the criterion for comparisons and show superior results compared to all the other methods. Additionally, we show diverse generations for the challenging tasks of *image-to-image translation* (Isola et al. 2017), *diverse-class data generation*, and face generation. It is non-trivial to devise a metric to evaluate

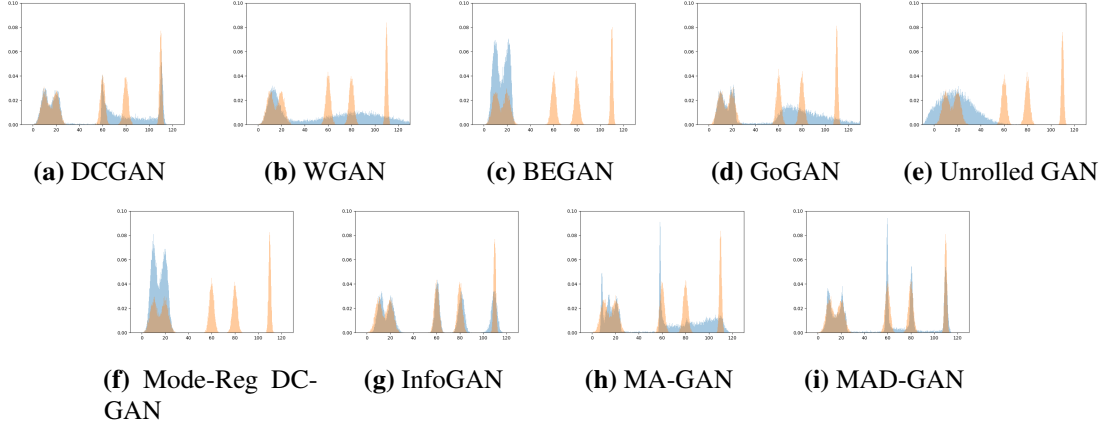


Figure 3.5.1: A toy example to understand the behaviour of different GAN variants in order to compare with MAD-GAN (each method was trained for 198000 iterations). The orange bars show the density estimate of the training data and the blue ones for the generated data points. After careful cross-validation, we chose the bin size of 0.1 to get smooth plots based on the number of samples. Changing the bin size does not change the order of the GAN variants.

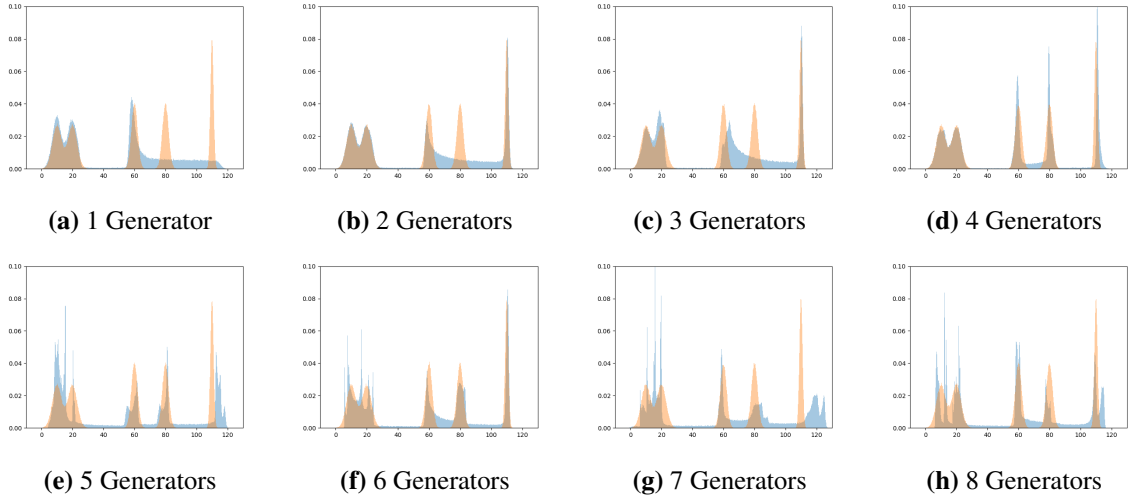


Figure 3.5.2: A toy example to understand the behaviour of MAD-GAN with a different number of generators (each method was trained for 198000 iterations). The orange bars show the density estimate of the training data and the blue ones for the generated data points. After careful cross-validation, we chose the bin size of 0.1.

GAN Variants	Chi-square($\times 10^5$)	KL-Div
DCGAN (Radford et al. 2015)	0.90	0.322
WGAN (Arjovsky et al. 2017)	1.32	0.614
BEGAN (Berthelot et al. 2017)	1.06	0.944
GoGAN (Juefei-Xu et al. 2017)	2.52	0.652
Unrolled GAN (Metz et al. 2017)	3.98	1.321
Mode-Reg DCGAN (Che et al. 2017)	1.02	0.927
InfoGAN (Chen et al. 2016)	0.83	0.21
MA-GAN	1.39	0.526
MAD-GAN (Our)	0.24	0.145

Table 3.5.1: Synthetic experiment on 1D GMM (Fig. 3.5.1). This table does not report an in-depth statistical analysis using different training runs, however, MAD-GAN was consistently superior to the baselines in all the iterations we tried during model development for this experiment.

# Generators	Chi-square ($\times 10^7$)	KL-Div
1	1.27	0.57
2	1.38	0.42
3	3.15	0.71
4	0.39	0.28
5	3.05	0.88
6	0.54	0.29
7	0.97	0.78
8	4.83	0.68

Table 3.5.2: Synthetic experiment with different number of MAD-GAN generators (same setup as in Fig. 3.5.1). This table does not report an in-depth statistical analysis using different training runs, however, MAD-GAN with four generators was consistently superior to MAD-GAN with the other number of generators in all the iterations we tried during model development for this experiment.

diversity on these high-quality generation tasks, so we perform a qualitative assessment. Note that, the image-to-image translation objective is known to learn the delta distribution, thus, it is agnostic to the input noise vector. However, we show that MAD-GAN can produce highly plausible diverse generations for this task. In the end, we show the efficacy of MAD-GAN in an unsupervised feature representation learning task. We provide a detailed overview of the architectures, datasets, and parameters used in our experiments in Appendix A.3.

In the case of InfoGAN (Chen et al. 2016), we varied the dimension of the categorical variable, depicting the number of modes, to obtain the best cross-validated results.

3.5.1 Non-Parametric Density Estimation

To understand the behavior of MAD-GAN and different state-of-the-art GAN models, we first perform a very simple synthetic experiment, much easier than generating high-dimensional complex images. We consider a distribution of 1D GMM (Bishop 2007) having five mixture components with modes at 10, 20, 60, 80 and 110, and standard deviations of 3, 3, 2, 2 and 1, respectively. While the first two modes overlap significantly, the fifth mode stands isolated as shown in Fig. 3.5.1. We train different GAN models using 200,000 samples from this distribution and generate 65,536 data points from each model. To compare the learned distribution with the ground truth distributions, we first estimate them using bins over the data points and create the histograms. These histograms are carefully created using different bin sizes and the best bin (found to be 0.1) is chosen. Then, we use the Chi-square distance and the KL divergence to compute the distance between the two histograms. From Fig. 3.5.1 and Tab. 3.5.1, it is evident that MAD-GAN can capture all the clustered modes which include significantly overlapped modes as well. MAD-GAN obtains the minimum value in terms of both Chi-square distance and the KL-divergence. In this experiment, both MAD-GAN and MA-GAN used four generators. In the case of InfoGAN, we used a 5 dimensional categorical variable, which provides the best result.

To understand the effect of varying the number of generators in MAD-GAN, we use the same synthetic experimental setup, *i.e.*, the real data distribution is the same GMM with 5 Gaussians. For a better non-parametric estimation of the distribution, we use 1 million sample points from the real distribution (instead of 65,536). We generate an equal number of points from each of the generators such that they sum up to 1 million. The results are shown in Fig. 3.5.2 and corresponding Tab. 3.5.2. It is quite clear that as the number of generators is increased up to 4, the sampling keeps getting more realistic. In the case when multiple modes are significantly overlapped/clustered, a generator can capture a cluster of modes. Therefore, for this real data distribution, 4 generators are enough to capture all the 5 modes. With 5 or more generators, all the modes were still captured, but the two overlapping modes have more than two generation peaks. This is mainly because multiple generators capture this region and all the generators (mixture components) were assigned *equal weights* during sampling.

Other work using more than one generators (Liu and Tuzel 2016; Arora et al. 2017) also use the number of generators as a hyper-parameter as knowing a-priori the number of modes in a real-world data (*e.g.*, images) in itself is an open problem.

GAN Variants	KL Div	# Modes Covered
DCGAN (Radford et al. 2015)	2.15	712
WGAN (Arjovsky et al. 2017)	1.02	868
BEGAN (Berthelot et al. 2017)	1.89	819
GoGAN (Juefei-Xu et al. 2017)	2.89	672
Unrolled GAN (Metz et al. 2017)	1.29	842
Mode-Reg DCGAN (Che et al. 2017)	1.79	827
InfoGAN (Chen et al. 2016)	2.75	840
MA-GAN	3.4	700
MAD-GAN (Our)	0.91	890

Table 3.5.3: Stacked-MNIST experiments and comparisons. Note that three generators are used for MAD-GAN.

GAN Variants	KL Div	# Modes Covered
DCGAN (Radford et al. 2015)	0.18	980
WGAN (Arjovsky et al. 2017)	0.25	1000
BEGAN (Berthelot et al. 2017)	0.19	999
GoGAN (Juefei-Xu et al. 2017)	0.87	972
Unrolled GAN (Metz et al. 2017)	0.091	1000
Mode-Reg DCGAN (Che et al. 2017)	0.12	992
InfoGAN (Chen et al. 2016)	0.47	990
MA-GAN	1.62	997
MAD-GAN (Our)	0.074	1000

Table 3.5.4: Compositional-MNIST experiments and comparisons. Note that three generators are used for MAD-GAN.

3.5.2 Stacked and Compositional MNIST

We now perform experiments on a more challenging setup, similar to Che et al. (2017) and Metz et al. (2017), to examine and compare MAD-GAN with other GAN variants. Metz et al. (2017) created a Stacked-MNIST dataset with 25,600 samples where each sample has three channels stacked together with a random MNIST digit in each of them. Thus, it creates 1000 distinct modes in the data distribution. Metz et al. (2017) used a stripped-down version of the generator and discriminator pair to reduce the modelling capacity. We do the same for fair comparisons and used the same architecture as mentioned in their paper. Similarly, Che et al. (2017) created Compositional-MNIST whereby they took 3 random MNIST digits and place them at the 3 quadrants of a 64×64 dimensional image. This also resulted in



Figure 3.5.4: Diverse generations for *edges-to-handbags* generation task. In each sub-figure, the first column is the input, columns 2-4 are generations by MAD-GAN (using three generators), and columns 5-7 are generations by InfoGAN (using three categorical codes). Clearly different generators of MAD-GAN are producing diverse results capturing different colours, textures, design patterns, *etc.* However, InfoGAN generations are visually almost the same, indicating mode collapse.



Figure 3.5.6: InfoGAN for *edges-to-handbags* task by sharing the discriminator and Q Network. In each sub-figure, the first column is the input, columns 2 – 4 are generations when input is categorical code besides conditioning image, and columns 5 – 7 are generations with noise as an additional input. The generations for both architectures are visually the same irrespective of the categorical code value, which clearly indicates that it is not able to capture diverse modes.

data distribution with 1000 hand-designed modes. The distribution of the resulting generated samples was estimated using a pretrained MNIST classifier to classify each of the digits either in the channels or the quadrants to decide the mode it belongs to.

Tables 3.5.3 and 3.5.4 provide a comparison of our method with variants of GAN in terms of KL divergence and the number of modes recovered for the Stacked and Compositional MNIST datasets, respectively. In Stacked-MNIST, as evident from Tab. 3.5.3, MAD-GAN outperforms all other variants of GAN in both criteria. Interestingly, in the case of Compositional-MNIST, as shown in Tab. 3.5.4, MAD-GAN, WGAN, and Unrolled GAN

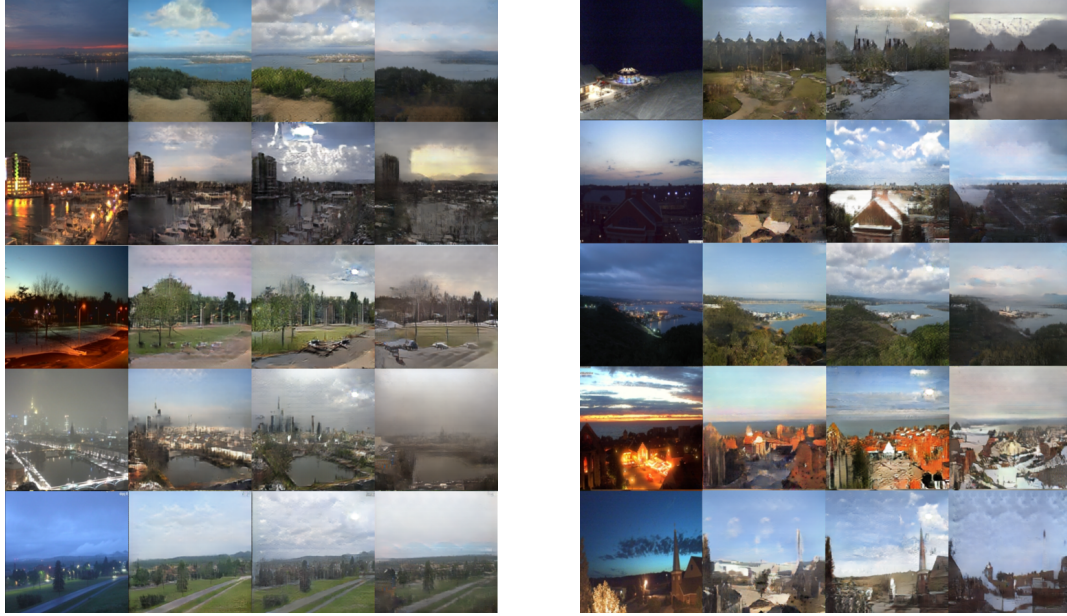


Figure 3.5.8: Diverse generations for *night-to-day* image generation task. The first column in each sub-figure represents the input. The remaining three columns show the diverse generations of three different generators of MAD-GAN (Our).

were able to recover all the 1000 modes. However, in terms of KL divergence, the distribution generated by MAD-GAN is the closest to the true data distribution.

3.5.3 Diverse Samples for Image-to-Image Translation and Comparison to InfoGAN

Here we present experimental results on the challenging task of image-to-image translation (Isola et al. 2017) which uses a conditional variant of GANs (Mirza and Osindero 2014). Conditional GAN for this task is known to learn the delta distribution, thus, generating the same image irrespective of the variations in the input noise vector. Generating diverse samples in this setting in itself is an open problem. We show that MAD-GAN can generate diverse samples in these experiments as well. We use three generators for MAD-GAN experiments and show three diverse generations. Note that, we do not claim to capture all the possible modes present in the data distribution because firstly we cannot estimate the number of modes a priori, and secondly, even if we could, we do not know how diverse the generations would be after using a certain number of generators. We follow the same approach as Isola et al. (2017) and employ patch-based conditional GAN.

We compare MAD-GAN with InfoGAN (Chen et al. 2016) in these experiments as it is closest to our approach and can be used in the image-to-image translation task. Theoretically, latent codes in InfoGAN should enable diverse generations. However, InfoGAN can only be used when the bias introduced by the categorical variables has a significant impact on the generator network. For image-to-image translation and high-resolution generations, the categorical variable does not have a sufficient impact on the generations. As will be seen shortly, we validate this hypothesis by comparing our method with InfoGAN for this task. For the InfoGAN generator, to capture three kinds of distinct modes, the categorical code is chosen to take three values. Since we are dealing with images, in this case, the categorical code is a 2D matrix in which we set one-third of the entries to 1 and the remaining to 0 for each category. The generator is fed an input image along with categorical code appended channel-wise to the image. The architecture of the Q network is the same as that of the pix2pix discriminator (Isola et al. 2017), except that the output is a vector of size 3 for the prediction of the categorical codes. Note that, we tried different variations of the categorical codes but did not observe any significant variation in the generations.

Fig. 3.5.4 shows generations by MAD-GAN and InfoGAN for the *edges-to-handbags* task, where given the edges of handbags, the objective is to generate real-looking handbags. Each MAD-GAN generator can produce meaningful images but is different from the remaining generators in terms of colour, texture, and patterns. However, InfoGAN generations are almost the same for all three categorical codes. The results shown for InfoGAN are obtained by not sharing the discriminator and Q network parameters.

To make our baseline as strong as possible, we did some more experiments with InfoGAN for the *edges-to-handbags* task. For Fig. 3.5.6, we conducted two experiments by sharing all the initial layers of the discriminator and Q network. In the first experiment, the input is the categorical code besides the conditional image. In the second experiment, noise is also added as an input. In Fig. 3.5.6, we show the results of both these experiments side by side. There are not many perceivable changes as we vary the categorical code values, as the generator learns to ignore the input noise (as was also seen in Isola et al. (2017)).

In addition, in Fig. 3.5.8, we show diverse generations for the *night-to-day* task, where given night images, the objective is to generate their corresponding day images. As can be seen, the generated day images in Fig. 3.5.8 differ in terms of lighting conditions, sky patterns, weather conditions, and many other minute yet useful cues.

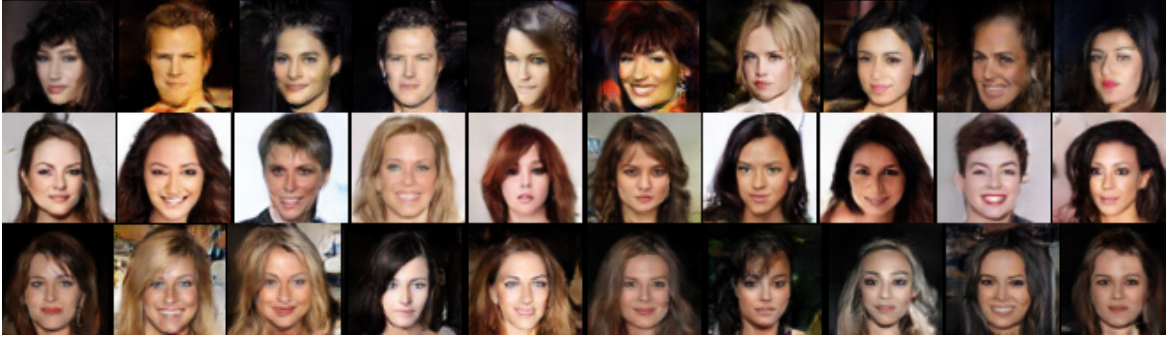


Figure 3.5.9: Face generations using MAD-GAN. Each generator employed is DCGAN. Each row represents a generator. Each column represents generations for a given random noise input z . Note that, the first generator is generating faces pointing to the left. The second generator is generating female faces with long hair, while the third generator generates images with light backgrounds.



Figure 3.5.11: Face generations using MAD-GAN. Each sub-figure shows generations by a single generator. The first generator is generating faces with very dark backgrounds. The second one is generating female faces with long hair in light backgrounds, while the third one is generating faces with coloured backgrounds and casual looks (based on facial direction and expression).

3.5.4 Diverse-Class Data Generation

To further explore the mode capturing capacity of MAD-GAN, we experimented with a much more challenging task of diverse-class data generation. In detail, we trained MAD-GAN (three generators) on a combined dataset consisting of various highly diverse images such as *islets*, *icebergs*, *broadleaf-forest*, *bamboo-forest*, and *bedroom*, obtained from the Places dataset (Zhou et al. 2017a). Images were randomly selected from each of them, creating a training dataset of 24,000 images. The generators have the same architecture as that of DCGAN. In this case, as the images in the dataset belong to different classes, we did not share the generator parameters. As shown in Fig. 3.1.1, to our surprise, we

found that even in this highly challenging setting, the generations from different generators belong to different classes. This indicates that the generators in MAD-GAN can disentangle inter-class variations. In addition, each generator for different noise inputs can generate diverse samples, indicating intra-class diversity.

3.5.5 Diverse Face Generation

Here we show diverse face generations (CelebA dataset) using MAD-GAN where we use DCGAN (Radford et al. 2015) as each of our three generators. Again, we use the same setting as provided in DCGAN. The high-quality face generations are shown in Fig. 3.5.9. We show additional generations in Fig. 3.5.11.

3.5.6 Unsupervised Representation Learning

Similar to DCGAN (Radford et al. 2015), we train our framework using SVHN dataset (Netzer et al. 2011). The trained discriminator is used to extract features. Using these features, we train an SVM for the classification task. For the MAD-GAN, with three generators, we obtained a misclassification error of 17.5% which is almost 5% better than the results reported by DCGAN (22.48%). This indicates that our framework can learn a better feature space in an unsupervised setting.

3.6 Conclusion

We presented a very simple and effective framework, Multi-Agent Diverse GAN (MAD-GAN), for generating diverse and meaningful samples. We showed the efficacy of our approach and compared it with various variants of GAN to show that it captures diverse modes while producing high-quality samples. We presented a theoretical analysis of MAD-GAN with conditions for global optimality. Looking forward, an interesting future direction would be to estimate a priori the number of generators needed for a particular dataset. It is not clear how to do that given that we do not have access to the true data distribution. In addition, we would also like to theoretically understand the limiting cases that depend on the relationship between the number of generators and the complexity of the data distribution. Another interesting direction would be to exploit different generators such that their combinations can be used to capture diverse modes.

4

Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation

Abstract

We propose a weakly-supervised approach to semantic segmentation using bounding box annotations. Bounding boxes are treated as noisy labels for the foreground objects. We predict a per-class attention map that saliently guides the per-pixel cross-entropy loss to focus on foreground pixels and refines the segmentation boundaries. This avoids propagating erroneous gradients due to incorrect foreground labels on the background. Additionally, we learn pixel embeddings to simultaneously optimize for high intra-class feature affinity while increasing discrimination between features across different classes. Our method, Box2Seg, achieves state-of-the-art segmentation accuracy on PASCAL VOC 2012 by significantly improving the mIOU metric by 2.1% compared to previous weakly supervised approaches. Our weakly-supervised approach is comparable to the recent fully supervised methods when fine-tuned with a limited amount of pixel-level annotations. Qualitative results and ablation studies show the benefit of different loss terms on the overall performance.

4.1 Introduction

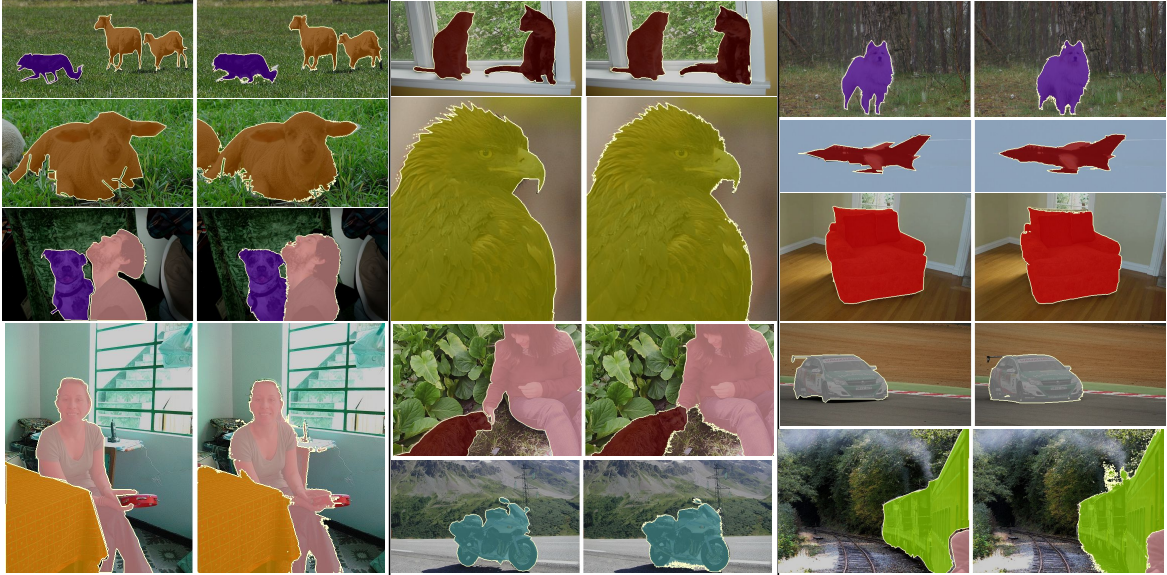


Figure 4.1.1: Our Box2Seg model is able to produce high-quality semantic segmentation using only bounding box annotations. Each image pair shows ground-truth (left) and predicted segmentation (right).

The accuracy of semantic segmentation approaches has improved significantly in recent years (Chen et al. 2017a; Chen et al. 2018b; Long et al. 2015; Noh et al. 2015; Zhang et al. 2018a; Zhang et al. 2019; Zheng et al. 2015). The mean Intersection-over-Union (mIoU) metric on the PASCAL VOC semantic segmentation benchmark has improved by over 20% in the last five years. The success of these efforts can be broadly attributed to (i) advancements in deep neural network architectures and loss functions, (ii) efficient processing (better GPUs), and (iii) the availability of large datasets of images with human-labelled per-pixel annotations (Everingham et al. 2010; Lin et al. 2014b). Improvements in network architectures and hardware capabilities benefit all deep learning tasks. However, large datasets with per-pixel semantic labels are both expensive and slow to obtain (Bearman et al. 2016; Everingham et al. 2010) (typically 4-10 minutes per image), making it challenging to scale to a large number of object categories. Consequently, even the largest semantic segmentation datasets (Lin et al. 2014b; Zhou et al. 2017b) include less than a couple of hundred object categories.

To address the scarcity of labelled data, some previous works have used synthetic datasets (Richter et al. 2017). While labelling synthetically generated datasets involves

little annotation effort, models trained on them do not always generalize well to the real world due to the domain gap between the real and the synthetic images. Alternative training strategies such as semi-, self-, or weak-supervision that require simpler/fewer labels (eg. image-level labels or bounding box annotations) have also been proposed. In this work, we show how to leverage real images with weak supervision to advance the state-of-the-art (SOTA) for semantic segmentation (refer to Fig. 4.1.1 for sample results).

Bounding box annotations yield high-quality ground truth at a small cost. According to Bearman et al. (2016) and Everingham et al. (2010), per-pixel labelling takes over 4 minutes per image compared to ~ 7 seconds (35x faster) for annotating bounding boxes (Papadopoulos et al. 2017). Furthermore, large datasets with bounding box annotations containing over 9 million images are publicly available (Kuznetsova et al. 2018). Weakly supervised approaches using bounding box annotations are more accurate when compared to methods that use only image-level labels. SOTA segmentation result on VOC using bounding box annotations (Song et al. 2019) outperforms the SOTA methods using image-level labels (Lee et al. 2019) by $\sim 4\%$. Our work, *Box2Seg*, builds upon previous approaches that use bounding box annotations.

A key intuition of our work is to consider bounding box annotations as containing *label noise* for the foreground object. Since bounding box annotation is a super-set of the actual object segmentation, this label noise is one-sided. In other words, some foreground labels are incorrectly assigned to background pixels within the bounding box. However, all foreground pixels inside the box and background pixels outside all boxes are correctly labelled. Typical fully supervised segmentation training considers the label for every pixel as correct and gradients are back-propagated from all pixels. This would be an issue for a weakly supervised segmentation algorithm. To handle this, our algorithm predicts a novel per-pixel class-specific attention map and pixel embeddings in addition to the per-pixel segmentation output. The attention map is used to modulate the per-pixel cross-entropy loss to handle label noise and reduces the propagation of incorrect gradients. Thus, the attention map allows us to automatically discover salient regions of the object within the bounding box. The attention map is regularized using a soft filling-rate constraint for each training image.

We learn discriminative feature embeddings to capture long-range pairwise relationships between pixels across an image. We pretrain these embeddings to maximize the affinity between pixels belonging to the same classes, while at the same time increasing the distance between features corresponding to different classes. During training, we define a novel loss function on pairs of pixels such that it encourages the pixel affinities to align with the

predicted segmentation probabilities. A few methods have also proposed using feature affinity as an explicit measure to improve segmentation in fully-supervised and co-segmentation settings (Chandra et al. 2017; Hsu et al. 2019; Ke et al. 2018; Liu et al. 2017; Maire et al. 2016). Affinities have also been used in some weakly supervised approaches (Ahn et al. 2019; Ahn and Kwak 2018), but they are trained in a fully-supervised manner using pseudo ground truth derived from class activation maps, as opposed to our embeddings which are trained by minimizing disagreements between them and estimated segmentation output (Sect. 4.3.4). We show how discriminative feature learning obtained using the model predictions can also be incorporated and is helpful in the context of weakly supervised semantic segmentation.

The remainder of this chapter is organized as follows. We discuss related work in Sect. 4.2. Our learning algorithm and loss functions are described in Sect. 4.3. We demonstrate SOTA results on the PASCAL VOC 2012 segmentation benchmark (Sect. 4.4), outperforming previous weakly-supervised approaches by 2.1% on the mIoU metric. We also show that our weakly supervised model serves as a good starting point for semi-supervised learning tasks, surpassing fully supervised baselines pre-trained with ImageNet (Deng et al. 2009) with only a fraction of pixel-level segmentation annotations.

4.2 Related Work

Previous works on weakly-supervised semantic segmentation have used image-level annotations (Hou et al. 2017; Huang et al. 2018; Kwak et al. 2017; Lee et al. 2019; Redondo-Cabrera et al. 2019; Wang et al. 2018b; Zeng et al. 2019), points/clicks (Bearman et al. 2016), scribbles (Lin et al. 2016; Tang et al. 2018a; Tang et al. 2018b; Wang et al. 2019), bounding box annotations (Dai et al. 2015; Hu et al. 2018; Papandreou et al. 2015; Rajchl et al. 2016; Song et al. 2019; Zhao et al. 2018) and adversarial training (Arandjelović and Zisserman 2019; Hung et al. 2018). We take a closer look at some of these methods and categorize them based on the labels required and their methodology.

Image Level Labels: Deep learning approaches that use image-level labels typically train a classification model first to recover coarse class activation maps (Zhou et al. 2016), which describe class-discriminative image regions. The predicted class activation maps are then used as ‘seeds’ for optimization methods that grow the coarse activation maps to larger pseudo-segmentation maps. A number of optimization methods such as superpixelization (Kwak et al. 2017), deep-seeded region growing (Huang et al. 2018), conditional

random fields (Redondo-Cabrera et al. 2019; Zeng et al. 2019), and combinatorial grouping (Pont-Tuset et al. 2016) have been proposed. Finally, the pseudo segmentation maps are used as ground truth to train the segmentation model (Li et al. 2018). Some approaches additionally employ a class-agnostic saliency estimation model (Chaudhry et al. 2017; Wang et al. 2018b; Zeng et al. 2019) to capture *objectness* of pixels, and others employ Expectation-Maximization (EM) (Hou et al. 2017) to iteratively refine the pseudo ground truth and the parameters of the segmentation model.

Bounding Box Labels: The availability of bounding box annotations alleviates the need to estimate class activation maps for localizing objects of interest. The bounding boxes serve as crude segmentation masks which are refined using heuristic cues (Khoreva et al. 2017a) and graph-based optimization algorithms such as GrabCut (Rother et al. 2004) or mean-field inference (Krähenbühl and Koltun 2011) on a densely-connected conditional random field (Papandreou et al. 2015; Rajchl et al. 2016; Song et al. 2019). Previous works have used the refined masks for training the segmentation model (Li et al. 2018). Some of these approaches additionally use EM (Dai et al. 2015; Khoreva et al. 2017a; Papandreou et al. 2015; Zhao et al. 2018) for iterative refinement of the ground truth and model parameters. However, training models with EM is time-consuming and is prone to the propagation of errors over iterations; In contrast, our proposed approach does not require iterative refinement and outperforms these methods by a large margin (Sect. 4.4).

Multi-tasking: Hu et al. (2018) train an object detector and segmentation model simultaneously using the Mask-RCNN framework (He et al. 2017) assuming a closed-form relationship between parameters of the detection and segmentation branches. While Hu et al. (2018) can benefit from the advantages of multi-tasking, it requires (a) (limited) training data with per-pixel annotations, (b) a region of interest (RoI) proposal method, and (c) uses the RoI warping module which captures local context alone. Our approach does not require any per-pixel annotations and uses a fully convolutional network. We capture global image context via long-range interactions by training pixel embeddings.

Noise Robust Loss: We consider bounding boxes as noisy labels for foreground objects. Noise robust loss functions are essential for training accurate deep neural networks in the presence of noisy (incorrect) labels. Label Smoothing (Pereyra et al. 2017) with the help of smooth labels instead of one-hot labels can help avoid overfitting to noisy labels. Some loss functions such as Mean Absolute Error (MAE) have been shown to be robust to noisy labels while the popular cross-entropy loss function has not been found to be robust to noise (Ghosh et al. 2017). However, training with MAE is not easy as gradient saturation

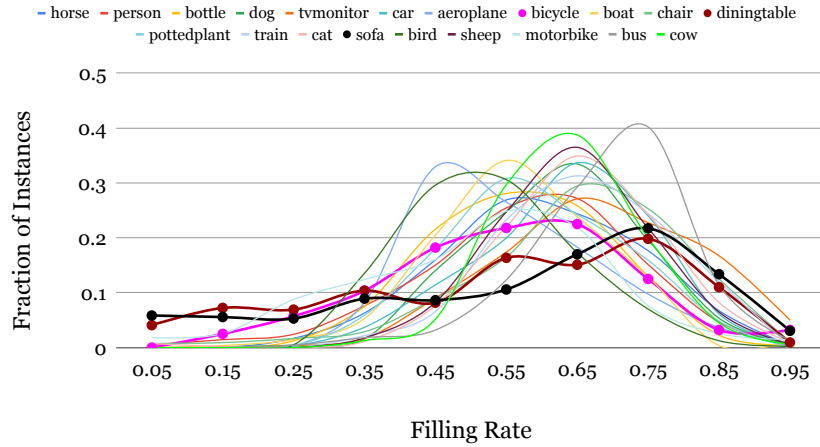


Figure 4.2.1: Distribution of filling rate in PASCAL VOC 2012 training dataset. The per-class filling rates vary widely, especially for the categories such as sofa, dining table and bicycle. Unlike Song et al. (2019) that proposes using 1-3 filling rates per class, we use a per-image class-specific filling rate.

can cause slow convergence (Zhang and Sabuncu 2018) and robust loss functions usually suffer from the underfitting problem (Ma et al. 2020).

Label Noise: Some fully-supervised approaches (Petit et al. 2018; Min et al. 2018) have tackled noise in the segmentation ground truth by consolidating predictions from two or more independent classifiers and discarding pixels with ambiguous predictions from the training data. Among weakly supervised methods, Song et al. (2019) propose the idea of using the filling rate as a cue to supervise training with bounding box ground truth. The filling rate of a class (Song et al. 2019) is defined as the average proportion of pixels in a bounding box that instances of the class occupy, and is estimated by applying dense-CRF (Krähenbühl and Koltun 2011) for foreground extraction within bounding boxes. For example, suppose in the PASCAL VOC 2012 dataset, instances of ‘sheep’ occupy roughly 60% of pixels in each bounding box. During training, their approach ignores gradients from the 40% pixels with the lowest confidences in each bounding box containing ‘sheep’. In contrast, we advocate estimating a *per-image class-specific* spatial attention map, to allow for large intra-class variations in object appearance (see Fig. 4.2.1). Instead of ignoring loss on pixels with low confidence as in Song et al. (2019), which is a hard constraint, we use the filling rate to regularize the attention map as a soft constraint. Therefore, our attention map offers a continuous modulation of cross-entropy loss, rather than a binary decision. Attention maps have also been employed in Ahn et al. (2019) and Zhu et al. (2019) where they are trained in a fully-supervised manner using pseudo segmentation ground truth. In contrast, our attention-

modulated loss offers a principled way of handling label noise present in segmentation ground truth derived from bounding box annotations. We validate our approach with quantitative and qualitative comparisons with Song et al. (2019) in Sect. 4.4.

4.3 Proposed Approach

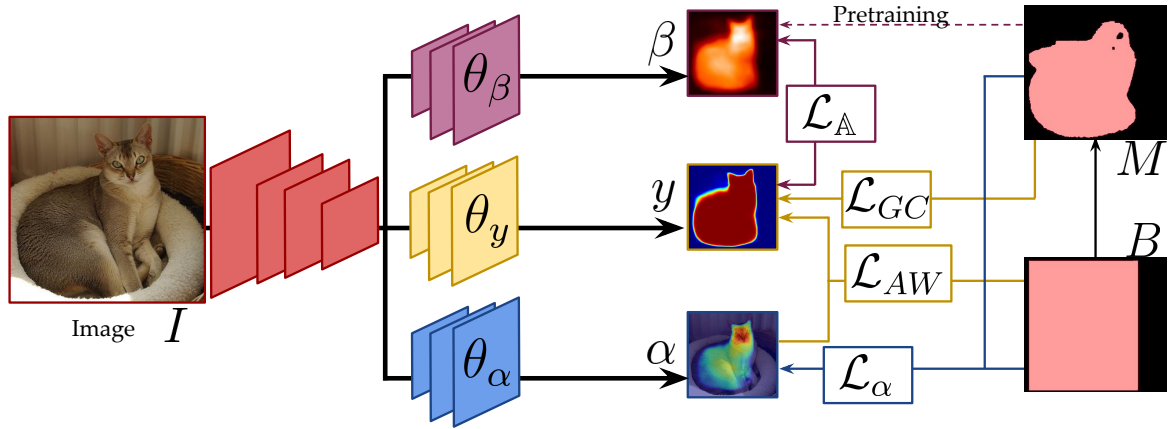


Figure 4.3.1: Overview of our weakly-supervised Box2Seg training pipeline. Our segmentation model is a three-branch CNN which outputs segmentation probabilities γ , per-class spatial attention maps α , and pixel embeddings β . B denotes segmentation masks defined using bounding box annotations. GrabCut is applied to B to obtain refined segmentation masks M . We use B and M to supervise the three outputs of the model. Pixel embeddings capture long-range pairwise relationships and the attention map refines the segmentation output γ by reducing the effect of label noise. At test time we only use the branch producing the segmentation probabilities γ , and discard the other branches that output α, β .

In this section, we describe our Box2Seg algorithm in detail. We discuss our pipeline and loss functions that allow us to learn a per-image, class-specific attention map as well as pixel embeddings.

4.3.1 Feed Forward Network Architecture

Let $I \in \mathbb{R}^{n \times 3}$ denote a 3-channel input color image with n pixels. Our segmentation model $\mathcal{S}$ is a fully-Convolutional Neural Network (CNN) which takes I as input and produces three outputs: (i) the segmentation output probabilities $\gamma \in \mathbb{P}^{m \times (L+1)}$, $\mathbb{P} \in [0, 1]$, (ii) the attention map $\alpha \in \mathbb{P}^{m \times L}$, and (iii) the pixel embeddings $\beta \in \mathbb{R}^{m \times d}$. Here d denotes the size of the pixel embeddings and m denotes the spatial resolution of the outputs. Each pixel in the output image can assume one of the $L + 1$ labels (L object categories and the background

class). The ground truth bounding boxes are denoted by $B_{box} \in \mathbb{R}^{K \times 5}$, where K is the number of bounding boxes comprising of 4 coordinates and a class label. To simplify the notation, we denote by $B \in \mathbb{R}^{m \times (L+1)}$ the box-segmentation tensor obtained by setting all pixels inside a bounding box to 1 for the corresponding class label (channel). In the case two boxes overlap, we assign 1 to the class corresponding to the smaller bounding box (assuming that the smaller box is in the front). Note that this assumption may not always be true and can result in incorrect label assignments for B (See eg. Fig. 4.4.3).

Similar to prior works (Papandreou et al. 2015; Rajchl et al. 2016; Song et al. 2019), we generate pseudo-ground-truth segmentation maps $M \in \mathbb{R}^{m \times (L+1)}$ by applying classical graph-based unsupervised segmentation approach (Grabcut (Rother et al. 2004)) on each bounding box in our training dataset. The segmentation masks obtained by classical methods, albeit noisy and imprecise, provide a good prior for training deep learning models.

Fig. 4.3.1 gives an overview of our approach. Training involves passing the input image I through a common feature encoder that feeds into the three branches of network $\mathcal{S}$ to produce y , α , and β , as follows:

$$y = \text{Softmax}(\mathcal{S}(I, \theta_y)); \alpha = \sigma(\mathcal{S}(I, \theta_\alpha)); \beta = \mathcal{S}(I, \theta_\beta),$$

where $\theta_y, \theta_\alpha, \theta_\beta$ denote the parameters of the model $\mathcal{S}$ for the respective branches. Softmax denotes the softmax over all the classes and σ denotes the sigmoid activation function. Note that the segmentation output probabilities, y , sum up to 1 for each pixel across classes due to softmax. However, the activation maps, α , use sigmoid output, making them independent for each class. Note that our model does not use additional parameters compared to any baselines we compare against in Sec. 4.4 as we discard the branches producing α, β at test time.

4.3.2 Box And GrabCut Based Losses

We use the box-segmentation tensor B to train a simple baseline by minimizing the following cross-entropy loss:

$$\mathcal{L}_{box} = -\frac{1}{m} \sum_{c=0}^L \sum_{i=1}^m B(i, c) \log(y(i, c)), \quad (4.1)$$

where $B(i, c)$ is 1 if i^{th} pixel is covered by the bounding box of c^{th} class label, and 0 otherwise. Similar to previous works, we use the GrabCut outputs M obtained from the bounding boxes to define another baseline by minimizing the following cross-entropy loss:

$$\mathcal{L}_{GC} = -\frac{1}{m} \sum_{c=0}^L \sum_{i=1}^m M(i, c) \log(y(i, c)). \quad (4.2)$$

Since the GrabCut algorithm provides reasonable segmentation outputs (Sect. 4.4), we use $\mathcal{L}_{GC}$ in addition to our loss functions described in the following sections.

4.3.3 Attention Weighted Segmentation Loss

Our novel attention-modulated cross-entropy loss considers bounding box annotations as *noisy labels* for the foreground object. Note that since the bounding box is a super-set of the actual object segmentation mask, the label noise is one-sided: foreground labels are incorrectly assigned to background pixels, but no true foreground labels are missing. Additionally, pixels outside all the bounding boxes can be considered *definite background* and do not have any label noise. Since supervised segmentation training typically considers all labels as correct, in the presence of label noise, erroneous gradients can be backpropagated during training. At pixels close to the bounding box and object boundaries, the network gets conflicting information about the foreground/background labels at similar pixels. This is the reason for the worse performance of baseline trained with box annotations only (Table 4.4.3).

Attention on Foreground Objects: We propose to modulate the per-pixel cross-entropy loss using the predicted attention map from the network by minimizing

$$\mathcal{L}_{fg} = \frac{-1}{\sum_i^m B(i, c)} \sum_{c=1}^L \sum_{i=1}^m \alpha(i, c) B(i, c) \log(y(i, c)). \quad (4.3)$$

Note that the attention map has the same spatial resolution as the segmentation output and is class-specific. The attention-weighted loss is only defined for the L foreground classes. In addition, the loss is normalized with the size of the bounding box, to give similar weighting to each class.

Background Loss: Since the pixels outside all the bounding boxes can be considered as definite background, the background loss is defined as

$$\mathcal{L}_{bg} = -\frac{\sum_i^m B(i, 0) \log(y(i, 0))}{\sum_i^m B(i, 0)}, \quad (4.4)$$

where 0 denotes the background class. We define the **Attention Weighted Loss (AWL)** as

$$\mathcal{L}_{AW} = \mathcal{L}_{fg} + \mathcal{L}_{bg}. \quad (4.5)$$

Attention Map Regularization: Without any regularization on the attention maps, the network can minimize $\mathcal{L}_{fg}$ by predicting all $\alpha(i, c) = 0$. To prevent this, we compared two approaches to regularize the attention map. The first approach regularizes the attention mask

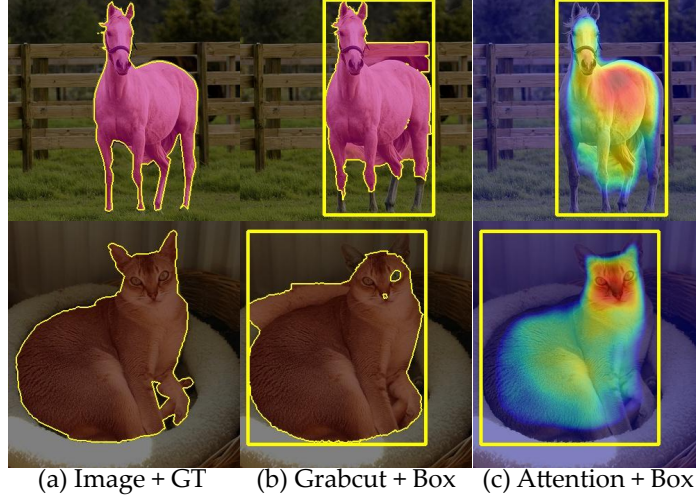


Figure 4.3.2: Visualizing attention maps produced by our network. (a) Input images with overlaid ground truth masks. (b) Grabcut output with ground-truth bounding box. (c) Predicted attention map. Note that while Grabcut output is erroneous and often includes background pixels, our attention maps are concentrated on the objects of interest.

using ground truth bounding boxes (similar to Song et al. (2019)), by minimizing the $\mathcal{L}_2$ loss between the attention maps and the bounding boxes.

$$\mathcal{L}_\alpha^{bbox} = \sum_{i=1}^m \sum_{c=1}^L \| B(i, c) - \alpha(i, c) \|^2. \quad (4.6)$$

The second approach regularizes the attention mask using fill-ratios obtained from GrabCut outputs M as follows. Let η_c denote the per class, per image, filling rate defined as the proportion of pixels in the pseudo-ground-truth M compared to its corresponding bounding box.

$$\eta_c = \frac{\sum_i^m M(i, c)}{\sum_i^m B(i, c)}. \quad (4.7)$$

Similarly, the predicted fill rate of the attention map is computed as

$$\eta'_c = \frac{\sum_i^m B(i, c) \alpha(i, c)}{\sum_i^m B(i, c)}. \quad (4.8)$$

We use a margin loss to ensure that the predicted fill rate is at least a factor γ of the fill rate obtained using M . $\mathcal{L}_\alpha^{fr} = \max(0, \gamma \eta_c - \eta'_c)$, where $\gamma \in [0, 1]$ is a hyper-parameter which is set to 0.7 in our experiments. Thus, the predicted fill rate (Eqn. 4.8) is allowed to

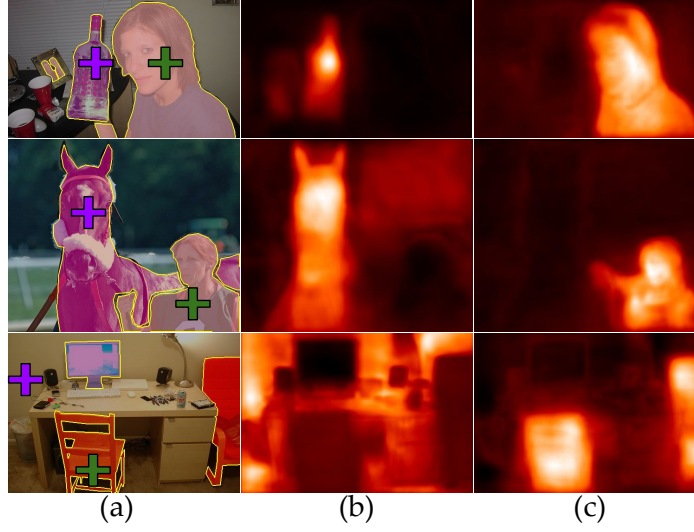


Figure 4.3.3: Visualization of affinities produced by our network. (a) Input image with overlaid ground truth masks and two pixels marked with **purple** and **green** + signs. (b,c) Affinity heatmaps (Eqn. 4.9) w.r.t. the **purple** and **green** pixels, resp.

vary between $\gamma\eta_c$ and 1. This enforces a soft constraint that the attention map should allow propagation of loss from at least 70% of η_c pixels inside the bounding box. Using L_{α}^{bbox} for regularization forces the attention mask to take the shape of the bounding box. Thus, it is prone to include background pixels in the attention map. Using L_{α}^{fr} provides a softer constraint and gives better results in our experiments. Figure 4.3.2 shows some qualitative examples where the attention maps are able to focus on foreground pixels despite errors in the underlying GrabCut segmentations, allowing Box2Seg to be robust to label noise.

4.3.4 Discriminative Feature Learning

We now describe the training of our pixel embeddings β which capture long-range pairwise relationships between different pixels. Pixel embeddings can be denoted as $\beta = \{\beta_i\}$, where β_i is the d -dimensional feature for the i^{th} pixel. The affinity between embeddings at pixel i and j is given by its normalized dot product

$$\mathbb{A}(i, j) = \beta_i \cdot \beta_j = \frac{\beta_j^T \beta_i}{\|\beta_j\| \|\beta_i\|}. \quad (4.9)$$

Intuitively, we want to achieve high affinity between feature vectors of two pixels that belong to the same class, while ensuring low affinity between features of two different classes. Similarly, a background pixel should have a low affinity with another pixel that belongs

to one of the L foreground classes. To achieve this, we define a novel loss function on pairs of pixels (i, j) , such that it encourages the pixel affinities to align with the predicted segmentation probabilities, as follows,

$$\mathcal{L}_{\mathbb{A}} = \sum_{i,j} \left(\mathbb{A}(i, j) - y_j^T y_i \right)^2. \quad (4.10)$$

However, training affinity matrices requires a large amount of memory. To avoid creating large affinity matrices of size $m \times m$, we randomly sample a small fraction of pixel embeddings equally from each class to compute this loss. Figure 4.3.3 shows the affinity maps computed from our class discriminative embeddings.

4.3.5 Training Box2Seg

Our approach optimizes the following loss function:

$$\mathcal{L} = \mathcal{L}_{GC} + \lambda_{AW} \mathcal{L}_{AW} + \lambda_{\alpha} \mathcal{L}_{\alpha} + \lambda_{\mathbb{A}} \mathcal{L}_{\mathbb{A}}, \quad (4.11)$$

where $\mathcal{L}_{\alpha}$ equals either $\mathcal{L}_{\alpha}^{bbox}$ or $\mathcal{L}_{\alpha}^{fr}$. λ_{AW} , λ_{α} and $\lambda_{\mathbb{A}}$ are weights applied to the individual losses. As long as λ_{AW} is greater than λ_{α} and $\lambda_{\mathbb{A}}$, it is quite robust to their variation.

4.4 Experiments and Results

We evaluate the performance of our approach on the PASCAL VOC 2012 (Everingham et al. 2010) dataset. Our ablation studies provide insights into our design choices. Finally, we demonstrate that our method provides a better initialization than Imagenet pretraining for the task of semi-supervised segmentation.

4.4.1 Implementation Details

Our segmentation network architecture is similar to UPerNet (Xiao et al. 2018) where the encoder backbone is ResNet-101 (He et al. 2016), and decoders consist of 2 convolutional layers. We employ the ResNet-101 backbone to ensure fair comparison with the three most recent SOTA works, SDI (Khoreva et al. 2017a), Li et al. (2018), and BCM (Song et al. 2019) as well as 4 other recent methods (Lee et al. 2019; Tang et al. 2018a; Tang et al. 2018b; Wang et al. 2019) in Table 4.4.1. We have three decoders, one each for the y , α , and β branches. The final results are spatially down-sampled by a factor of 4, *i.e.*, $m = n/16$. We start with ImageNet pretrained (Deng et al. 2009) weights to initialize our encoder.

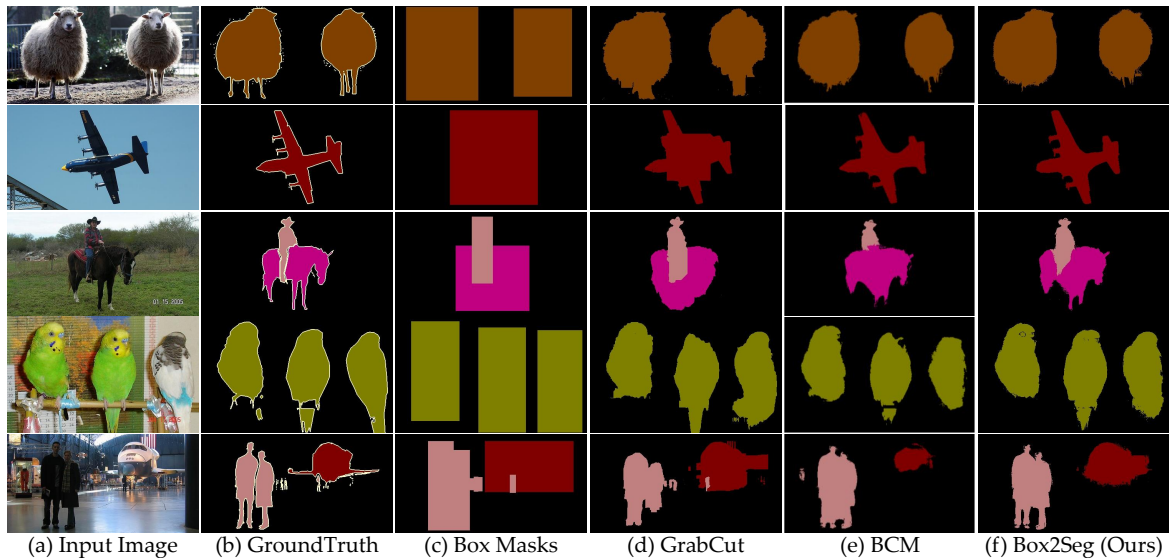


Figure 4.4.1: Comparison of our segmentation results with those of BCM (Song et al. 2019).

We train our network in two stages. First, we pre-train the feature representations using the affinity loss by randomly sampling 10% of pixel pairs using the Grabcut outputs M . This is done by minimizing Eqn. 4.10 using M_i in place of y_i . Note that the pretraining phase is meant to only serve the task of weight initialization for our discriminative feature learning. Our final feature representations are robust to noise in the Grabcut outputs since they are trained eventually to agree with our predictions using Eqn. 4.10. After the pretraining phase, we enable the decoder branch to also output y and α and train the entire network end-to-end to optimize the loss function in Eqn. 4.11. After doing a grid search of hyper-parameters on a held-out validation set, λ_{AW} , λ_α and $\lambda_{\mathbb{A}}$ are set to 10, 1, and 1 respectively. We use Stochastic Gradient Descent to train our models for 40 epochs with an initial learning rate $= 1e - 4$, momentum $= 0.9$, weight decay $= 5e^{-4}$ and a polynomially decaying learning rate as in Chen et al. (2014). Our implementation uses PyTorch (Paszke et al. 2017) and is trained on Nvidia’s TitanX GPUs. Note that at test time we discard the decoders yielding α, β , therefore our method uses no additional parameters compared to any of our baselines in the following sections.

4.4.2 Quantitative and Qualitative Evaluation

PASCAL VOC 2012 is one of the gold standard benchmarks for semantic segmentation. Following Chen et al. (2017b), Dai et al. (2015), and Long et al. (2015), we use the augmented

Method	Annotations	Backbone	mIoU
GrabCut-NoTrain-GT	box	-	71.6
GrabCut-NoTrain-Det	box	-	68.5
SSNet (Zeng et al. 2019)	image-level	DenseNet-169	63.3
F2FA (Lee et al. 2019)	image-level	ResNet-101	66.5
ScribbleSup (C) (Lin et al. 2016)	scribble	VGG-16	63.1
NormalCut (Tang et al. 2018a)	scribble	ResNet-101	72.8
BPG (Wang et al. 2019)	scribble	ResNet-101	73.2
KernelCut (Tang et al. 2018b)	scribble	ResNet-101	73.0
WSSL (C) (Papandreou et al. 2015)	box	VGG-16	60.6
BoxSup (C) (Dai et al. 2015)	box	VGG-16	62.0
SDI (C) (Khoreva et al. 2017a)	box	ResNet-101	69.4
BCM (C) (Song et al. 2019)	box	ResNet-101	70.2
Li et al. (2018) (C)	box	ResNet-101	74.3
Box2Seg	box	ResNet-101	74.9
Box2Seg (C)	box	ResNet-101	76.4

Table 4.4.1: Comparison of Box2Seg to previous weakly supervised semantic segmentation methods on PASCAL VOC validation set. C=dense-CRF post-processing.

annotation set (Hariharan et al. 2011) consisting of 10582 training and 1449 validation images. Performance is measured using the mIoU metric on the validation set.

We compare the accuracy of our algorithm to the SOTA weakly supervised segmentation methods on the VOC validation set in Table 4.4.1. Our approach shows a large improvement over previous methods (Papandreou et al. 2015; Dai et al. 2015; Khoreva et al. 2017a; Song et al. 2019; Li et al. 2018), some of which have also used VGG-16 backbones for feature representation. Li et al. (2018) report a mIoU of 74.3%, using bounding boxes as

Method	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mIoU
BCM VGG (CRF) (Song et al. 2019)	89.8	68.3	27.1	73.7	56.4	72.6	84.2	75.6	79.9	35.2	78.3	53.2	77.6	66.4	68.1	73.1	56.8	80.1	45.1	74.7	54.6	66.8
BCM ResNet (CRF) (Song et al. 2019)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	70.2
Li et al. (2018) (CRF)	93.3	85.0	35.9	88.6	70.3	77.9	91.9	83.6	90.5	39.2	84.5	59.4	86.5	82.4	81.5	84.3	57.0	85.9	55.8	85.8	70.4	75.7
Box2Seg	92.5	66.5	31.7	78.9	65.5	83.4	90.4	86.7	86.0	55.1	81.8	59.9	80.5	74.1	76.0	75.7	65.3	85.1	72.5	87.8	77.7	74.9
Box2Seg (CRF)	93.3	72.4	33.0	84.2	64.9	83.5	90.9	86.7	88.7	57.2	83.6	62.5	82.6	76.8	77.0	77.8	63.3	87.2	75.1	88.3	74.1	76.4

Table 4.4.2: Per-class results on PASCAL VOC 2012 Validation set. We compare our *Box2Seg* (with and without dense CRF) with those of the previous state-of-the-art methods. Please note that Li et al. (CRF) results require COCO annotations. Per-class results from Song et al. (2019) for their ResNet-101 model are not available.

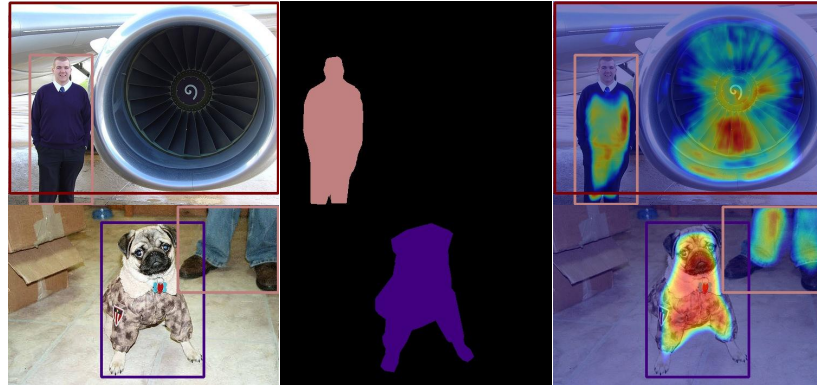


Figure 4.4.2: (Left) Input images with overlaid bounding box labels. (Middle) Missing pixel-level annotations on foreground objects such as *Aeroplane* and *Person*. Note that the bounding box annotations are correct. (Right) Predicted attention map with boxes. Please refer to Sect. 4.4.4 for a discussion on how our AWL can help in cases with conflicting ground truth during fully-supervised training.

supervision and dense CRF post-processing. Our method yields a mIoU of 74.9% without dense CRF, and 76.4% mIoU after dense CRF post processing resulting in an improvement of 2.1%. Box2Seg also outperforms recent weakly supervised methods that use image-level labels (Lee et al. 2019; Zeng et al. 2019) or scribbles (Lin et al. 2016; Tang et al. 2018a; Tang et al. 2018b; Wang et al. 2019). Per-category results comparing Box2Seg against the SOTA methods are reported in Table 4.4.2.

A qualitative comparison of our results with BCM (Song et al. 2019) is shown in Fig. 4.4.1. Our method can produce higher-quality object segmentations compared to BCM. It’s also robust to false segmentations in some cases, *e.g.*, false detection closer to the left edge of the BCM result on the last row is suppressed by our method. Please refer to Fig. 4.1.1 for additional examples of segmentation using our approach. We also show some failure cases of our approach in Fig. 4.4.3.

Accuracy of GrabCut (no training): We also evaluated the accuracy of the GrabCut algorithm itself, without any training, against the segmentation ground-truth. Interestingly, GrabCut output on *ground truth* bounding boxes (*GrabCut-NoTrain-GT*) results in a strong weakly-supervised baseline with 71.6% mIoU. However, since the ground truth bounding boxes are not available at inference, a more practical baseline is to obtain bounding boxes on the validation set using an object detector (we used SNIPER (Singh et al. 2018)) and then run GrabCut on those. This baseline (*GrabCut-NoTrain-Det*) obtains 68.5% mIoU.

Method	$\mathcal{L}_{box}$	$\mathcal{L}_{GC}$	$\mathcal{L}_{AW}$	$\mathcal{L}_{\alpha}$	$\mathcal{L}_{\mathbb{A}}$	mIoU
Box	✓					59.3
GrabCut		✓				72.7
Affinity		✓			✓	73.9
AW-box		✓	✓	$\mathcal{L}_{\alpha}^{bbox}$		74.1
AW-fr		✓	✓	$\mathcal{L}_{\alpha}^{fr}$		74.6
Box2Seg		✓	✓	$\mathcal{L}_{\alpha}^{fr}$	✓	74.9

Table 4.4.3: Ablation study showing the effect of our loss functions in improving the performance over baseline methods.

4.4.3 Ablation studies

Table 4.4.3 shows the efficacy of our loss functions in improving the performance of our approach. The trivial *Box* baseline obtained by training the segmentation network with bounding box supervision ($\mathcal{L}_{box}$ loss only) results in a low mIoU of 59.3%, as expected. Training the segmentation network with *GrabCut* masks ($\mathcal{L}_{GC}$ loss only) without affinity or attention losses resulted in 72.7% mIoU. Introducing our feature embeddings to the pipeline (Affinity) improves the mIoU to 73.9%. Our novel AWL terms ($\lambda_{AW}\mathcal{L}_{AW} + \lambda_{\alpha}\mathcal{L}_{\alpha}$) significantly improve the mIoU to 74.6% with filling-rate regularization (*AW-fr*) (more about AWL in Sect. 4.4.4). Finally, combining all losses (Eqn 4.11), *Box2Seg* obtains 74.9% mIoU.

4.4.4 Attention Weighted Loss in the Fully-Supervised Setting

We demonstrate that AWL boosts segmentation accuracy in the fully-supervised case if we have disagreements between bounding-box and per-pixel annotations.

During our analysis, we found that roughly 10% of training images in PASCAL VOC dataset (Everingham et al. 2010; Hariharan et al. 2011) have disagreements between the bounding box and pixel-level annotations. Fig. 4.4.2 shows a few examples, where pixel-level annotations for object categories are missing, but their corresponding bounding box labels are correctly provided. Due to incorrect pixel-level annotations, fully supervised training would back-propagate erroneous gradients on these pixels. Since our novel AWL ($\mathcal{L}_{AW}$ in Sect. 4.3.3) is effective in dealing with label noise for weakly supervised networks, we analyze if it can further improve the performance of a fully supervised network also. In cases where conflicting sources of ground truth exist (as in Fig. 4.4.2), AWL can allow correct

Method	CE Loss	+ $\mathcal{L}_{AW}$	Δ
Supervised baseline	73.6	75.1	+1.5

Table 4.4.4: Improvements in segmentation accuracy on the PASCAL VOC 2012 validation set in the fully supervised setting using AWL.

Method	mIoU
BCM (Song et al. 2019) † (S=0%) (C)	70.2
BCM (Song et al. 2019) † (S=13.8%) (C)	71.6
DeepLab † (S=100%) (C)	74.5
Box2Seg (S=0%)	74.9
Box2Seg (S=5%)	78.7
Box2Seg (S=10%)	83.1
Box2Seg (S=100%)	86.4

Table 4.4.5: Semi-supervised segmentation using Box2Seg model as initialization. (S= $\tau\%$) implies $\tau\%$ images with pixel annotations are used for semi-supervised fine-tuning. C=dense-CRF post-processing. †: ResNet-101 backbone (refer Song et al. (2019)).

gradients to propagate back due to correct bounding box annotations (see the predicted attention maps in Fig. 4.4.2), thereby reducing the effect of incorrect pixel-level annotations.

Table 4.4.4 shows the improvement obtained by adding AWL to the fully *Supervised baseline*. Adding the AWL (+ $\mathcal{L}_{AW}$) to the fully-*Supervised baseline* improved the segmentation mIoU by 1.5%. Thus, AWL is effective at improving segmentation accuracy, both in weakly and fully-supervised settings.

4.4.5 Semi-supervised Semantic Segmentation

A weakly-supervised trained method can naturally serve as a starting point for semi-supervised segmentation. To study this, we fine-tuned our Box2Seg model using different amounts of pixel-level annotations. We observe significant improvements in accuracy, even with a small amount of supervision as shown in Table 4.4.5. For comparison, we show the result of the semi-supervised fine-tuned BCM (Song et al. 2019) model trained with 1464 images (13.8% of the data) and another fully supervised baseline (*DeepLab*) using ImageNet as the initialization. Our weakly supervised baseline results improved from 74.9% mIoU to 83.1% with just 10% of supervised data. Therefore, our Box2Seg model can serve as a good starting point and provides better results compared to ImageNet-based initialization.

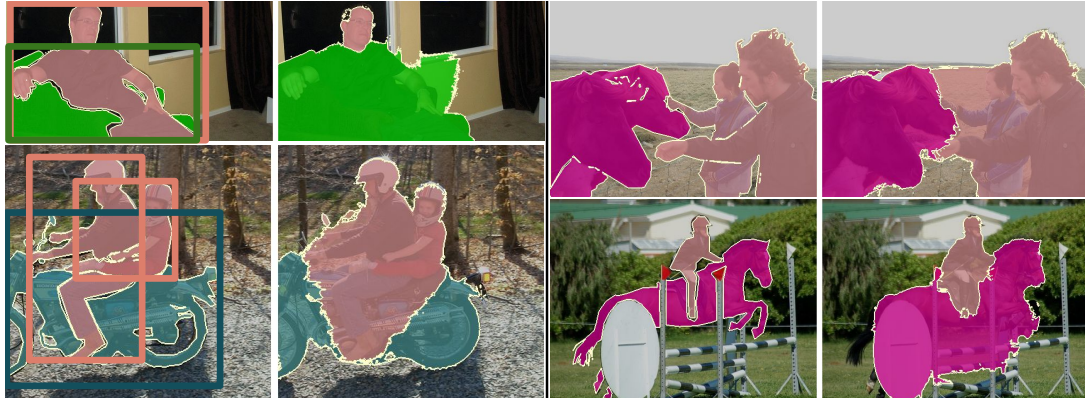


Figure 4.4.3: Failure Cases. (Left) Incorrect label assignment on box annotations B can happen when the smaller bounding box is physically behind the larger bounding box. (Right) Predicted segmentation bleeds into the background.

4.5 Conclusions

In this work, we proposed a pipeline for training a weakly supervised semantic segmentation method from bounding box annotations. We showed that bounding box annotations can be treated as noisy labels for foreground objects and proposed a novel attention-weighted loss to reduce the effect of erroneous gradients due to incorrect labels. We also proposed pixel embeddings to capture global context via long-range pairwise interactions. We showed qualitative improvements over the previous SOTA on the PASCAL VOC semantic segmentation benchmark and pushed the mIoU metric forward by 2.1%. Interestingly, fully supervised methods can also benefit from attention-weighted loss in the presence of exhaustive bounding box annotations but missing pixel-level annotations. Future work would involve using an edge detector as a cue to learn class boundaries, and also extending our method to benefit from image-level supervision.

5

Calibrating Deep Neural Networks using Focal Loss

Abstract

Miscalibration – a mismatch between a model’s confidence and its correctness – of Deep Neural Networks (DNNs) makes their predictions hard to rely on. Ideally, we want networks to be accurate, calibrated, and confident. We show that, as opposed to the standard cross-entropy loss, focal loss (Lin et al. 2017) allows us to learn models that are already very well-calibrated. When combined with temperature scaling, whilst preserving accuracy, it yields state-of-the-art calibrated models. We provide a thorough analysis of the factors causing miscalibration and use the insights we glean from this to justify the empirically excellent performance of focal loss. To facilitate the use of focal loss in practice, we also provide a principled approach to automatically select the hyperparameter involved in the loss function. We perform extensive experiments on a variety of computer vision and NLP datasets, and with a wide variety of network architectures, and show that our approach achieves state-of-the-art calibration without compromising on accuracy in almost all cases. Code is available at https://github.com/torrvision/focal_calibration.

5.1 Introduction

Deep neural networks have dominated computer vision and machine learning in recent years, and this has led to their widespread deployment in real-world systems (Cao et al. 2018; Chen et al. 2018a; Kamilaris and Prenafeta-Boldú 2018; Ker et al. 2018; Wang et al. 2018a). However, many current multi-class classification networks in particular are poorly calibrated, in the sense that the probability values that they associate with the class labels they predict overestimate the likelihood of those class labels being correct in the real world. This is a major problem, since if networks are routinely overconfident, then downstream components cannot trust their predictions. The underlying cause is hypothesized to be that these networks’ high capacity leaves them vulnerable to overfitting on the negative log-likelihood (NLL) loss they conventionally use during training (Guo et al. 2017).

Given the importance of this problem, numerous suggestions for how to address it have been proposed. Much work has been inspired by approaches that were not originally formulated in a deep learning context, such as Platt scaling (Platt 1999), histogram binning (Zadrozny and Elkan 2001), isotonic regression (Zadrozny and Elkan 2002), and Bayesian binning and averaging (Naeini et al. 2015; Naeini and Cooper 2016). As deep learning has become more dominant, however, various works have begun to directly target the calibration of deep networks. For example, Guo et al. (2017) have popularised a modern variant of Platt scaling known as *temperature scaling*, which works by dividing a network’s logits by a scalar $T > 0$ (learned on a validation subset) before performing softmax. Temperature scaling has the desirable property that it can improve the calibration of a network without in any way affecting its accuracy. However, whilst its simplicity and effectiveness have made it a popular network calibration method, it does have downsides. For example, whilst it scales the logits to reduce the network’s confidence in incorrect predictions, this also slightly reduces the network’s confidence in predictions that were correct (Kumar et al. 2018). Moreover, it is known that temperature scaling does not calibrate a model under data distribution shift (Ovadia et al. 2019).

By contrast, Kumar et al. (2018) initially eschew temperature scaling in favour of minimizing a differentiable proxy for calibration error at training time, called Maximum Mean Calibration Error (MMCE), although they do later also use temperature scaling as a post-processing step to obtain better results than cross-entropy followed by temperature scaling (Guo et al. 2017). Separately, Müller et al. (2019) propose training models on cross-entropy loss with label smoothing instead of one-hot labels and show that label smoothing

has a very favourable effect on model calibration.

In this chapter, we propose a technique for improving network calibration that works by replacing the cross-entropy loss conventionally used when training classification networks with the focal loss proposed by Lin et al. (2017). We observe that unlike cross-entropy, which minimizes the KL divergence between the predicted (softmax) distribution and the target distribution (one-hot encoding in classification tasks) over classes, focal loss minimizes a regularised KL divergence between these two distributions, which ensures minimization of the KL divergence whilst *increasing the entropy* of the predicted distribution, thereby preventing the model from becoming overconfident. Please note that Pereyra et al. (2017) found that explicitly maximizing entropy while minimizing KL divergence regularizes the network by avoiding overfitting and it is similar to label smoothing (Szegedy et al. 2016). In label smoothing uniform noise is added to the labels which can possibly also make the network noise-robust through loss correction (Ma et al. 2020). This further opens the doors for noise-regularized loss functions which can be explored from the perspective of network calibration. Since the focal loss, as shown in §5.4, is dependent on a hyperparameter, γ , that needs to be cross-validated, we also provide a method for choosing γ automatically for each sample, and show that it outperforms all the baseline models.

The intuition behind using focal loss is to direct the network’s attention during training towards samples for which it is currently predicting a low probability for the correct class, since trying to reduce the NLL on samples for which it is already predicting a high probability for the correct class is liable to lead to NLL overfitting, and thereby miscalibration (Guo et al. 2017). More formally, we show in §5.4 that focal loss can be seen as *implicitly* regularising the weights of the network during training by causing the gradient norms for confident samples to be lower than they would have been with cross-entropy, which we would expect to reduce overfitting and improve the network’s calibration.

Overall, we make the following contributions:

1. In §5.3, we study the link that Guo et al. (2017) observed between miscalibration and NLL overfitting in detail, and show that the overfitting is associated with the predicted distributions for misclassified test samples becoming peakier as the optimizer tries to increase the magnitude of the network’s weights to reduce the training NLL.
2. In §5.4, we propose the use of focal loss for training better-calibrated networks and provide both theoretical and empirical justifications for this approach. In addition, we provide a principled method for automatically choosing γ for each sample during training.

3. In §5.5, we show, via experiments on a variety of classification datasets and network architectures, that DNNs trained with focal loss are more calibrated than those trained with cross-entropy loss (both with and without label smoothing), MMCE or Brier loss (Brier 1950). Finally, we also make the interesting observation that whilst temperature scaling may not work for detecting out-of-distribution (OoD) samples, our approach can. We show that our approach is better at detecting out-of-distribution samples, taking CIFAR-10 as the in-distribution dataset, and SVHN and CIFAR-10-C as out-of-distribution datasets.

5.2 Problem Formulation

Let $D = \langle (\mathbf{x}_i, y_i) \rangle_{i=1}^N$ denote a dataset consisting of N samples from a joint distribution $\mathcal{D}(\mathcal{X}, \mathcal{Y})$, where for each sample i , $\mathbf{x}_i \in \mathcal{X}$ is the input and $y_i \in \mathcal{Y} = \{1, 2, \dots, K\}$ is the ground-truth class label. Let $\hat{p}_{i,y} = f_\theta(y|\mathbf{x}_i)$ be the probability that a neural network f with model parameters θ predicts for a class y on a given input $\mathbf{x}_i$. The class that f predicts for $\mathbf{x}_i$ is computed as $\hat{y}_i = \operatorname{argmax}_{y \in \mathcal{Y}} \hat{p}_{i,y}$, and the predicted confidence as $\hat{p}_i = \max_{y \in \mathcal{Y}} \hat{p}_{i,y}$. The network is said to be *perfectly calibrated* when, for each sample $(\mathbf{x}, y) \in D$, the confidence $\hat{p}$ is equal to the model accuracy $\mathbb{P}(\hat{y} = y|\hat{p})$, *i.e.*, the probability that the predicted class is correct. For instance, of all the samples to which a perfectly calibrated neural network assigns confidence of 0.8, 80% should be correctly predicted.

A popular metric used to measure model calibration is the *expected calibration error* (ECE) (Naeini et al. 2015), defined as the expected absolute difference between the model’s confidence and its accuracy, *i.e.*, $\mathbb{E}_{\hat{p}}[|\mathbb{P}(\hat{y} = y|\hat{p}) - \hat{p}|]$. Since we only have finite samples, the ECE cannot in practice be computed using this definition. Instead, we divide the interval $[0, 1]$ into M equispaced bins, where the i^{th} bin is the interval $\left(\frac{i-1}{M}, \frac{i}{M}\right]$. Let B_i denote the set of samples with confidences belonging to the i^{th} bin. The accuracy A_i of this bin is computed as $A_i = \frac{1}{|B_i|} \sum_{j \in B_i} \mathbb{1}(\hat{y}_j = y_j)$, where $\mathbb{1}$ is the indicator function, and $\hat{y}_j$ and y_j are the predicted and ground-truth labels for the j^{th} sample. Similarly, the confidence C_i of the i^{th} bin is computed as $C_i = \frac{1}{|B_i|} \sum_{j \in B_i} \hat{p}_j$, *i.e.*, C_i is the average confidence of all samples in the bin. The ECE can be approximated as a weighted average of the absolute difference between the accuracy and confidence of each bin: $\text{ECE} = \sum_{i=1}^M \frac{|B_i|}{N} |A_i - C_i|$.

A similar metric, the *maximum calibration error* (MCE) (Naeini et al. 2015), is defined as the maximum absolute difference between the accuracy and confidence of each bin: $\text{MCE} = \max_{i \in \{1, \dots, M\}} |A_i - C_i|$.

AdaECE: One disadvantage of ECE is the uniform bin width. For a trained model, most of the samples lie within the highest confidence bins, and hence these bins dominate the value of the ECE. We thus also consider another metric, AdaECE (Adaptive ECE), for which bin sizes are calculated so as to evenly distribute samples between bins (similar to the adaptive binning procedure in Nguyen and O’Connor (2015)): $\text{AdaECE} = \sum_{i=1}^M \frac{|B_i|}{N} |A_i - C_i|$ s.t. $\forall i, j \cdot |B_i| = |B_j|$.

Classwise-ECE: The ECE metric only considers the probability of the predicted class, without considering the other scores in the softmax distribution. A stronger definition of calibration would require the probabilities of all the classes in the softmax distribution to be calibrated (Kull et al. 2019; Vaicenavicius et al. 2019; Widmann et al. 2019; Kumar et al. 2019). This can be achieved with a simple classwise extension of the ECE metric: $\text{ClasswiseECE} = \frac{1}{K} \sum_{i=1}^M \sum_{j=1}^K \frac{|B_{i,j}|}{N} |A_{i,j} - C_{i,j}|$, where K is the number of classes, $B_{i,j}$ denotes the set of samples from the j^{th} class in the i^{th} bin, $A_{i,j} = \frac{1}{|B_{i,j}|} \sum_{k \in B_{i,j}} \mathbb{1}(j = y_k)$ and $C_{i,j} = \frac{1}{|B_{i,j}|} \sum_{k \in B_{i,j}} \hat{p}_{kj}$.

A common way of visualising calibration is to use a *reliability plot* (Niculescu-Mizil and Caruana 2005), which plots the accuracies of the confidence bins as a bar chart (see Appendix Figure C.1.1). For a perfectly calibrated model, the accuracy for each bin matches the confidence, and hence all of the bars lie on the diagonal. By contrast, if most of the bars lie above the diagonal, the model is more accurate than it expects and is under-confident, and if most of the bars lie below the diagonal, then it is over-confident.

5.3 What Causes Miscalibration?

We now discuss why high-capacity neural networks, despite achieving low classification errors on well-known datasets, tend to be miscalibrated. A key empirical observation made by Guo et al. (2017) was that poor calibration of such networks appears to be linked to overfitting the negative log-likelihood (NLL) during training. In this section, we further inspect this observation to provide new insights.

For the analysis, we train a ResNet-50 network on CIFAR-10 with state-of-the-art performance settings (*Train CIFAR10 with PyTorch* 2019). We use Stochastic Gradient Descent (SGD) with a mini-batch of size 128, a momentum of 0.9, and a learning rate schedule of $\{0.1, 0.01, 0.001\}$ for the first 150, next 100, and last 100 epochs, respectively. We minimise cross-entropy loss (a.k.a. NLL) $\mathcal{L}_c$, which, in a standard classification context, is $-\log \hat{p}_{i,y_i}$, where $\hat{p}_{i,y_i}$ is the probability assigned by the network to the correct class

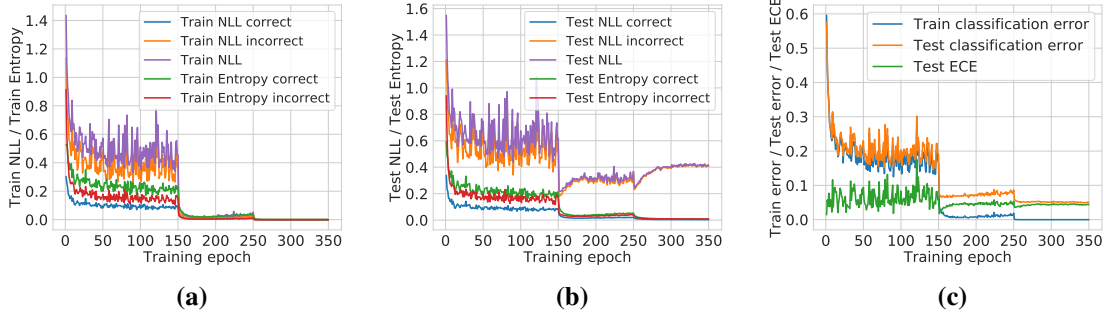


Figure 5.3.1: Metrics related to calibration plotted whilst training a ResNet-50 network on CIFAR-10.

y_i for the i^{th} sample. Note that the NLL is minimised when for each training sample i , $\hat{p}_{i,y_i} = 1$, whereas the classification error is minimised when $\hat{p}_{i,y_i} > \hat{p}_{i,y}$ for all $y \neq y_i$. This indicates that even when the classification error is 0, the NLL can be positive, and the optimisation algorithm can still try to reduce it to 0 by further increasing the value of $\hat{p}_{i,y_i}$ for each sample (see Appendix C.1).

To study how miscalibration occurs during training, we plot the average NLL for the train and test sets at each training epoch in Figures 5.3.1(a) and 5.3.1(b). We also plot the average NLL and the entropy of the softmax distribution produced by the network for the correctly and incorrectly classified samples. In Figure 5.3.1(c), we plot the classification errors on the train and test sets, along with the test set ECE.

Curse of misclassified samples: Figures 5.3.1(a) and 5.3.1(b) show that although the average train NLL (for both correctly and incorrectly classified training samples) broadly decreases throughout training, after the 150^{th} epoch (where the learning rate drops by a factor of 10), there is a marked rise in the average test NLL, indicating that the network starts to overfit on average NLL. This increase in average test NLL is caused only by the incorrectly classified samples, as the average NLL for the correctly classified samples continues to decrease even after the 150^{th} epoch. We also observe that after epoch 150, the test set ECE rises, indicating that the network is becoming miscalibrated. This corroborates the observation in Guo et al. (2017) that miscalibration and NLL overfitting are linked.

Peak at the wrong place: We further observe that the entropies of the softmax distributions for both the correctly and incorrectly classified *test* samples decrease throughout training (in other words, the distributions get peakier). This observation, coupled with the one we made above, indicates that *for the wrongly classified test samples, the network gradually becomes more and more confident about its incorrect predictions.*

Weight magnification: The increase in confidence of the network’s predictions can happen if the network increases the norm of its weights W to increase the magnitudes of the logits. In fact, the cross-entropy loss is minimised when for each training sample i , $\hat{p}_{i,y_i} = 1$, which is possible only when $\|W\| \rightarrow \infty$. Cross-entropy loss thus inherently induces this tendency of weight magnification in neural network optimization. The promising performance of weight decay (Guo et al. 2017) (regulating the norm of weights) on the calibration of neural networks can perhaps be explained using this. This increase in the network’s confidence during training is one of the key causes of miscalibration.

5.4 Improving Calibration using Focal Loss

As discussed in §5.3, overfitting on NLL, which is observed as the network grows more confident in all of its predictions irrespective of their correctness, is strongly related to poor calibration. One cause of this is that the cross-entropy objective minimizes the difference between the softmax distribution and the ground-truth one-hot encoding over an entire mini-batch, irrespective of how well a network classifies individual samples in the mini-batch. In this work, we study an alternative loss function, popularly known as *focal loss* (Lin et al. 2017), that tackles this by weighting loss components generated from individual samples in a mini-batch by how well the model classifies them. For classification tasks where the target distribution is a one-hot encoding, it is defined as $\mathcal{L}_f = -(1 - \hat{p}_{i,y_i})^\gamma \log \hat{p}_{i,y_i}$, where γ is a user-defined hyperparameter¹.

Why might focal loss improve calibration? We know that cross-entropy forms an upper bound on the KL-divergence between the target distribution q and the predicted distribution $\hat{p}$, i.e., $\mathcal{L}_c \geq \text{KL}(q||\hat{p})$, so minimising cross-entropy results in minimising $\text{KL}(q||\hat{p})$. Interestingly, a general form of focal loss can be shown to be an upper bound on the regularised KL-divergence, where the regulariser is the negative entropy of the predicted distribution $\hat{p}$, and the regularisation parameter is γ , the hyperparameter of focal loss (a proof of this can be found in Appendix C.2):

$$\mathcal{L}_f \geq \text{KL}(q||\hat{p}) - \gamma \mathbb{H}[\hat{p}]. \quad (5.1)$$

¹We note in passing that unlike cross-entropy loss, focal loss in its general form is not a proper loss function, as minimising it does not always lead to the predicted distribution $\hat{p}$ being equal to the target distribution q (see Appendix C.2 for the relevant definition and a longer discussion). However, when q is a one-hot encoding (as in our case, and for most classification tasks), minimizing focal loss does lead to $\hat{p}$ being equal to q .

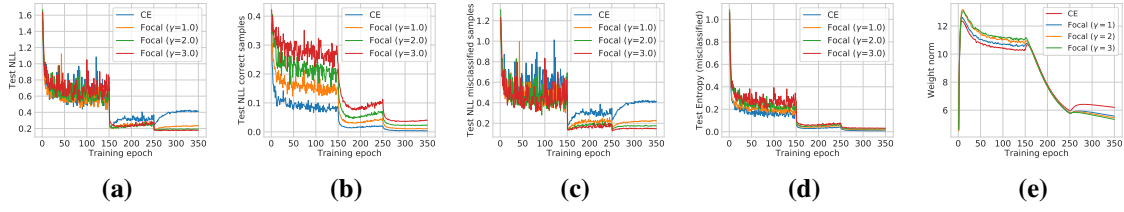


Figure 5.4.1: How metrics related to model calibration change whilst training several ResNet-50 networks on CIFAR-10, using either cross-entropy loss, or focal loss with γ set to 1, 2 or 3.

The most interesting property of this upper bound is that it shows that replacing cross-entropy with focal loss has the effect of adding a maximum-entropy regulariser (Pereyra et al. 2017) to the implicit minimization that was previously being performed. In other words, trying to minimize focal loss minimizes the KL divergence between $\hat{p}$ and q , whilst simultaneously increasing the entropy of the predicted distribution $\hat{p}$. Note, in the case of ground truth with one-hot encoding, only the component of the entropy of $\hat{p}$ corresponding to the ground-truth index, $\gamma(-\hat{p}_{i,y_i} \log \hat{p}_{i,y_i})$, will be maximised (refer Appendix C.2). Encouraging the predicted distribution to have higher entropy can help avoid the overconfident predictions produced by DNNs (see the ‘Peak at the wrong place’ paragraph of §5.3), and thereby improve calibration.

Empirical observations: To analyze the behaviour of neural networks trained on focal loss, we use the same framework as mentioned above, and train four ResNet-50 networks on CIFAR-10, one using cross-entropy loss, and three using focal loss with $\gamma = 1, 2$ and 3. Figure 5.4.1(a) shows that the test NLL for the cross-entropy model significantly increases towards the end of training (before saturating), whereas the NLLs for the focal loss models remain low. To better understand this, we analyze the behaviour of these models for correctly and incorrectly classified samples. Figure 5.4.1(b) shows that even though the NLLs for the correctly classified samples broadly speaking decrease throughout training for all the models, the NLLs for the focal loss models remain consistently higher than that for the cross-entropy model throughout training, implying that the focal loss models are relatively less confident than the cross-entropy model for samples that they predict correctly. This is important, as we have already discussed that it is overconfidence that normally makes deep neural networks miscalibrated. Figure 5.4.1(c) shows that in contrast to the cross-entropy model, for which the NLL for misclassified test samples increases significantly after epoch 150, the rise in this value for the focal loss models is much less severe. Additionally, in Figure 5.4.1(d), we notice that the entropy of the softmax distribution for misclassified test samples is consistently (if marginally) higher for focal loss than for cross-entropy (consistent with Equation 5.1).

Note that from Figure 5.4.1(a), one may think that applying early stopping when training a model on cross-entropy can provide better calibration scores. However, there is no ideal way of doing early stopping that provides the best calibration error and the best test set accuracy. For a fair comparison, we chose 3 intermediate models for each loss function with the best validation set ECE, NLL, and accuracy, and observed that: a) for every stopping criterion, focal loss outperforms cross-entropy in both test set accuracy and ECE, b) when using validation set ECE as a stopping criterion, the intermediate model for cross-entropy indeed improves its test set ECE, but at the cost of a significantly higher test error. Please refer to Appendix C.10 for more details.

As per §5.3, an increase in the test NLL and a decrease in the test entropy for misclassified samples, along with no corresponding increase in the test NLL for the correctly classified samples, can be interpreted as the network starting to predict softmax distributions for the misclassified samples that are ever more peaky in the wrong place. Notably, our results in Figures 5.4.1(b), 5.4.1(c) and 5.4.1(d) clearly show that this effect is significantly reduced when training with focal loss rather than cross-entropy, leading to a better-calibrated network whose predictions are less peaky in the wrong place.

Theoretical justification: As mentioned previously, once a model trained using cross-entropy reaches high training accuracy, the optimizer may try to further reduce the training NLL by increasing the confidence for the correctly classified samples. It may achieve this by magnifying the network weights to increase the magnitudes of the logits. To verify this hypothesis, we plot the L_2 norm of the weights of the last linear layer for all four networks as a function of the training epoch (see Figure 5.4.1(e)). Notably, although the norms of the weights for the models trained on focal loss are initially higher than that for the cross-entropy model, *a complete reversal* in the ordering of the weight norms occurs between epochs 150 and 250. In other words, as the networks start to become miscalibrated, the weight norm for the cross-entropy model also starts to become greater than those for the focal loss models. In practice, this is because focal loss, by design, starts to act as a regulariser on the network’s weights once the model has gained a certain amount of confidence in its predictions. This behaviour of focal loss can be observed even on a much simpler setup like a linear model (see Appendix C.3). To better understand this, we start by considering the following proposition (proof in Appendix C.4):

Proposition 1. *For focal loss $\mathcal{L}_f$ and cross-entropy $\mathcal{L}_c$, the gradients $\frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} g(\hat{p}_{i,y_i}, \gamma)$, where $g(p, \gamma) = (1-p)^\gamma - \gamma p(1-p)^{\gamma-1} \log(p)$, $\gamma \in \mathbb{R}^+$ is the focal loss hyperparameter, and $\mathbf{w}$ denotes the parameters of the last linear layer. Thus $\left\| \frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} \right\| \leq \left\| \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} \right\|$ if $g(\hat{p}_{i,y_i}, \gamma) \in [0, 1]$.*

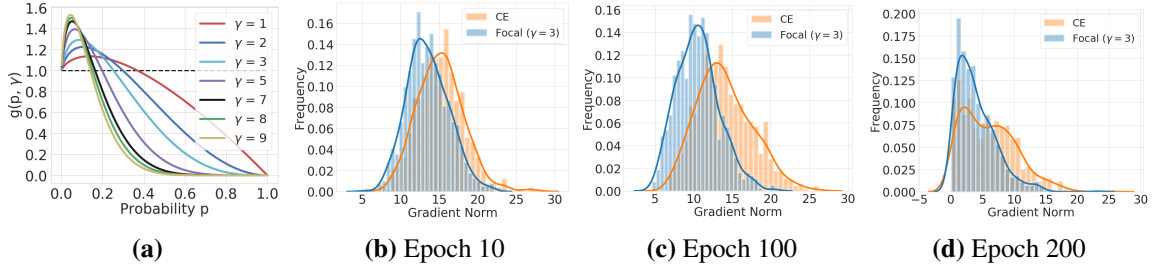


Figure 5.4.2: (a): $g(p, \gamma)$ vs. p and (b-d): histograms of the gradient norms of the last linear layer for both cross-entropy and focal loss.

Proposition 1 shows the relationship between the norms of the gradients of the last linear layer for focal loss and cross-entropy loss, for the same network architecture. Note that this relation depends on a function $g(p, \gamma)$, which we plot in Figure 5.4.2(a) to understand its behaviour. It is clear that for every γ , there exists a (different) threshold p_0 such that for all $p \in [0, p_0]$, $g(p, \gamma) \geq 1$, and for all $p \in (p_0, 1]$, $g(p, \gamma) < 1$. (For example, for $\gamma = 1$, $p_0 \approx 0.4$.) We use this insight to further explain why focal loss provides implicit weight regularisation.

Implicit weight regularisation: For a network trained using focal loss with a fixed γ , during the initial stages of the training, when $\hat{p}_{i, y_i} \in (0, p_0)$, $g(\hat{p}_{i, y_i}, \gamma) > 1$. This implies that the confidence of the focal loss model’s predictions will initially increase faster than they would for cross-entropy. However, as soon as $\hat{p}_{i, y_i}$ crosses the threshold p_0 , $g(\hat{p}_{i, y_i}, \gamma)$ falls below 1 and reduces the size of the gradient updates made to the network weights, thereby having a regularising effect on the weights. This is why, in Figure 5.4.1(e), we find that the weight norms of the models trained with focal loss are initially higher than that for the model trained using cross-entropy. However, as training progresses, we find that the ordering of the weight norms reverses, as focal loss starts regularising the network weights. Moreover, we can draw similar insights from Figures 5.4.2(b), 5.4.2(c) and 5.4.2(d), in which we plot histograms of the gradient norms of the last linear layer (over all samples in the training set) at epochs 10, 100 and 200, respectively. At epoch 10, the gradient norms for cross-entropy and focal loss are similar, but as training progresses, those for cross-entropy decrease less rapidly than those for focal loss, indicating that the gradient norms for focal loss are consistently lower than those for cross-entropy throughout training.

Finally, observe in Figure 5.4.2(a) that for higher γ values, the fall in $g(p, \gamma)$ is steeper. We would thus expect a greater weight regularisation effect for models that use higher values of γ . This explains why, of the three models that we trained using focal loss, the one with $\gamma = 3$ outperforms (in terms of calibration) the one with $\gamma = 2$, which in turn outperforms

the model with $\gamma = 1$. Based on this observation, one might think that, in general, a higher value of gamma would lead to a more calibrated model. However, this is not the case, as we notice from Figure 5.4.2(a) that for $\gamma \geq 7$, $g(p, \gamma)$ reduces to nearly 0 for a relatively low value of p (around 0.5). As a result, using values of γ that are too high will cause the gradients to die (*i.e.*, reduce to nearly 0) early, at a point at which the network's predictions remain ambiguous, thereby causing the training process to fail.

How to choose γ : As discussed, focal loss provides implicit entropy and weight regularisation, which heavily depend on the value of γ . Finding an appropriate γ is normally done using cross-validation. Also, traditionally, γ is fixed for all samples in the dataset. However, as shown, the regularisation effect for a sample i depends on $\hat{p}_{i,y_i}$, *i.e.*, the predicted probability for the ground truth label for the sample. It thus makes sense to choose γ for each sample based on the value of $\hat{p}_{i,y_i}$. To this end, we provide Proposition 2 (proof in Appendix C.4), which we use to find a solution to this problem:

Proposition 2. *Given a p_0 , for $1 \geq p \geq p_0 > 0$, $g(p, \gamma) \leq 1$ for all $\gamma \geq \gamma^* = \frac{a}{b} + \frac{1}{\log a} W_{-1}\left(-\frac{a^{(1-a/b)}}{b} \log a\right)$, where $a = 1 - p_0$, $b = p_0 \log p_0$, and W_{-1} is the Lambert-W function (Corless et al. 1996). Moreover, for $p \geq p_0 > 0$ and $\gamma \geq \gamma^*$, the equality $g(p, \gamma) = 1$ holds only for $p = p_0$ and $\gamma = \gamma^*$.*

It is worth noting that there exist multiple values of γ where $g(p, \gamma) \leq 1$ for all $p \geq p_0$. For a given p_0 , Proposition 2 allows us to compute γ s.t. (i) $g(p_0, \gamma) = 1$; (ii) $g(p, \gamma) \geq 1$ for $p \in [0, p_0]$; and (iii) $g(p, \gamma) < 1$ for $p \in (p_0, 1]$. This allows us to control the magnitude of the gradients for a particular sample i based on the current value of $\hat{p}_{i,y_i}$, and gives us a way of obtaining an informed value of γ for each sample. For instance, a reasonable policy might be to choose γ s.t. $g(\hat{p}_{i,y_i}, \gamma) > 1$ if $\hat{p}_{i,y_i}$ is small (say less than 0.25), and $g(\hat{p}_{i,y_i}, \gamma) < 1$ otherwise. Such a policy will have the effect of making the weight updates larger for samples having a low predicted probability for the correct class and smaller for samples with a relatively higher predicted probability for the correct class.

Following the aforementioned arguments, we choose a threshold p_0 of 0.25, and use Proposition 2 to obtain a γ policy such that $g(p, \gamma)$ is observably greater than 1 for $p \in [0, 0.25]$ and $g(p, \gamma) < 1$ for $p \in (0.25, 1]$. In particular, we use the following schedule: if $\hat{p}_{i,y_i} \in [0, 0.2)$, then $\gamma = 5$, otherwise $\gamma = 3$ (note that $g(0.2, 5) \approx 1$ and $g(0.25, 3) \approx 1$: see Figure 5.4.2(a)). We find this γ policy to perform consistently well across multiple classification datasets and network architectures. Having said that, one can calculate multiple

such schedules for γ following Proposition 2, using the intuition of having a relatively high γ for low values of $\hat{p}_{i,y_i}$ and a relatively low γ for high values of $\hat{p}_{i,y_i}$.

5.5 Experiments

We conduct image and document classification experiments to test the performance of focal loss. For the former, we use CIFAR-10/100 (Krizhevsky 2009) and Tiny-ImageNet (Deng et al. 2009), and train ResNet-50, ResNet-110 (He et al. 2016), Wide-ResNet-26-10 (Zagoruyko and Komodakis 2016) and DenseNet-121 (Huang et al. 2017) models, and for the latter, we use 20 Newsgroups (Lang 1995) and Stanford Sentiment Treebank (SST) (Socher et al. 2013) datasets and train Global Pooling CNN (Lin et al. 2014a) and Tree-LSTM (Tai et al. 2015) models. Further details on the datasets and training can be found in Appendix C.5.

Baselines Along with cross-entropy loss, we compare our method against the following baselines: a) *MMCE* (Maximum Mean Calibration Error) (Kumar et al. 2018), a continuous and differentiable proxy for calibration error that is normally used as a regulariser alongside cross-entropy, b) *Brier loss* (Brier 1950), the squared error between the predicted softmax vector and the one-hot ground truth encoding (Brier loss is an important baseline as it can be decomposed into calibration and refinement (DeGroot and Fienberg 1983)), and c) *Label smoothing* (Müller et al. 2019) (LS): given a one-hot ground-truth distribution $\mathbf{q}$ and a smoothing factor α (hyperparameter), the smoothed vector $\mathbf{s}$ is obtained as $s_i = (1 - \alpha)\mathbf{q}_i + \alpha(1 - \mathbf{q}_i)/(K - 1)$, where s_i and $\mathbf{q}_i$ denote the i^{th} elements of $\mathbf{s}$ and $\mathbf{q}$ respectively, and K is the number of classes. Instead of $\mathbf{q}$, $\mathbf{s}$ is treated as the ground truth. We train models using $\alpha = 0.05$ and $\alpha = 0.1$ but find $\alpha = 0.05$ to perform better. We thus report the results obtained from LS-0.05 with $\alpha = 0.05$.

Focal Loss: As mentioned in §5.4, our proposed approach is the sample-dependent schedule FLSD-53 ($\gamma = 5$ for $\hat{p}_y \in [0, 0.2)$, and $\gamma = 3$ for $\hat{p}_y \in [0.2, 1]$), which we find to perform well across most classification datasets and network architectures. In addition, we also train other focal loss baselines, including ones with γ fixed to 1, 2 and 3, and also ones that have a training epoch-dependent schedule for γ . Among the focal loss models trained with a fixed γ , using validation sets we find $\gamma = 3$ (FL-3) to perform the best. Details of all these approaches can be found in Appendix C.6.

Temperature Scaling: To compute the optimal temperature, we use two different methods: (a) learning the temperature by minimizing validation set NLL, and (b) performing a grid search over temperatures between 0 and 10, with a step of 0.1, and finding the one that

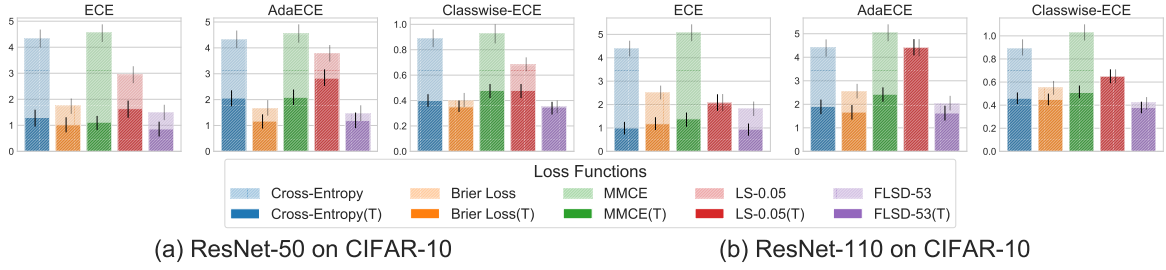


Figure 5.5.1: Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE, computed for ResNet-50 (first 3 figures) and ResNet-110 (last 3 figures) on CIFAR-10. For each loss, the calibration metric before temperature scaling is represented by the dark colour and after temperature scaling is represented by the light colour. Lower is better. Clearly, our proposed FLSD-53 provides better results both before and after temperature scaling.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FL-3 (Ours)		FLSD-53 (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	17.52	3.42(2.1)	6.52	3.64(1.1)	15.32	2.38(1.8)	7.81	4.01(1.1)	5.13	1.97(1.1)	4.5	2.0(1.1)
	ResNet-110	19.05	4.43(2.3)	7.88	4.65(1.2)	19.14	3.86(2.3)	11.02	5.89(1.1)	8.64	3.95(1.2)	8.56	4.12(1.2)
	Wide-ResNet-26-10	15.33	2.88(2.2)	4.31	2.7(1.1)	13.17	4.37(1.9)	4.84	4.84(1)	2.13	2.13(1)	3.03	1.64(1.1)
	DenseNet-121	20.98	4.27(2.3)	5.17	2.29(1.1)	19.13	3.06(2.1)	12.89	7.52(1.2)	4.15	1.25(1.1)	3.73	1.31(1.1)
CIFAR-10	ResNet-50	4.35	1.35(2.5)	1.82	1.08(1.1)	4.56	1.19(2.6)	2.96	1.67(0.9)	1.48	1.42(1.1)	1.55	0.95(1.1)
	ResNet-110	4.41	1.09(2.8)	2.56	1.25(1.2)	5.08	1.42(2.8)	2.09	2.09(1)	1.55	1.02(1.1)	1.87	1.07(1.1)
	Wide-ResNet-26-10	3.23	0.92(2.2)	1.25	1.25(1)	3.29	0.86(2.2)	4.26	1.84(0.8)	1.69	0.97(0.9)	1.56	0.84(0.9)
	DenseNet-121	4.52	1.31(2.4)	1.53	1.53(1)	5.1	1.61(2.5)	1.88	1.82(0.9)	1.32	1.26(0.9)	1.22	1.22(1)
Tiny-ImageNet	ResNet-50	15.32	5.48(1.4)	4.44	4.13(0.9)	13.01	5.55(1.3)	15.23	6.51(0.7)	1.87	1.87(1)	1.76	1.76(1)
20 Newsgroups	Global Pooling CNN	17.92	2.39(3.4)	13.58	3.22(2.3)	15.48	6.78(2.2)	4.79	2.54(1.1)	8.67	3.51(1.5)	6.92	2.19(1.5)
SST Binary	Tree-LSTM	7.37	2.62(1.8)	9.01	2.79(2.5)	5.03	4.02(1.5)	4.84	4.11(1.2)	16.05	1.78(0.5)	9.19	1.83(0.7)

Table 5.5.1: ECE (%) computed for different approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets. $T \approx 1$ indicates an innately calibrated model.

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	FL-3 (Ours)	FLSD-53 (Ours)
CIFAR-100	ResNet-50	23.3	23.39	23.2	23.43	22.75	23.22
	ResNet-110	22.73	25.1	23.07	23.43	22.92	22.51
	Wide-ResNet-26-10	20.7	20.59	20.73	21.19	19.69	20.11
	DenseNet-121	24.52	23.75	24.0	24.05	23.25	22.67
CIFAR-10	ResNet-50	4.95	5.0	4.99	5.29	5.25	4.98
	ResNet-110	4.89	5.48	5.4	5.52	5.08	5.42
	Wide-ResNet-26-10	3.86	4.08	3.91	4.2	4.13	4.01
	DenseNet-121	5.0	5.11	5.41	5.09	5.33	5.46
Tiny-ImageNet	ResNet-50	49.81	53.2	51.31	47.12	49.69	49.06
20 Newsgroups	Global Pooling CNN	26.68	27.06	27.23	26.03	29.26	27.98
SST Binary	Tree-LSTM	12.85	12.85	11.86	13.23	12.19	12.8

Table 5.5.2: Test set error (%) computed for different approaches.

minimizes validation set ECE. We find the second approach to produce *stronger baselines* and report results obtained using this approach.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FL-3 (Ours)		FLSD-53 (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-10/SVHN	ResNet-110	61.71	59.66	94.80	95.13	85.31	85.39	68.68	68.68	96.74	96.92	90.83	90.97
	Wide-ResNet-26-10	96.82	97.62	94.51	94.51	97.35	97.95	84.63	84.66	98.19	98.05	98.29	98.20
CIFAR-10/CIFAR-10-C	ResNet-110	77.53	75.16	84.09	83.86	71.96	70.02	72.17	72.18	82.27	82.18	85.05	84.70
	Wide-ResNet-26-10	81.06	80.68	85.03	85.03	82.17	81.72	71.10	71.16	82.17	81.86	87.05	87.30

Table 5.5.3: AUROC (%) computed for models trained on CIFAR-10 (in-distribution), and using SVHN and CIFAR-10-C (Gaussian Noise corruption with severity level 5) respectively as the OoD datasets.

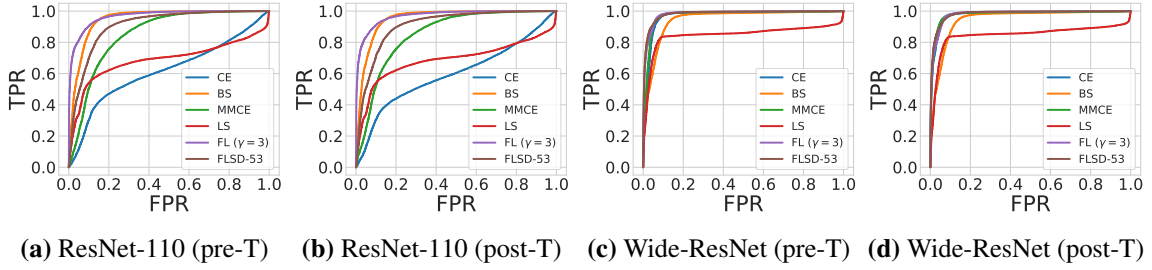


Figure 5.5.2: ROC plots obtained from ResNet-110 and Wide-ResNet-26-10 architectures trained on CIFAR-10 (in-distribution) and tested on SVHN (OoD), both pre and post-temperature scaling.

Performance Gains: We report $ECE\%$ (computed using 15 bins) along with optimal temperatures in Table 5.5.1, and test set error in Table 5.5.2. We report the other calibration scores (AdaECE, Classwise-ECE, MCE, and NLL) in Appendix C.6. Firstly, for all dataset-network pairs, we obtain very competitive classification accuracies (shown in Table 5.5.2). Secondly, it is clear from Table 5.5.1, and Tables C.6.1 and C.6.2 in the appendix, that focal loss with sample-dependent γ and with $\gamma = 3$ outperform all the baselines: cross-entropy, label smoothing, Brier loss and MMCE. They broadly produce the lowest calibration errors both before and after temperature scaling. This observation is particularly encouraging, as it also indicates that a principled method of obtaining values of γ for focal loss can produce a very calibrated model, with no need to use a validation set for tuning γ . As shown in Figure 5.5.1, we also compute 90% confidence intervals for ECE, AdaECE and Classwise-ECE using 1000 bootstrap samples following Kumar et al. (2019), and using ResNet-50/110 trained on CIFAR-10 (see Appendix C.7 for more results). Note that FLSD-53 produces the lowest calibration errors in general, and the difference in the metric values between FLSD-53 and other approaches (except Brier loss) is mostly statistically significant (*i.e.*, confidence intervals don't overlap), especially before temperature scaling. In addition to the lower calibration errors, there are other advantages of focal loss as well, which we explore next.

More advantages of focal loss: *Behaviour on Out-of-Distribution (OoD) data:* A

perfectly calibrated model should have low confidence whenever it misclassifies, including when it encounters data which is OoD (Thulasidasan et al. 2019). Although temperature scaling calibrates a model under the i.i.d. assumption, it is known to fail under distributional shift (Ovadia et al. 2019). Since focal loss has implicit regularisation effects on the network (see §5.4), we investigate if it helps to learn representations that are more robust to OoD data. To do this, we use ResNet-110 and Wide-ResNet-26-10 trained on CIFAR-10 and consider the SVHN (Netzer et al. 2011) test set and CIFAR-10-C (Hendrycks and Dietterich 2019) with Gaussian noise corruption at severity 5 as OoD data. We use the entropy of the softmax distribution as the measure of confidence or uncertainty, and report the corresponding AUROC scores both before and after temperature scaling in Table 5.5.3. For both SVHN and CIFAR-10-C (using Gaussian noise), models trained on focal loss obtain the highest AUROC scores. *Note that Focal loss even without temperature scaling performs better than other methods with temperature scaling.* We also present the ROC plots pre and post-temperature scaling for models trained on CIFAR-10 and tested on SVHN in Figure 5.5.2. Thus, it is quite encouraging to note that models trained on focal loss are not only better calibrated under the i.i.d. assumption, but also seem to perform better than other competitive loss functions when we try shifting the distribution from CIFAR-10 to SVHN or CIFAR-10-C (pre and post temperature scaling).

Confident and Calibrated Models: It is worth noting that focal loss with sample-dependent γ has optimal temperatures that are very close to 1, mostly lying between 0.9 and 1.1 (see Table 5.5.1). This property is shown by the Brier loss and label smoothing models as well, albeit with worse calibration errors. By contrast, the temperatures for cross-entropy and MMCE models are significantly higher, with values lying between 2.0 and 2.8. An optimal temperature close to 1 indicates that the model is innately calibrated, and cannot be made significantly more calibrated by temperature scaling. A temperature much greater than 1 can make a model underconfident in general, as it is applied irrespective of the correctness of model outputs. We observe this empirically for ResNet-50 and ResNet-110 trained on CIFAR-10. Although models trained with cross-entropy have much higher confidence before temperature scaling than those trained with focal loss, after temperature scaling, focal loss models are significantly more confident in their predictions. We provide quantitative and qualitative empirical results to support this claim in Appendix C.8.

5.6 Conclusion

In this chapter, we have studied the properties of focal loss, an alternative loss function that can yield classification networks that are more naturally calibrated than those trained using the conventional cross-entropy loss, while maintaining accuracy. In particular, we show in §5.4 that focal loss implicitly maximizes entropy while minimizing the KL divergence between the predicted and the target distributions. We also show that, because of its design, it naturally regularises the weights of a network during training, reducing NLL overfitting and thereby improving calibration. Furthermore, we empirically observe that models trained using focal loss are not only better calibrated under i.i.d. assumptions, but can also be better at detecting OoD samples which we show by taking CIFAR-10 as the in-distribution dataset and SVHN and CIFAR-10-C as out-of-distribution datasets, something that temperature scaling fails to achieve.

6

Similarity Learning for One-Shot Video Object Segmentation

Abstract

One-shot video object segmentation is an emerging problem in computer vision with many interesting real-world applications. To tackle this, we introduce a similarity learning approach, that can learn to perform dense label transfer from one image to the other. More specifically, the objective is to learn a similarity metric for dense pixel-wise correspondence between two images. This learned model can then be used in a label transfer framework to propagate object annotation from the reference frame to all the subsequent frames in a video. Unlike previous methods, our similarity learning approach works fairly well across various domains, even when no domain adaptation is involved. We demonstrate the effectiveness of our method on two standard datasets with favourable results. Using our approach, we also achieved second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

6.1 Introduction

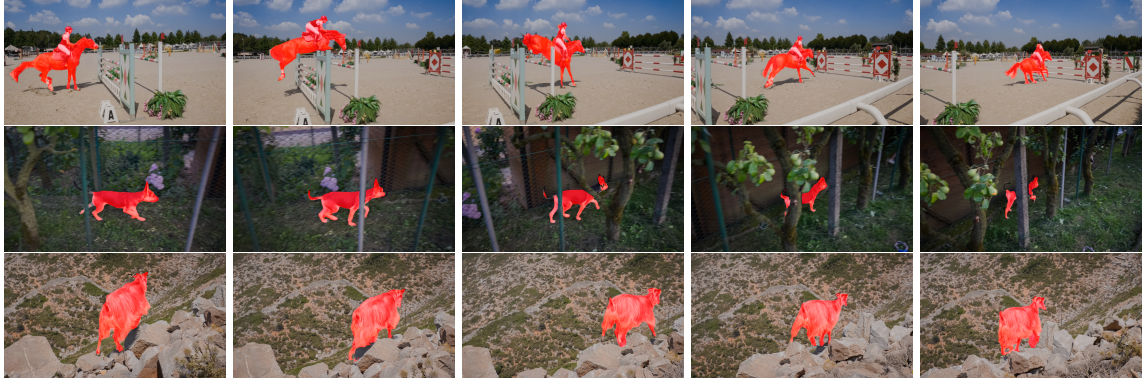


Figure 6.1.1: One-shot segmentation outputs for sample videos from DAVIS-2016, obtained using our model without any test domain adaptation, online learning or interaction. Each row illustrates outputs for different time frames of the video.

Object segmentation in a video is an important problem in computer vision with many interesting real-world applications in surveillance, robotics, and autonomous driving. Given one or more target objects, the task is to track and segment those objects in every frame of the video. In this work, we consider the scenario where the ground truth segmentation mask, marking the objects of interest, is given only for the first frame of the video, and the task is to segment all subsequent frames; such a problem is often referred to as *one-shot video object segmentation*. In contrast to tracking, this is a far more challenging problem, as the algorithm has to produce pixel-wise segmentation masks at each frame instead of merely locating the objects, however tracking also expects to know how the object is moving. In a nutshell, to produce robust segmentation masks throughout the video, the algorithm should be able to adapt to the pose and appearance changes of the targets, in addition to being robust to these variations.

Recently, to tackle one-shot video object segmentation, several methods have been proposed (Caelles et al. 2017; Voigtlaender and Leibe 2017; Perazzi et al. 2017; Khoreva et al. 2017b; Jampani et al. 2017; Li et al. 2017). To adapt to the dynamic variations in the video and produce fine-grained segmentation masks, most of these methods either rely on off-the-shelf object detectors (Li et al. 2017), superpixel segmentation methods (Caelles et al. 2017), pre-trained objectness models (Voigtlaender and Leibe 2017), or heavily engineered fine-tuning procedure (Voigtlaender and Leibe 2017; Khoreva et al. 2017b; Li et al. 2017). Even though these methods perform well on the standard benchmarks, their overly complex nature

and high running time make such approaches less attractive to real-world applications. In fact, for most of these baselines, the boost in performance is mainly due to the heavy engineering involved in the fine-tuning phase, and removing it often leads to dramatic degradation in performance. More recently, Heo et al. (2020) uses two different modules for converting user scribbles to an object mask, and propagating the object mask bidirectionally across the video to the other frames both directly and temporally. Further, Cheng et al. (2021) uses three different modules for converting user scribbles to an object mask, temporally propagating the object mask across video, and difference-aware module to fuse the masks before and after each user interaction by using space-time memory. As opposed to these methods, we do not use temporal propagation of object masks but only directly propagate from the scribble frame to the target frame.

In this chapter, we introduce a simple and flexible method for one-shot video object segmentation based on *similarity learning*. More specifically, we turn the problem of video object segmentation into a dense binary classification problem using a Siamese network, where the objective is to predict whether or not a pair of selected pixels from two frames belong to the same object (or instance of an object). Our goal here is to learn an embedding space in which, pixels of the same object are encouraged to be close to each other, whereas pixels of different objects are pushed away from each other. In short, we train a similarity metric for pixel correspondence between two images. We then use the trained model in a *label transfer* framework to propagate object annotations from a frame with ground-truth information (*e.g.*, the first frame of the video in case of the one-shot video object segmentation task) to other frames. Note that similarity learning has been successfully used for visual object recognition, verification, and tracking in the past (Koch et al. 2015; Vinyals et al. 2016; Parkhi et al. 2015; Bertinetto et al. 2016b; Bertinetto et al. 2016a) but to the best of our knowledge this is the first time such a technique has been transposed to video object segmentation.

We would like to point out that, with this simple similarity learning approach, our method performs fairly well across various domains and outperforms state-of-the-art methods when no domain adaptation is involved. However, note that our method also has the flexibility of incorporating a domain adaptation step which can further improve the robustness of our method against dynamic changes in the video. Furthermore, unlike previous approaches (Voigtlaender and Leibe 2017), multi-target segmentation comes at virtually no cost in our approach. Figure 6.1.1 demonstrates qualitative results computed using our method for one-shot object segmentation on sample videos, with no test domain adaptation, online learning, or human interaction involved.

We demonstrate the effectiveness of our method on both DAVIS-2016 (Perazzi et al. 2016) and DAVIS-2017 (Pont-Tuset et al. 2017) datasets with favourable results. We also consider the scenario of *interactive* video object segmentation with scribbles, as suggested by Caelles et al. (2018). In this process, the segmentation system learns interactively by getting feedback from the evaluation system. Moreover, at each interaction, the method is fine-tuned using only a set of squiggles from one video frame, which is much easier to obtain compared to dense ground-truth object masks but makes the problem more challenging. Following this protocol, we developed an interactive segmentation system based on our dense similarity learning and label transfer method, and achieved second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

6.2 Related Work

One-shot video object segmentation has gained considerable popularity in the last couple of years with the release of new benchmarks such as DAVIS (Perazzi et al. 2016) and strong baseline methods such as *OSVOS* (Caelles et al. 2017), and *MaskTrack* (Perazzi et al. 2017). Caelles et al. (2017) proposed a simple but effective strategy, in which a fully convolutional network is developed for segmenting the objects of interest in each video frame. In particular, the fully convolutional network (pre-trained on Image-Net) is initially trained to densely segment the foreground within each frame in the training dataset. This network is then adapted to the test video through a fine-tuning process, given the dense ground-truth segmentation of the first frame.

Subsequently, more sophisticated approaches were proposed based on the *OSVOS* framework, which further improved the state-of-the-art on video-object segmentation task (Voigtlaender and Leibe 2017; Shaban et al. 2017). For example, in *OnAVOS* (Voigtlaender and Leibe 2017), the authors proposed to further adapt the *OSVOS* model to the test domain by performing online fine-tuning throughout the video. More specifically, they used highly confident predictions in the test video frames as training samples to gradually adapt the model to new scenes and make it more robust against sharp and significant changes in the object’s shape and pose during the video.

In contrast to *OSVOS* which processes each frame independently, *MaskTrack* proposed leveraging the temporal information by using the prediction masks at each time point as an input to the next time frame. More specifically, they predict the object mask at time t given the video frame at time t together with the predicted mask at time $t - 1$, which is

fed to the fully convolutional network via a fourth input channel. This idea was further improved by Khoreva et al. (2017b) and Li et al. (2017) using complementary modules such as data augmentation, re-identification, and object detectors. In particular, Khoreva et al. (2017b) proposed a complex data augmentation technique, called *Lucid Data Dreaming*, to produce potential future frames for the first frame, whose ground-truth annotation is available during test time. This allows the system to be fine-tuned on a larger set of images, instead of only the individual first frame of the video. This data augmentation technique has been extensively used in other video object segmentation approaches as well. Li et al. (2017) augment *MaskTrack* with a re-identification module that traverses the video back and forth for any potential instance of the object that might have gone missing during the video. Once such an instance is found, its mask is propagated bidirectionally to segment that object in other frames. This approach, however, may not be practical for streaming purposes, where the decision on a frame may not be refined at later times. Moreover, most of the above-mentioned works need to be repeated in the case of multiple object segmentation, which is not desirable. In contrast, our approach requires only one forward pass for all the objects to be segmented and does not utilize any future information through this process.

Video Propagation Network (VPN) (Jampani et al. 2017) is a recent technique that proposes a temporal bilateral filtering network for label propagation. Similar to our model, this approach also produces satisfactory results without any extra test domain adaptation. This is unlike the above-mentioned approaches which are very dependent on the fine-tuning step.

6.3 Method

Our approach builds upon the standard Siamese network to learn the dense correspondence between the target objects in two given images. To this end, we first review the Siamese architecture and then introduce our idea for using them in a dense label transfer framework.

6.3.1 Siamese Networks

The Siamese network is a special type of neural network, which is trained to make comparisons between pairs of input data. In other words, it learns whether two input items belong to the same class or not. Figure 6.3.1 illustrates an example of a Siamese network that is used in our framework.

Let $\{x, y\}$ be a pair of input data (usually a pair of images) which is fed into two identical deep neural networks with a shared set of parameters $\mathcal{W}$. These networks generate two

embeddings $\{\Phi(\mathbf{x}; \mathcal{W}), \Phi(\mathbf{y}; \mathcal{W})\}$ at their outputs. The inner product of these embeddings can be considered as the similarity between $\mathbf{x}$ and $\mathbf{y}$. Also let ℓ be the target of the Siamese network which is defined as $\ell = \mathbb{1}[l_{\mathbf{x}} == l_{\mathbf{y}}]$, where $l_{\mathbf{x}}$ and $l_{\mathbf{y}}$ are the class labels of $\mathbf{x}$ and $\mathbf{y}$, respectively. In other words, ℓ is set to 1 if the input pair has the same class labels and 0 otherwise. Now the objective of the standard Siamese network can be defined as:

$$\min_{\mathcal{W}} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\langle \Phi(\mathbf{x}_i; \mathcal{W}), \Phi(\mathbf{y}_i; \mathcal{W}) \rangle, \ell_i), \quad (6.1)$$

where i ranges over the training set of data pairs of size N , $\mathcal{L}$ is a standard loss function, *e.g.*, binary cross-entropy, and $\langle \cdot, \cdot \rangle$ denotes the inner-product operation. The hypothesis is that the embedding space obtained by this optimization process corresponds to a similarity metric, in which, data of similar classes are clustered together, whereas dissimilar data points are sufficiently far from each other.

6.3.2 Object Segmentation using Siamese Networks

In this section, we extend the idea of similarity learning for object segmentation. Figure 6.3.1 illustrates the proposed structure, where each branch of the Siamese network is a Deeplab-Resnet-101 (Chen et al. 2017a) model with dilated convolutions to preserve spatial information, connected to a decoder network which upsamples the embeddings to a higher resolution. Note that, as shown in Chen et al. (2017a) and Long et al. (2015), preserving spatial information is crucial for segmentation tasks. The outputs of the decoders are two pixel embedding maps $\{\Phi(\mathbf{x}; \mathcal{W}), \Phi(\mathbf{y}; \mathcal{W})\}$, where $\mathbf{x}$ and $\mathbf{y}$ are input images of size $[H, W]$ and Φ is a three-dimensional tensor of size $[h, w, d]$, where $h = H/2$, $w = W/2$ and d is the embedding size.

Next, we compute the similarities of the pixels between images $\mathbf{x}$ and $\mathbf{y}$ by taking the inner product of their embeddings. The output of this step is a dense similarity map of size $[h \times w, h \times w]$, which indicates the similarity of every pixel embedding in $\Phi_{\mathbf{x}}$ with all the pixel embeddings in $\Phi_{\mathbf{y}}$. At this point, we apply batch-norm (Ioffe and Szegedy 2015) to the output of the inner product to prevent the gradients from exploding. Finally, sigmoid nonlinearity is applied to convert the similarity scores into probabilities. Furthermore, the target of the Siamese network is a matrix of size $[h \times w, h \times w]$ (denoted as $\mathbf{L}$) which is a binary map that shows if two pixels in the images belong to the same object class.

The proposed network is trained according to the objective function in Eq. 6.1, but for an aggregate loss over $[h \times w, h \times w]$ data points for each pair of images. After training,

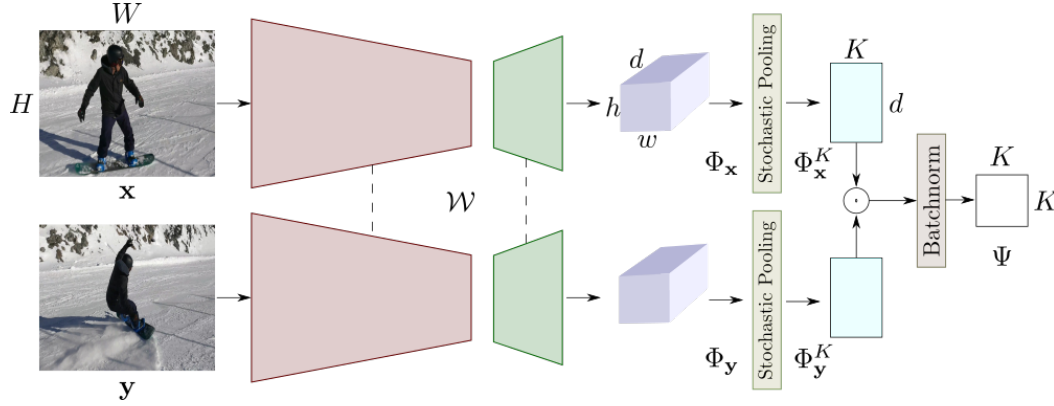


Figure 6.3.1: The Siamese architecture used for learning dense correspondence. The model takes in a pair of images and outputs the similarity of their pixels by computing the inner product of their subsampled pixel embeddings ($\langle \Phi_x^K, \Phi_y^K \rangle$). During the evaluation, stochastic pooling is not applied on the target image y and the inner-product output is $\langle \Phi_x^K, \Phi_y \rangle$.

the output similarity map can be used to predict the label correspondence between pixels of images x and y . Then, during test time, we leverage this similarity map to predict the segmentation map of the second image y , given a ground-truth segmentation map of the first image x , via a *label transfer* approach, as described in Sec. 6.3.2.2.

The main challenge when using the Siamese architecture to learn dense correspondence is the high memory requirement. Note that, the output of the inner-product $\Psi = \langle \Phi_x, \Phi_y \rangle$ is of dimension $[h \times w, h \times w]$. In our case, $h = 240$ and $w = 427$, which leads to 1×10^{10} floating-point values to be stored. This is infeasible to store on a standard GPU. To tackle this, we present a simple stochastic pooling approach.

6.3.2.1 Stochastic Pooling

As mentioned earlier, it is intractable to compute the dense pixel-wise correspondence in the naïve way. Therefore, the idea is to approximate the inner-product $\Psi = \langle \Phi_x, \Phi_y \rangle$ using only a subset of pixels¹.

We observe that, in an image, a large portion of the pixels are highly correlated to each other. Hence, it is possible to estimate the embedding of the image pixels using only a small fraction of them. Based on this argument, we introduce a stochastic pooling layer to the network, that subsamples k pixels uniformly at random for each object class present in the

¹In this section, for brevity, we call the pixel embedding as a pixel. In particular, the embedding Φ_x of image x can be thought of as an image (of size $[h \times w]$ with d dimensional feature), and a pixel precisely refers to a pixel in this image.

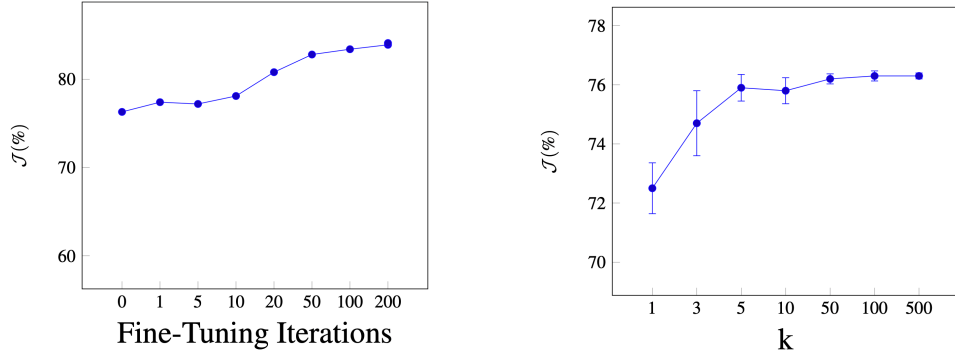


Figure 6.3.2: a) Left image: pixel embeddings are highly correlated and selecting more than $k = 100$ pixels via Stochastic Pooling does not affect the performance (The results are obtained without adaptation.) b) Right image: segmentation performance improves by increasing the number of fine-tuning iterations on the first frame of the test video.

image. This can be ensured, as the ground-truth information is available for all the pixels in the training phase. Thus, total K pixels are chosen from the image. This subsampling procedure is done for both the embeddings Φ_x and Φ_y independently and an equal number of samples are chosen. Now, the inner product can be approximated as:

$$\Psi = \langle \Phi_x, \Phi_y \rangle \approx \langle \Phi_x^K, \Phi_y^K \rangle, \quad (6.2)$$

where Φ_x^K and Φ_y^K represent the embeddings corresponding to the chosen subsets of x and y , respectively. Note that, Ψ is of size $[K, K]$, where K is the chosen subset size. Also, note that the same chosen subsets are used to create the target map L^K for the Siamese network. In Figure 6.3.2, we experimentally demonstrate that the performance plateaus for $k > 100$, which supports the above argument. This effectively reduces the memory requirement from 1×10^{10} to 1×10^4 (assuming one object in each image), and also makes it faster in both the training and inference phases.

We would like to point out that our stochastic pooling can be thought of as a subset or subspace selection and more sophisticated strategies such as principal component analysis (Aubry and Russell 2015) or low-rank factorization (Chandra et al. 2017) can be employed. We leave this for future work.

6.3.2.2 Label Transfer

Once the network is trained, one can label a query image y by matching its pixels with the pixels of a reference image x (whose ground-truth annotation is available) using the learned similarity model. At test time, the reference image is fed into the top branch

of the Siamese network, while the query image is given to the bottom branch. Then a representative set of pixel embeddings are subsampled from the reference image via the stochastic pooling layer, and their similarities with the embeddings of the query pixels are computed. Subsequently, the class labels of the selected reference pixels are transferred to all the query pixels based on their similarity scores. Note that, in this step, the stochastic pooling is applied only to the reference image and the similarity scores using the inner product are computed for all the query pixels, *i.e.*, $\Psi \approx \langle \Phi_x^K, \Phi_y \rangle$ where Φ_x^K is of size $[K, d]$, Φ_y is of size $[h \times w, d]$, and Ψ is of size $[K, h \times w]$.

Let $\mathcal{N}_{l'}$ be the selected subset of pixels from the reference image for class l' , then the class label of pixel $i \in \mathcal{Q}$ within the query image $\mathcal{Q}$ is predicted using the following formulation:

$$l_i = \arg \max_{l'} \sum_{j \in \mathcal{N}_{l'}} \tilde{\Psi}_{ij}, \quad (6.3)$$

where $\tilde{\Psi}$ is the similarity score after applying the batch norm.

This completes our description of the similarity learning model for dense label transfer. As evidenced by our experiments, this simple model outperforms state-of-the-art methods in various domains when no domain adaptation techniques are used. Nonetheless, for completeness, we apply domain adaptation to our method to have a more thorough comparison with the state-of-the-art.

6.3.3 Domain Adaptation

Although our method obtains competitive results without any adaptation to the test domain (as shown in Figure 6.1.1 and Table 6.4.1), it can be further improved by adapting the learned similarity model to the test scene and objects. At test time, since the ground-truth segmentation of the first frame of the test video is available, we use this information to fine-tune our Siamese network for a fixed number of iterations in a one-shot setup. This step essentially adapts the learned embedding space to accommodate the new scene. Figure 6.4.2 illustrates how domain adaptation improves the segmentation results for two videos.

Furthermore, note that there are more complex domain adaptation methods that can be employed, such as online fine-tuning (Voigtlaender and Leibe 2017; Khoreva et al. 2017b; Li et al. 2017). Even though, this improves the segmentation accuracy, in our opinion, the heavy dependence on hyperparameter tuning and high running time make such models less attractive for real-world applications. Therefore, it is important to develop segmentation

models that are simple and robust to dynamic changes in the video and we believe, our similarity learning approach takes one step closer to this goal.

6.4 Experiments

6.4.1 Datasets

DAVIS-2016 This dataset was introduced by Perazzi et al. (2016) as a standard benchmark for video object segmentation task. It consists of 50 videos with dense ground-truth annotations of one foreground object throughout each video. The dataset is split into two partitions the training set (30 videos) and the validation set (20 videos). Videos are released in both Full-HD and 480p resolutions. However, all the experiments in this chapter are performed on 480×854 resolution videos.

DAVIS-2017 This dataset (Pont-Tuset et al. 2017) is a super-set of DAVIS-2016 that contains 150 videos (with similar resolutions to those in DAVIS-2016) and it is subdivided into four different splits of training (60 videos), validation (30 videos), test-dev (30 videos), and test-challenge (30 videos). The main addition to this dataset is multi-object segmentation, where the number of objects/instances to be segmented per video ranges from one to ten. Also, note that, since there is no overlap between the train split of DAVIS-2017 and the validation set of DAVIS-2016, we use the DAVIS-2017 training set for training, and test on both datasets.

6.4.2 Experiment Setup

Here we first introduce the evaluation metrics for the video object segmentation that were used in this chapter, next we describe the interactive segmentation task and also detail the corresponding evaluation metrics. Then we describe the network structure in detail, after which we elaborate on the training and label transfer stages, and lastly we present the experiments and obtained results.

Evaluation metrics for video object segmentation Following the standard benchmarks for video object segmentation, we report our results based on two metrics: *Jaccard* index ($\mathcal{J}$) and boundary F-score ($\mathcal{F}$). The former ($\mathcal{J}$) is defined as the intersection-over-union (IoU) of the predicted object mask and its respective ground-truth mask. The latter ($\mathcal{F}$)

indicates how accurate the boundaries of the predicted mask are with respect to the ground-truth boundaries (Pont-Tuset et al. 2017). One can obtain per-object evaluation results by computing the average of these two metrics over the video that contains the object. The final evaluation result for a dataset is defined as the average of per-object performances.

Interactive video object segmentation task As the proposed approach can learn similarity matrix potentially from very few labelled pixels. Hence, it was also evaluated using the interactive toolkit released by Caelles et al. (2018). Initially, a set of squiggles representing the objects of interest in one of the video frames are provided. The segmentation system is fine-tuned based on this sparse set of ground-truth information and predicts pixel-level masks for targets over the entire video. The outputs are then analyzed, and the video frame with the worst segmentation output is chosen to provide the user with feedback on which regions the method has failed. Note that the feedback is also in the form of scribbles. This interaction improves the segmentation system, as it learns where it fails the most and updates its parameters such that those mistakes are resolved. We evaluated our method interactively on DAVIS-2017 (Pont-Tuset et al. 2017) validation (30 videos) and Test-dev (30 videos) sets. For validation, we used a model pre-trained on the DAVIS-2017 training set (60 videos), however in the test phase, we pre-trained our model using the entire training+validation set (90 videos).

Evaluation metrics for interactive video object segmentation Two evaluation metrics are designed for the interactive task by Caelles et al. (2018), which take into account both segmentation accuracy and speed. The first one is the Area Under the Curve (AUC) of the plot Time vs Jaccard ($\mathcal{J}$) performance. Note that $\mathcal{J}$ is a standard metric that is defined as the intersection-over-union (IoU) of the predicted object mask and its respective ground-truth mask. Each point in the AUC- $\mathcal{J}$ plot is computed considering the average time and the average Jaccard, obtained for a certain interaction, for the whole dataset. The second metric is called $\mathcal{J}@60$ which evaluates what Jaccard performance a method can reach within 60 seconds.

Network Structure As shown in Figure 6.3.1, our network is built on a pair of deep fully convolutional neural networks whose parameters are shared. Each branch of the network is a Deeplab-Resnet-101 model (pre-trained on ImageNet (Deng et al. 2009) and Microsoft-coco image dataset (Lin et al. 2014b), as in Voigtlaender and Leibe (2017) and

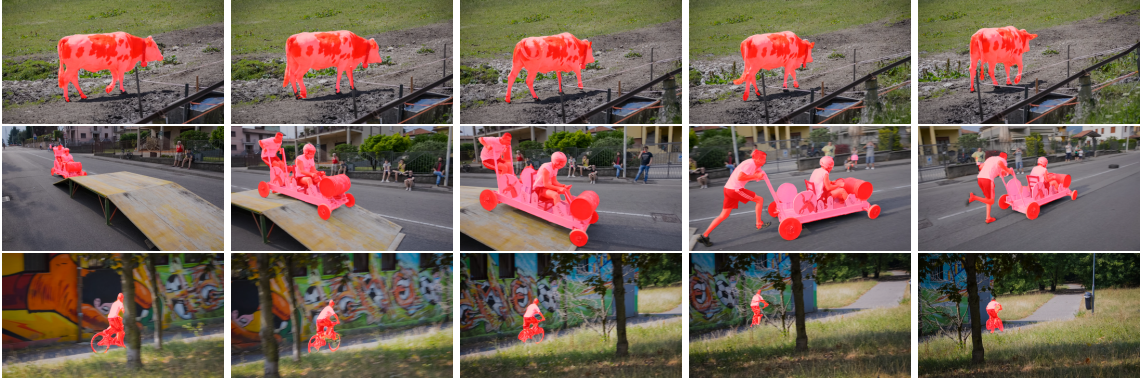


Figure 6.4.1: Qualitative segmentation outputs for some sample videos from DAVIS-2016 (each row), obtained using our model after fine-tuning.

Li et al. (2017)), connected to a decoder to recover spatial information of the embeddings and learn similarity at a higher resolution.

More specifically, the Deeplab-Resnet-101 model produces feature maps whose resolutions are eight times smaller than the resolution of input images. As a result, many details are diminished and the boundary information is not preserved properly. To address this problem, the decoder network upsamples the feature maps and yields high-resolution embeddings. The decoder consists of a transposed convolution layer (with stride = 2) and a non-parametric upsampling layer (again with stride = 2). We tried other decoder configurations (*e.g.*, upsampling with stride = 4, or two transposed convolution layers), but none of them led to better performance. Upsampling the feature maps to the size of input data was not feasible as the embeddings could not fit into our single GPU memory.

Training and Label Transfer At each training iteration, a video is randomly selected from the training pool, and then a pair of frames are drawn randomly from the video to feed the network. The target of the network was constructed by checking whether or not any two pixels in the image pair refer to the same object. As explained in Sec. 6.3.2.1, only a small fraction of the labelled pixels are used at each iteration ($k = 100$ pixel embeddings in our experiments). In the results, we show that selecting more pixels does not affect the performance, while it slows down the training process. The network is trained for 100000 iterations, with the base learning rate of 2.5×10^{-4} for the branches and a learning rate factor of two for the decoder ($\text{lr}_{\text{decoder}} = 5 \times 10^{-4}$, with a momentum of 0.9 and weight decay of 0.0005. In addition, learning rates are decreased polynomially, with a power of 0.9.

Note that during training, we have access to the dense ground-truth maps of both input images to our network. However, at the validation/test time, the second branch of the network is fed by a query image, and we utilize the learned similarity model to transfer the ground-truth labels of the training input image in the top branch (first frame of the video) to the query image (any subsequent frame).

Label transfer, as explained in Sec. 6.3.2.2, is merely a label matching operation, which does not leverage any prior information over the image pixels. Further, we are matching every frame with only the frame with scribbles to get video object segmentation, without using any temporal segmentation propagation. Hence we can further improve the output of this step by passing it through a dense CRF (Krähenbühl and Koltun 2013), where Flow information together with RGB cues, are utilized as pairwise consistency constraints.

For the interactive video object segmentation task, the interactive process included a sequence of domain adaptation (fine-tuning based on the provided set of ground-truth scribbles), label transfer, and post-processing (CRF), which were designed differently at different interactions. We tailored the following strategy based on a set of experiments on the validation set. We initially fine-tuned our pre-trained model using the first set of scribbles for 5s but extended this fine-tuning time to 20s for the second interaction and 60s for all remaining interactions. At each interaction, a new video frame together with a set of scribbles was provided by the evaluation toolkit. We augmented the frame by random horizontal flipping, scaling (0.5 to 1.5), and rotation (-30° to 30°). The augmented data (with different augmentation parameters) fed the branches of the network and fine-tuned the model using their scribbles, together with the images and scribbles supplied at previous interactions. In addition, we stopped fine-tuning when the maximum value of loss dropped below 0.04 at each interaction. Also, we skipped CRF for the first couple of interactions to speed up the evaluation. Furthermore, the provided scribbles at each interaction were dilated with a square kernel of size three, and we also denoted the image pixels that were far from all of the object scribbles, as background pixels. Note that whenever the feedback did not include any region from the background during interactions, pixels which were the least similar to any of the initial ground-truth object scribbles were selected to represent the background.

6.4.3 Results

We evaluated the proposed model (without any fine-tuning) on the DAVIS-2016 validation set, and achieved a Jaccard score of 76.5% and an F-measure of 75.4%, which outperforms

		Mean($\mathcal{J} \& \mathcal{F}$)	$\mathcal{J}$	$\mathcal{F}$
No D.A.	MSK (Perazzi et al. 2017)	-	69.9	-
	VPN (Jampani et al. 2017)	65.5	70.2	67.8
	Ours	75.4	76.5	76.0
D.A. (1 st frame)	MSK (Perazzi et al. 2017)	77.5	79.7	75.4
	OSVOS (Caelles et al. 2017)	80.6	79.8	80.2
	OnAVOS (Voigtlaender and Leibe 2017)	81.1	81.7	81.4
	Lucid (Khoreva et al. 2017b)	83.5	84.8	82.3
	Ours	83.6	84.1	83.1

		Mean($\mathcal{J} \& \mathcal{F}$)	$\mathcal{J}$	$\mathcal{F}$
OnAVOS (Voigtlaender and Leibe 2017)		56.5	53.4	59.6
Lucid(ensemble) (Khoreva et al. 2017b)		66.6	63.4	69.9
Ours		64.0	60.3	67.6

Table 6.4.1: Results on DAVIS-2016 (left) and DAVIS-2017 (right) validation Dataset. The reported numbers are in (%) units. D.A.: Domain Adaptation. Note that in the case of the DAVIS-2017 dataset, we did not have access to the results of other methods without fine-tuning, so all the results in this table are with domain adaptation. Statistical test to get confidence significance was not done using different training runs.

all other methods with no domain adaptation. Table 6.4.1 shows the result of our approach compared to the state-of-the-art. Among other methods, only Jampani et al. (2017) and Perazzi et al. (2017) have reported their results without fine-tuning on the test video. This is mainly because other methods directly target an object segmentation problem, in which their models should learn how the object of interest looks like and its parameters should be tuned accordingly given the ground truth labels of new objects. On the contrary, our approach is a pre-trained comparator that uses ground-truth labels only as a reference for labelling query pixels, and no learning is required per se. Figure 6.1.1 demonstrates qualitative results obtained using the proposed model for some example videos in the DAVIS-2016 dataset.

Our method exhibits its strength among other techniques particularly when domain adaptation is not an option due to, *e.g.*, time constraints. Nevertheless, we experimented to show the potential of our method when equipped with test domain adaptation. To this end, we fine-tuned our model on the first frame of each test video, as described in Sec. 6.3.3 for 500 iterations. As shown in Table 6.4.1, we achieved $\mathcal{J} = 84.1\%$ and $\mathcal{F} = 83.1\%$, which are on par with the state-of-the-art. Note that Khoreva et al. (2017b) uses a very sophisticated and complex data augmentation technique and also ensembles multiple models to get the best possible results. This has led to a very long fine-tuning procedure (3.5h per video). On the contrary, our model reaches the state-of-the-art by fine-tuning on each video for less than eight minutes. In addition, once fine-tuning is over, our model can process each frame at a rate of 1 frame per second, whereas in Khoreva et al. (2017b), each frame is processed in 5s. Figure 6.4.1 depicts the result of evaluating our model after domain adaptation on some videos in the DAVIS-2016 dataset.

As reported in Table 6.4.1, we evaluated our method on the Test-dev split of the DAVIS-2017 dataset and outperformed *OnAVOS* (Voigtlaender and Leibe 2017) by a good margin.

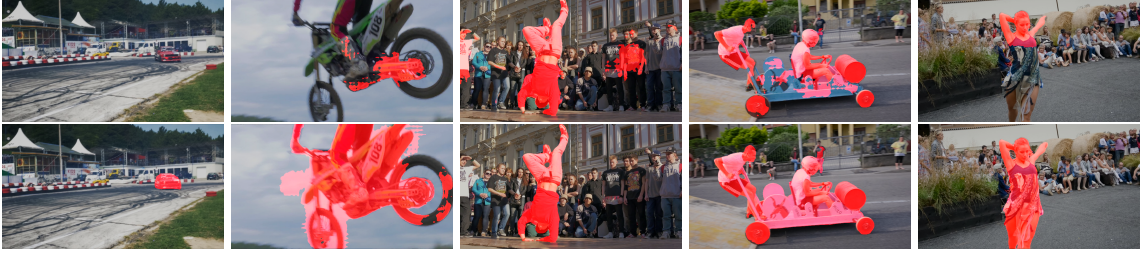


Figure 6.4.2: Some examples where domain adaptation (second row) has improved the results over the pre-trained model (first row).

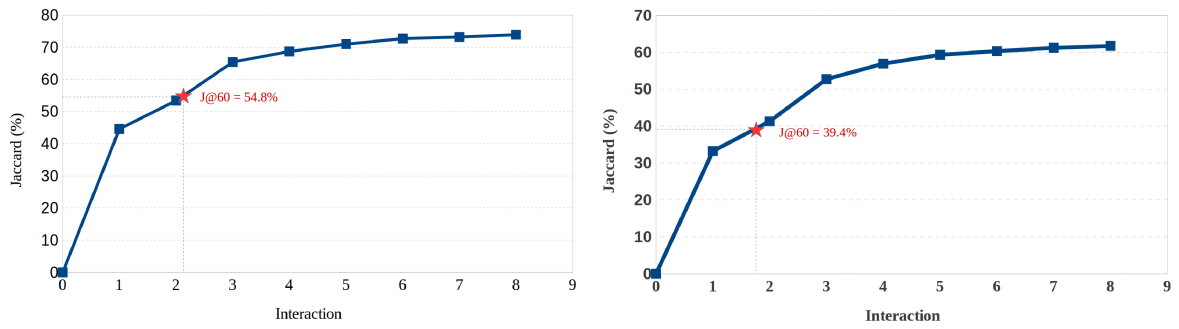


Figure 6.4.3: The interactive task performance of our method on DAVIS-2017 validation split (left) with $AUC = 70.2\%$ and $\mathcal{J}@60 = 54.8\%$, and Test-Dev split (right) with $AUC = 54.5\%$ and $\mathcal{J}@60 = 39.4\%$. Note that in the Test-Dev set, the 60s time-point occurs before the second interaction because more complex videos with a larger number of objects are present and processed in this data split.

Note that *Lucid* (Khoreva et al. 2017b) used an ensemble of four different models based on image and flow information and also leveraged an off-the-shelf state-of-the-art semantic segmentation module (PSPNet (Zhao et al. 2017)) to achieve this result. In fact, our model (with a global mean of 64%) outperforms each of the individual models used in the ensemble (63.5%, 61.1%, 61.3%, 59.9%) (Khoreva et al. 2017b).

An advantage of fine-tuning is for tackling the problem of instance segmentation, where the pre-trained model might not be able to distinguish between multiple instances of the same semantic class in a new video. Figure 6.4.2 demonstrates a few cases where our pre-trained model has failed to segment the object of interest properly from the background due to, *e.g.*, the presence of other instances of that object class in the image. However, through fine-tuning on the first frame, the model has learned to better differentiate between the main object and its surrounding.

For the interactive video object segmentation task, we achieved second place in both the quality ($AUC-\mathcal{J}$) and speed ($\mathcal{J}@60$) tracks. The charts in Figure 6.4.3 show the results



Figure 6.4.4: The impact of successive interactions on the quality of the segmentation outputs. The top row illustrates the input images and the segmentation results for consecutive interactions are depicted in the 2nd to 8th rows.

obtained using our model on the validation and test-dev splits of the DAVIS-2017 dataset. As shown in the figure, the model has consistently improved by learning from previous mistakes. Figure 6.4.4 demonstrates some examples, where the interactive process has incrementally improved the segmentation outputs.

6.5 Conclusion

In this work, we tackle one-shot video object segmentation using sparse ground-truth data, through similarity learning. In particular, we train a Siamese network, whose objective is to predict whether or not two pixels from two video frames belong to the same object class. Next, a dense label transfer approach is proposed that uses the learned similarity to propagate annotations from one video frame to the rest of the videos. Experimental results show that our method works fairly well without any domain adaptation. The proposed model can learn similarity matrix potentially from very few labelled pixels. Hence, the proposed method also achieved second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

7

Conclusion

In this integrated thesis, we worked on solving some of the important challenges in computer vision using prediction distribution with differently weighted loss functions. First, in Chapter 3 we tackled the problem of the mode-collapse problem faced in generative adversarial networks (GANs) using a simple generalization of the discriminator network’s objective to consider multiple generators. The discriminator objective used generator ‘weighted’ cross-entropy loss. To the best of our knowledge, this is the first work that could generate diverse realistic images in an image-to-image translation task using GAN architecture. Another important application of computer vision is image segmentation. While recent methods have achieved amazing results in semantic segmentation, they have mostly relied on dense per-pixel image annotations which are very expensive to obtain. In Chapter 4 we used weak supervision in the form of bounding boxes to learn semantic segmentation. This method was required to attend to segmentation in the bounding boxes and remove background noise using attention-weighted cross-entropy loss. The method could achieve better results than contemporary methods. Even though recent deep neural network architectures can get astonishing results in different tasks, they are not trustworthy to be used in critical applications. In Chapter 5 we discuss the possible reasons for this miscalibration in classifier neural networks. With the obtained insights, we claim that focal loss and its adaptations are better than standard loss functions in getting better calibration across datasets and networks. Focal loss is like cross-entropy loss where each term is weighted by the incorrectness of the prediction. Besides calibration, one more challenge facing deep learning is real-time inference, especially in the

case of neural networks working on video streams. In Chapter 6 we considered the task of one-shot video object segmentation where different methods have achieved great quality but at the cost of inference time. We proposed a method based on similarity learning that can efficiently perform dense label transfer from one image to the other. It uses per-image feature distribution to propagate the labels to other frames. It achieved the second position in the DAVIS challenge for interactive video object segmentation in terms of both quality and speed.

Next, we will summarise the contributions from each of the Chapters 3- 6 in section 7.1, brief about remaining challenges in section 7.2, and add the final concluding remarks in section 7.3.

7.1 Discussion of contributions

Chapter 3: Multi-Agent Diverse Generative Adversarial Networks

In this chapter, we worked on the mode-collapse problem faced by GANs. We started by analyzing GANs on a simple scenario of a 1D mixture of Gaussians with overlapping modes. Given enough training data, ideally, a GAN should be able to generate similar distribution as the training data but as we know, GANs suffer from mode collapse because the true equilibrium is hard to reach. It could not capture even this simple scenario. We proposed a very simple and effective framework, Multi-Agent Diverse GAN (MAD-GAN), for generating diverse and realistic samples. MAD-GAN was able to capture all the modes. Briefly, our solution MAD-GAN mainly focuses on increasing the modelling capacity of GANs by increasing the number of generators.

Our main idea is built on the intuition that if a dataset has images of artworks by three different real artists and a single painter is given the task of painting all the different kinds, as in the case of a vanilla GAN with a single generator, then the painter would be overwhelmed and not be able to do all of the styles perfectly. To produce better results, we design a model with multiple generators, where the discriminator pushes each generator to master an individual style. We modify the discriminator's objective function such that the discriminator's task now is to not only identify whether a given image is real or generated but also to identify correctly which generator produced a particular image. For this, the new objective uses generator 'weighted' cross-entropy loss. Intuitively, to excel in this task the discriminator may have to keep the modes of each generator's generation distribution different in its feature space while generators try to produce realistic images. This can help the generators capture diverse modes of data distribution.

Besides presenting a theoretical analysis of MAD-GAN with conditions for global optimality, we also showed the efficacy of our approach in capturing diverse modes while producing high-quality samples and compared it with various variants of GAN. Our formulation works even on the challenging task with the real-world application of image-to-image translation. Please note that on this task, conditional GANs are known to learn delta distribution. Still, MAD-GAN can generate diverse and realistic images. When we trained our model on a mixed dataset consisting of forests, icebergs, and indoor scenes, we found that the three natural clusterings emerged. Each generator learned to generate images from a different dataset, showing its ability to implicitly disentangle not only inter-class variations but also intra-class variations. This empirical evidence supports our theoretical claim that MAD-GAN learns a mixture model where each generator focuses on one mixture component. When our model is trained on a very diverse sketch-to-bags dataset, then given a single sketch, we could generate diverse images of bags. The generators produced bags of visually different colors and textures. Similarly, for the night-to-day task, when given a night image, MAD-GAN could produce three plausible daylight images. Each generator consistently produced visually different realistic daylight images.

Chapter 4: Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation

In this chapter, we focus on weakly-supervised semantic segmentation where we propose an approach using only bounding box annotations during training time. We treat bounding boxes as noisy labels for the foreground objects. As with all the earlier approaches, we also use GrabCut pre-processing of the bounding boxes, to recover pseudo-segmentation ground truth. One may directly train a fully-supervised segmentation model using the GrabCut outputs. However, while the bounding box is a superset of foreground pixels, GrabCut on the other hand can both over-estimate or under-estimate the foreground, so we still cannot ignore the bounding box annotations. For this specific purpose, we propose a novel loss function called attention-weighted loss. The attention-weighted loss tries to deal with label noise in bounding box annotations. We modify the standard Cross-Entropy loss function by introducing a multiplicative attention score per pixel. We predict the per-class attention map that saliently guides the per-pixel cross-entropy loss to focus on foreground pixels and refines the segmentation boundaries. This avoids propagating erroneous gradients due to incorrect foreground labels on the background. We regularize the attention map with a soft filling-rate constraint, which is obtained using the GrabCut output.

Additionally, we learn pixel embeddings to simultaneously optimize for high intra-class feature affinity while increasing discrimination between features across different classes. This can help the network capture global context via long-range pairwise interactions. The goal here is to train our network to produce features, that can help us identify which pixels are likely to take the same label. For this, we define a pairwise affinity matrix A , such that the affinity between pixels i and j is the cosine distance between their features. In this context, pixels in the image which should take the same label should have similar features. We introduce a loss function to the training objective that minimizes disagreements between the affinities produced by the discriminative features, and the actual segmentation output. By minimizing this loss, we try to harmonize the two kinds of information coming from the two sources.

Once trained, we can discard the branches producing attention maps and discriminative features, and directly use the segmentation branch's output as the prediction during the test time. We evaluated our approach against other state-of-the-art approaches for the task of weakly-supervised semantic segmentation on the PASCAL VOC 2012 dataset, and we outperform the contemporary papers in this domain, pushing the state of the art by more than 2%. Our weakly-supervised approach is also comparable to the recent fully supervised methods. The attention-weighted loss also helps in the fully supervised segmentation in the case of the PASCAL VOC 2012 dataset. We analyzed the VOC dataset and found that roughly 10% of the segmentation ground truth in VOC is missing foregrounds. However, the bounding boxes are exhaustive and cover all objects in the dataset. Because our attention-weighted loss can automatically discover attention maps, it can capitalize on missing annotations, and therefore we can produce a 1.5% improvement in the mean IoU on top of the fully-supervised baseline simply by introducing our Attention Weighted Loss. Our other qualitative results and ablation studies show the benefit of different loss terms on the overall performance of our model Box2Seg.

Chapter 5: Calibrating Deep Neural Networks using Focal Loss

A classification neural network is said to be miscalibrated when there is a mismatch between the model's confidence and its correctness. Ideally, we want networks to be accurate and calibrated so that their predictions can be trusted by a downstream task. It is much more important in critical cases of medical diagnosis and driverless cars. In this chapter, we first try to observe the possible factors of miscalibration. For example, we observe that the network gets overconfident in its predictions irrespective of the correctness as during training even

after getting full accuracy it can still reduce the loss further by increasing its confidence in the predictions. This may be caused by the magnification of weights.

We then use these insights to justify the strength of focal loss over cross-entropy loss in getting better calibration. Each of the focal loss terms is weighted by the incorrectness of the prediction, which helps it attend incorrect predictions over correct ones. We show that it implicitly maximizes entropy while minimizing the KL divergence between the predicted and the target distributions. We also show theoretically and empirically that, because of its design, it naturally regularises the weights of a network during training, reducing NLL overfitting and thereby improving calibration as compared to cross-entropy loss. We further use this insight to provide a principled approach to automatically select the hyperparameter γ involved in the focal loss to facilitate its use in practice. To do this we use a sample-dependent adaptive approach.

We conduct extensive experiments on a variety of computer vision and NLP datasets and a wide variety of network architectures. We measure calibration using several metrics like expected calibration error (ECE), adaptive ECE (AdaECE), and classwise ECE. Our results show that in almost all cases, networks trained with focal loss are more calibrated than those trained with cross-entropy loss, label smoothing, Brier score, and MMCE, without affecting the accuracy. Also, the calibration is not affected much by the temperature scaling. This makes their predictions much easier for downstream components to trust. Furthermore, we empirically observe that models trained using focal loss can also be better at detecting OoD samples which we show by taking CIFAR-10 as the in-distribution dataset and SVHN and CIFAR-10-C as out-of-distribution datasets, something which temperature scaling fails to achieve.

Chapter 6: Similarity Learning for Dense Label Transfer

In the one-shot video object segmentation task, a model has to separate an object from the background in a video given the object mask in one of the video frames. While, in the interactive video object segmentation task the difference is that at test time we are given a set of scribbles, instead of an object mask. Then, based on the model's segmentation performance, we may be provided feedback as scribbles on the worst segmentation output frame. In this chapter, we try to tackle both of these tasks using the same model and with sparse ground-truth data. Interactive video object segmentation is specifically challenging as the model has to learn dense object representation from very sparse annotation and it should

be able to consider scribbles provided as feedback on the worst output frame. Also, the model should balance good segmentation accuracy with inference time for real-time usage.

We introduce a simple and flexible method for video object segmentation based on similarity learning. The proposed method can learn to perform dense label transfer from one image to the other. More specifically, the objective is to learn a similarity metric for dense pixel-wise correspondence between two images. In particular, we train a Siamese network, whose objective is to predict whether or not two pixels from two video frames belong to the same object class. The idea is that for a given object to be segmented, most of its pixel embeddings are correlated to each other, so even a few pixels should be sufficient to learn the similarity metric, which is the approach we use as part of stochastic pooling. Also, just a few pixel embeddings of an image’s feature distribution should be enough to propagate the segmentation label to the other images.

This learned model can then be used in a label transfer framework to propagate object annotations from a reference frame to all the other frames in a video. Unlike previous methods, our similarity learning approach works fairly well across various domains, even when no domain adaptation is involved. The proposed model can learn similarity potentially from very few labelled pixels, as suggested above by the stochastic pooling approach. Hence, the proposed method could also achieve second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

7.2 Remaining challenges and future directions

While a lot of work has been done to make deep neural networks succeed in several important tasks, a lot of challenges remain that need to be overcome in the future. In this section, we discuss some of these as possible future directions.

Models for multi-tasking Humans can learn quickly with a very small amount of data as compared to millions of images needed for a deep learning model for a computer vision task. One possible reason can be that humans interact with their surroundings and get to understand them in several different ways: vision, sound, text, smell, *etc.* which may help them understand different concepts and their relationships more easily, and combine their existing skills to learn new tasks faster and more effectively. However, most deep learning models are trained in isolation on a particular dataset. It may be more efficient if they can be simultaneously trained on different tasks and different data modalities like 2D images, 3D

objects, videos, *etc.* (Dean 2021). There needs to be more work on multi-tasking and there is a possibility that it may also easily generalize to a new task (Hou et al. 2021).

Explainable machine learning Humans can generally tell whether they have seen a novel object before and if they are doubtful then they can tell that. However, it is not easy to extract similar information from current deep learning architectures. Explainable machine learning is critical to ensure any deployed machine learning model is responsible and trustworthy. There is some work on getting posthoc interpretability of networks (Selvaraju et al. 2017) but such attempts are limited in terms of the information they can extract, maybe there is a need for a complete redesign of architectures from the explainability point of view. Further, it should be possible to explain the reasoning behind a model's prediction as it becomes more important in the current socio-political environment to retain users' trust. Moreover, when a network fails, it should be possible to decide exactly what failed: the data collected, the optimizer used, *etc.* (Wong et al. 2021). Also, while a lot of research work has been done in the case of calibrating classifier neural networks, influential work in calibrating networks for other tasks is yet to be seen.

Going beyond CNNs Convolutional neural network (CNN) architectures have been successful in a lot of computer vision tasks. However, recent work like Kolesnikov et al. (2021) suggest that while the inductive bias in CNNs (Cao and Wu 2022) is helpful for smaller datasets, for larger datasets more general architectures like transformers may be more beneficial. This is possibly due to certain limitations of CNNs in learning distant spatial relations among others. Also as suggested earlier, for networks to work on different modalities of data for multi-task learning, it is more likely that we need more general architecture than CNN which was created from the vision point of view. While there has been a lot of work on automatically learning and adapting networks based on the task, *e.g.*, meta-learning (Finn et al. 2017) and neural architecture search (Tan and Le 2019), they are still computationally expensive.

3D modelling When humans look at objects with two eyes, they have some idea of depth which can be useful to imagine the 3D structure of the objects. When humans are learning object recognition, 3D objects are more helpful than 2D objects (Anderson et al. 2019) which further indicates that the human brain can utilize 3D visualization for different tasks. While there are monocular depth cues such as the relative size of objects, occlusion, shading,

and motion parallax (Cutting and Vishton 1995), still stereoscopic display seems beneficial for tasks involving judging distances, discerning relative positions, finding and identifying objects, and navigation (McIntire et al. 2014). We can extend these ideas from human vision to computer vision to go beyond monocular images. There is a need to work on 3D datasets to have better cues for scene understanding. Recently there has been an emergence of a lot of devices for capturing 3D details of the scene, however, the quantity of 3D datasets for different surroundings still lags behind their 2D counterparts.

7.3 Concluding remarks

Deep learning has had a lot of impact on several computer vision tasks, especially in the last decade thanks to the availability of large datasets and an increase in computation. This thesis has tried to understand and improve deep learning performance on different tasks, all coordinating together for better scene understanding. There are still several important challenges remaining to make computer vision systems more efficient and trustworthy and thus more common in society.

Appendices



MADGAN: Multi-Agent Diverse Generative Adversarial Networks

Here, we first give better insights about the Theorem 1 and discuss how and when MADGAN leads to diverse generations. In Appendix A.2, we introduce another way of getting different generators to generate diverse samples. We introduce an intuitive similarity-based competing objective (MAD-GAN-Sim) which encourages different generators to generate diverse samples. Finally, in Appendix A.3, we provide architecture details and data preparations for all the experiments reported for MAD-GAN and MAD-GAN-Sim.

A.1 Insights for Diversity in MAD-GAN

One obvious question that could arise is that *is it possible that all the generators learn to capture the same mode?*. The short answer is, theoretically *yes* and in practice *no*. Let us begin with the discussion to understand this. Theoretically, according to Theorem 1 if $p_{g_i} = p_d$, for all i , then also the minimum objective value can be achieved. This implies, that in the worst case, MAD-GAN would perform the same as the standard GAN. However, as discussed below, this is possible in the following *highly unlikely* situations:

- all the generators *always generate exactly similar* samples so that the discriminator is not able to differentiate them. In this case, the discriminator will learn a uniform

distribution over the generator indices, thus, the gradients passed through the discriminator will be the same for all the generators. However, this situation, in general, is not possible as all the generators are initialized differently. Even a slight variation in the samples from the generators will be enough for the discriminator to identify them and pass different gradient information to each generator. In addition, the objective function of generators is *only* to generate *real* samples, thus, there is nothing that encourages them to generate the same samples.

- the discriminator does not have enough capacity to learn the optimal parameters. This is in contrast to the assumption made in Theorem 1, which is that the discriminator is *optimal*. Thus, it should have enough capacity to learn a feature representation such that it can correctly identify samples from different generators. In practice, this is a very easy task and we did not have to modify anything up to the feature representation stage of the architecture of the discriminator. We used the standard architectures (explained in Appendix A.3) for all the tasks.

Hence, with random initializations and sufficient capacity generator/discriminator, we can easily avoid the trivial solution in which all the generators focus on the same region of the true data distribution. This has been very clearly supported by various experiments showing diverse generations by MAD-GAN.

A.2 Similarity-based competing objective

We have discussed the MAD-GAN architecture using a generator identification-based objective. In this section, we propose a different extension to the standard GAN: *similarity-based competing objective*, which we call as MAD-GAN-Sim. Here, we augment the GAN objective function with a diversity-enforcing term. It ensures that the generations from different generators are diverse where the diversity depends on a user-defined task-specific function.

The architecture is the same as MAD-GAN discussed in Section 3.4.1 (refer Fig. A.2.1).

A.2.1 Approach

The approach presented here is motivated by the fact that the samples from different modes must look different. For example, in the case of images, these samples should differ in terms of texture, colour, shading, and various other cues. Thus, different generators must generate dissimilar samples where the dissimilarity comes from a task-specific function. Before

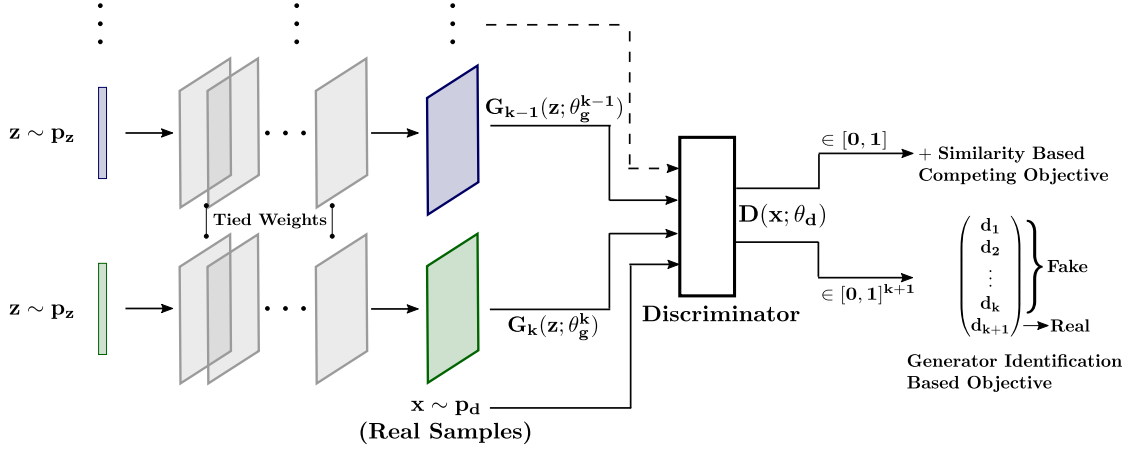


Figure A.2.1: MAD-GAN-Sim compared with MAD-GAN. All the generators share parameters of all the layers except the last one. Two proposed diversity enforcing objectives, ‘competing’ (MAD-GAN-Sim) and ‘generator identification’ (MAD-GAN), are shown at the end of the discriminator

delving into the details, let us first define some notations to avoid clutter. We denote θ_g^i as the parameters of the i -th generator. The set of generators is denoted as $K = \{1, \dots, k\}$. Given random noise z to the i -th generator, the corresponding generated sample $G_i(z; \theta_g^i)$ is denoted as $g_i(z)$. Using these notations and following the above-discussed intuitions, we impose the following constraints over the i -th generator while updating its parameters:

$$D(G_i(z; \theta_g^i); \theta_d) \geq D(G_j(z; \theta_g^j); \theta_d) + \Delta(\phi(g_i(z)), \phi(g_j(z))), \quad \forall j \in K \setminus i \quad (\text{A.1})$$

where, $\phi(g_i(z))$ denotes the mapping of the generated image $g_i(z)$ by the i -th generator into a feature space and $\Delta(.,.) \in [0, 1]$ is the similarity function. The higher the value of $\Delta(.,.)$ more similar the arguments are. Intuitively, the above set of constraints ensures that the discriminator score for each generator should be higher than all other generators with a margin proportional to the similarity score. If the samples are similar, the margin increases and the constraints become more active. We use unsupervised learning-based representation as our mapping function $\phi(.)$. Precisely, given a generated sample $g_i(z)$, $\phi(g_i(z))$ is the feature vector obtained using the discriminator of our framework. This is motivated by the feature matching-based approach to improving the stability of the training of GANs (Salimans et al. 2016). The $\Delta(.,.)$ function used in this work is the standard cosine similarity-based function. The above-mentioned constraints can be satisfied by maximizing

an equivalent unconstrained objective function as defined below:

$$U(\theta_g^i, \theta_d) := f\left(D(G_i(z; \theta_g^i); \theta_d) - \frac{1}{k-1} \sum_{j \in K \setminus i} \left(D(G_j(z; \theta_g^j); \theta_d) + \Delta(\psi_i, \psi_j)\right)\right)$$

where, $f(a) = \min(0, a)$, $\psi_i = \phi(g_i(z))$, and $\psi_j = \phi(g_j(z))$. Intuitively, if the argument of $f(\cdot)$ is positive, then the desirable constraint is satisfied and there is no need to do anything. Otherwise, maximize the argument with respect to θ_g^i . Note that instead of using all the constraints independently, we use the average of all of them. Another approach would be to use the constraint corresponding to the j -th generator that maximally violates the set of constraints shown in Eq. A.1. Experimentally we found that the training process of the average constraint-based objective is more stable than the maximum violated constraint-based objective. The intuition behind using these constraints comes from the well know 1-slack formulation of the structured SVM framework (Joachims et al. 2009; Tsochantaridis et al. 2004). Thus, the overall objective for the i -th generator is:

$$\min_{\theta_g^i} V(\theta_d, \theta_g^i) - \lambda U(\theta_g^i, \theta_d)$$

where $\lambda \geq 0$ is the hyperparameter. Algorithm 1 shows how to compute gradients corresponding to different generators for the above-mentioned objective function. Notice that, once sampled, the same z is passed through all the generators to enforce constraints over a *particular generator* (as shown in Eq. A.1). However, for constraints to not contradict each other while updating another generator, a different z is sampled again from the p_z . The Algorithm 1 is shown for the batch of size one which can be trivially generalized for any given batch size. In the case of the discriminator, the gradients will have the same form as the standard GAN objective. The only difference is that in this case the fake samples are being generated by k generators, instead of one.

A.2.2 Experiments

We present the efficacy of MAD-GAN-Sim on real-world datasets.

A.2.3 Diverse Samples for Image-to-Image Translation

We show diverse and highly appealing results using the diversity-promoting objective. We use cosine-based similarity to enforce diverse generations, an important criterion for real images.

Algorithm 1 Updating generators for MAD-GAN-Sim

input $\theta_d; p(z); \theta_g^i, \forall i \in \{1, \dots, k\}; \lambda$.

- 1: **for** each generator $i \in \{1, \dots, k\}$ **do**
- 2: Sample noise from the given noise prior $z \sim p_z$.
- 3: Obtain the generated sample $G_i(z; \theta_g^i)$ and corresponding feature vector $\psi_i = \phi(G_i(z; \theta_g^i))$.
- 4: $\nu \leftarrow 0$.
- 5: **for** each generator $j \in \{1, \dots, k\} \setminus i$ **do**
- 6: Compute feature vector $\psi_j = \phi(G_j(z; \theta_g^j))$.
- 7: $\nu \leftarrow \nu + D(G_j(z; \theta_g^j); \theta_d) + \Delta(\psi_i, \psi_j)$.
- 8: **end for**
- 9: $\nu \leftarrow D(G_i(z; \theta_g^i); \theta_d) - \frac{\nu}{k-1}$.
- 10: **if** $\nu \geq 0$ **then**
- 11: $\nabla_{\theta_g^i} \log(1 - D(G_i(z; \theta_g^i); \theta_d))$.
- 12: **else**
- 13: $\nabla_{\theta_g^i} \left(\log(1 - D(G_i(z; \theta_g^i); \theta_d)) - \lambda U(\theta_g^i, \theta_d) \right)$.
- 14: **end if**
- 15: **end for**

output

As before, we show results for the following two situations where the diverse solution is useful: (1) given the edges of handbags, generate real-looking handbags as in Fig. A.2.3; and (2) given night images of places, generate their equivalent day images as in Fig. A.2.5. We notice that each generator can produce meaningful and diverse images.

A.2.4 Unsupervised Representation Learning

We do the same experiment using SVHN as done in Section 3.5.6. For MAD-GAN-Sim we obtained a misclassification error of 18.3% which is better than DCGAN (22.48%). It indicates that MAD-GAN-Sim can learn better feature representation in an unsupervised setting.

A.3 Network Architectures and Parameters

Here we provide all the details about the architectures and the parameters used in various experiments. For the experiment concerning non-parametric density estimation, the MAD-GAN parameters are randomly initialized using Xavier initialization with normal distributed random sampling (Glorot and Bengio 2010). For all the other experiments, the initialization done is the same as the base architecture used to adapt MAD-GAN.



Figure A.2.3: MAD-GAN-Sim: Diverse generations for ‘edges-to-handbags’ image generation task. The first column in each sub-figure represents the input. The remaining three columns show the diverse outputs of different generators. It is evident that different generators are able to produce very diverse results capturing colour (brown, pink, black), texture, design pattern, and shininess, among others.

DCGAN, Unrolled GAN, In- foGAN, MA- GAN Disc	Mode- Reg DC- GAN Disc	Mode- Reg DC- GAN Enc	WGAN, GoGAN Disc	BEGAN Enc	BEGAN Dec	MAD- GAN Disc	InfoGAN QNet
Input: 1	1	1	32	1	1	1	1
fc: 128, leaky relu							
fc: 128, leaky relu							
fc: 1	1	64	1	32	1	5 (nGen+1)	5
sigmoid		identity				softmax	

Table A.3.1: Non-Parametric density estimation architecture for discriminators (Disc), encoders (Enc), decoders (Dec), and Q Network (QNet). nGen is a number of generators, fc is fully connected layer.

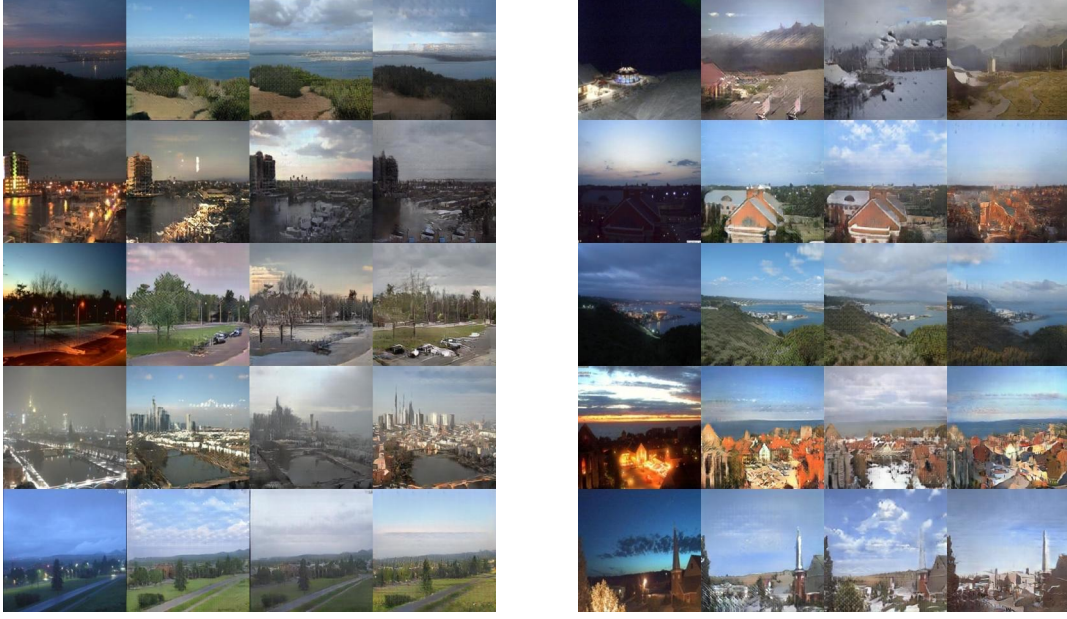


Figure A.2.5: MAD-GAN-Sim: Diverse generations for ‘night to day’ image generation task. The first column in each sub-figure represents the input. The remaining three columns show the diverse outputs of different generators. It is evident that different generators are able to produce very diverse results capturing different lighting conditions, different sky patterns (cloudy vs clear sky), different weather conditions (winter, summer, rain), and different landscapes, among many other minute yet useful cues.

A.3.1 Non-Parametric Density Estimation

Architecture Details: The generator has two fully connected hidden layers with 128 neurons (each of which is followed by an exponential linear unit) and a fully connected outer layer. In the case of MAD-GAN and MA-GAN, we used 4 generators with parameters of the first two layers shared. The generator generates 1D samples. Input to each generator is a uniform noise $U(-1, 1)$ of 64 dimension. In the case of InfoGAN, the 5 dimensional categorical code is further concatenated with the uniform noise to form the input. The categorical code is randomly sampled from the multinomial distribution. The discriminator architecture for respective networks is shown in Tab. A.3.1. Mode-Regularized GAN architecture has an encoder, BEGAN has an encoder and decoder, and InfoGAN has a Q Network whose details are also present in Tab. A.3.1.

MAD-GAN has multi-label cross-entropy loss. MA-GAN has binary cross-entropy loss. For training, we use Adam optimizer with a batch size of 128 and a learning rate of $1e - 4$. In each mini-batch, for MAD-GAN we have 128 samples from each of the generators as

	number outputs	stride
Input: $z \sim \mathcal{N}(0, I_{256})$		
Fully connected	$4 * 4 * 64$	
Reshape to image 4,4,64		
Transposed Convolution	32	2
Transposed Convolution	16	2
Transposed Convolution	8	2
Convolution	3	1

Table A.3.2: Generator architecture for 1000 class stacked-MNIST experiment. For MAD-GAN, all the layers except those mentioned in the last two rows are shared.

well as real distribution, while for MA-GAN 128 samples are chosen from real distribution as well as all the generators combined.

Dataset Generation We generated synthetic 1D data using GMM with 5 Gaussians and select their means at 10, 20, 60, 80, and 110. The standard deviation used is 3, 3, 2, 2 and 1. The first two modes overlap significantly while the fifth one is peaky and stands isolated.

A.3.2 Stacked and compositional MNIST Experiments

Architecture details: The architecture for stacked-MNIST is similar to the one used in Metz et al. (2017). Please refer to Tab. A.3.2 for generator architecture and Tab. A.3.3 for discriminator architecture and Q network architecture of InfoGAN. The architecture for the compositional-MNIST experiment is the same as DCGAN (Radford et al. 2015). Please refer to Tab. A.3.4 for discriminator architecture and Q network architecture of InfoGAN. In both experiments, the Q network of InfoGAN shares all except the last layer with the discriminator.

Dataset preparation: MNIST database of handwritten digits is used for both tasks.

A.3.3 Image-to-Image Translation

A.3.3.1 MAD-GAN/MAD-GAN-Sim

Architecture details: The network architecture is adapted from Isola et al. (2017) and the experiments were conducted with the U-Net architecture and patch-based discriminator.

	number outputs	stride
Input: 32x32 Color Image		
Convolution	4	2
Convolution	8	2
Convolution	16	2
Flatten		
Fully Connected	1	

Table A.3.3: Discriminator architecture for 1000 class stacked-MNIST experiment. For MAD-GAN, with k generators, it is adapted to have $k + 1$ dimensional last layer output. For InfoGAN, with 156 dimensional salient variables and 100-dimensional incompressible noise, it is adapted to have 156 dimensional output for the Q network.

	number outputs	stride
Input: Color Image (64x64)		
Convolution	64	2
Convolution	128	2
Convolution	256	2
Convolution	512	2
Flatten		
Fully Connected	1	

Table A.3.4: Discriminator architecture for 1000 class compositional-MNIST experiment. For MAD-GAN, with k generators, it is adapted to have $k + 1$ dimensional last layer output. For InfoGAN, with 156 dimensional salient variables and 100-dimensional incompressible noise, it is adapted to have 156 dimensional output for the Q network.

In more detail, let C_k denote a Convolution-BatchNorm-ReLU layer with k filters and CD_k represent a Convolution-BatchNorm-Dropout-ReLU layer with a dropout rate of 50%. All Convolutions are 4×4 spatial filters with a stride of 2. Convolutions in the encoder, and the discriminator, downsample by a factor of 2, whereas in the decoder they upsample by a factor of 2.

Generator Architectures We used the U-Net generator-based architecture from Isola et al. (2017) as follows:

- U-Net Encoder: C64-C128-C256-C512-C512-C512-C512
- U-Net Decoder: CD512-CD1024-CD1024-C1024-C1024-C512-C256-C128. Note

that, in the case of MAD-GAN, the last layer does not share parameters with other generators.

After the last layer in the decoder, a convolution is applied to map the number of output channels to 3, followed by a tanh function. BatchNorm is not applied to the first C64 layer in the encoder. All ReLUs in the encoder are leaky, with a slope of 0.2, while ReLUs in the decoder are not leaky. The U-Net architecture has skip connections between each layer i in the encoder and layer $n - i$ in the decoder, where n is the total number of layers. The skip connections concatenate activations from layer i to layer $n - i$. This changes the number of channels in the decoder.

Discriminator Architectures The patch-based 70×70 discriminator architecture was used in this case: C64-C128-C256-C512.

Diversity term

- MAD-GAN: After the last layer, a convolution is applied to map the output layer to the dimension of $k + 1$ (where k is the number of generators in MAD-GAN) followed by the softmax layer for the normalization.
- MAD-GAN-Sim: After the last layer, a convolution is applied to map to a 1-dimensional output followed by a Sigmoid function. For the unsupervised feature representation $\phi(\cdot)$, the feature activations from the penultimate layer C256 of the discriminator were used as the feature activations for the computation of the cosine similarity.

For the training, we used Adam optimizer with a learning rate of $2e-4$ (for both generators and discriminator), $\lambda_{L1} = 10$ (hyperparameter corresponding to the L_1 regularizer), $\lambda = 1e-3$ (corresponding to MAD-GAN-Sim), and batch size of 1.

A.3.3.2 InfoGAN

The network architecture is adapted from Isola et al. (2017) and the experiments were conducted with the U-Net architecture and patch-based discriminator.

Generator Architectures The U-Net generator is the same as in Isola et al. (2017) except that the number of input channels is increased from 3 to 4. For the experiment done for Fig. 3.5.6, to take noise as input, input channels are increased to 5 (one extra input channel for noise).

Discriminator Architectures The discriminator is exactly same as in Isola et al. (2017): C64 - C128 - C256 - C512

Q network Architectures The Q network architecture is C64 - C128 - C256 - C512 - Convolution3 - Convolution3. Here the first Convolution3 gives an output of 30×30 patches with 3 channels while the second Convolution3 just gives 3 dimensional output. All the layers except the last two are shared with the discriminator to perform the experiments for Fig. 3.5.6.

Diversity term To capture three kinds of distinct modes, the categorical code can take three values. Hence, in this case, the categorical code is a 2D matrix in which one-third of entries is set to 1 and the remaining to 0 for each category. The generator is fed the input image along with categorical code appended channel-wise to the image. For the experiment done for Fig. 3.5.6, to take noise as input, the generator input is further channel-wise appended with a 2D matrix of normal noise.

For the training, we used Adam optimizer with a learning rate of $2e - 4$ (for both generator and discriminator), $\lambda_{L1} = 10$ (hyperparameter corresponding to the L_1 regularizer) and batch size of 1.

Dataset Preparation:

- Edges-to-Handbags: We used 137,000 Amazon handbag images from Zhu et al. (2016). The random split into train and test was kept the same as done by Zhu et al. (2016).
- Night-to-Day: We used 17,823 training images extracted from 91 webcams. We thank Jun-Yan Zhu for providing the dataset.

A.3.4 Diverse-Class Data Generation

Architecture details: The network architecture is adapted from DCGAN (Radford et al. 2015). Concretely, the discriminator architecture is described in Table A.3.6 and the generator architecture in Table A.3.5. We use three generators without sharing any parameters. The residual layers helped in improving the image quality since the data manifold was much more complicated and the discriminator needed more capacity to accommodate it.

Diversity terms For the training, we used Adam optimizer with the learning rate of $2e - 4$ (both generator and discriminator) and batch size of 64.

Generator G
Input $\in \mathbb{R}^{100}$
4x4 upconv. 512 RELU.batchnorm.shared
4x4 upconv. 256 RELU. stride 2.batchnorm.shared
4x4 upconv. 128 RELU. stride 2.batchnorm.shared
4x4 upconv. 64 RELU. stride 2.batchnorm.shared
4x4 upconv. 3 tanh. stride 2

Table A.3.5: DCGAN Generator: All the layers except the last one are shared among all the three generators.

Dataset preparation: Training data is obtained by combining dataset consisting of various highly diverse images such as *islets*, *icebergs*, *broadleaf-forest*, *bamboo-forest* and *bedroom*, obtained from the Places dataset (Zhou et al. 2017a). To create the training data, images were randomly selected from each of them, creating a dataset consisting of 24,000 images.

A.3.5 Diverse Face Generations with DCGAN

Architecture details: The network architecture is adapted from DCGAN (Radford et al. 2015). Concretely, the discriminator architecture is described in Table A.3.6 and the generator architecture in Table A.3.5. In this case, all the parameters of the generators except the last layer were shared. The residual layers helped in improving the image quality since the data manifold and the manifold of each of the generators were much more complicated and the discriminator needed more capacity to accommodate it.

Diversity terms For the training, we used an Adam optimizer with a learning rate of $2e - 4$ (both generator and discriminator) and a batch size of 64.

Dataset preparation: We used the CelebA dataset as mentioned for face generation-based experiments. For Image generation, all the images (14, 197, 122) from the Imagenet-1k dataset (Deng et al. 2009) were used to train the DCGAN with 3 Generators alongside the MAD-GAN objective. The images from both CelebA and Imagenet-1k were resized into 64×64 .

Residual Discriminator D
Input 64x64 Color Image
7x7 conv. 64 leakyRELU. stride 2. pad 1. batchnorm
3x3 conv. 64 leakyRELU. stride 2. pad 1. batchnorm
3x3 conv. 128 leakyRELU. stride 2. pad 1. batchnorm
3x3 conv. 256 leakyRELU. stride 2. pad 1. batchnorm
3x3 conv. 512 leakyRELU. stride 2. pad 1. batchnorm
3x3 conv. 512 leakyRELU. stride 2. pad 1. batchnorm
3x3 conv. 512 leakyRELU. stride 2. pad 1. batchnorm
RESIDUAL-(N512, K3, S1, P1)
RESIDUAL-(N512, K3, S1, P1)
RESIDUAL-(N512, K3, S1, P1)

Table A.3.6: Discriminator architecture for diverse-class data generation and diverse face generation: The last layer output is $k + 1$ dimensional for MAD-GAN with k generators (normalizer is softmax). ‘RESIDUAL’ layer is elaborated in Table A.3.7.

RESIDUAL-Residual Layer
Input: previous-layer-output
c1: CONV-(N512, K3, S1, P1), BN, ReLU
c2: CONV-(N512, K3, S2, P1), BN
SUM(c2,previous-layer-output)

Table A.3.7: Residual layer description for Tab. A.3.6.

A.3.6 Unsupervised Representation Learning

Architecture details: Our architecture uses the one proposed in DCGAN (Radford et al. 2015). Similar to the DCGAN experiment on SVHN dataset ($32 \times 32 \times 3$) (Netzer et al. 2011), we removed the penultimate layer of the generator (second last row in Tab. A.3.5) and the first layer of discriminator (first convolution layer in Tab. A.3.8).

Classification task: We trained our model on the available SVHN dataset (Netzer et al. 2011). For feature extraction using a discriminator, we followed the same method as mentioned in the DCGAN paper (Radford et al. 2015). The features were then used for training a regularized linear L2-SVM. The ablation study is presented in Tab. A.3.9.

Dataset preparation: We used SVHN dataset (Netzer et al. 2011) consisting of 73,257 digits for the training, 26,032 digits for the testing, and 53,1131 extra training samples.

Discriminator D
Input 64x64 Color Image
4x4 conv. 64 leakyRELU. stride 2. batchnorm
4x4 conv. 128 leakyRELU. stride 2. batchnorm
4x4 conv. 256 leakyRELU. stride 2. batchnorm
4x4 conv. 512 leakyRELU. stride 2. batchnorm
4x4 conv. output leakyRELU. stride 1

Table A.3.8: DCGAN Discriminator: It is adapted to have $k + 1$ dimensional last layer output for MAD-GAN with k generators. (normalizer is softmax).

Technique	2 Gen	3 Gen	4 Gen
MAD-GAN	20.5%	18.2%	17.5%
MAD-GAN-Sim	20.2%	19.6%	18.3%

Table A.3.9: The misclassification error of MAD-GAN and MAD-GAN-Sim on SVHN while varying the number of generators.

As done in DCGAN (Radford et al. 2015), we used 1000 uniformly class distributed random samples for training, 10,000 samples from the non-extra set for validation and 1000 samples for testing.

For the training, we used Adam optimizer with a learning rate of $2e - 4$ (both generator and discriminator), $\lambda = 1e - 4$ (competing objective), and batch size of 64.

B

Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation

We provide supplementary qualitative and quantitative results for our weakly-supervised Box2Seg approach, as well as implementation details and results for three additional Grabcut-based baselines. In Sec. B.1, we show additional qualitative results demonstrating the superior performance of our Box2Seg method compared to Box and Grabcut baselines. In Sec. B.2, we provide implementation details and results for three Grabcut-based baselines, which outperform some of the previously published approaches. In Sec. B.3, we show examples of noisy bounding-box annotations from the OpenImages dataset (Kuznetsova et al. 2018).

B.1 Qualitative results

We show more qualitative results in Figure B.1.1 comparing the outputs of our *Box2Seg* model against several baselines we considered in Table 4.4.3 of the main chapter.

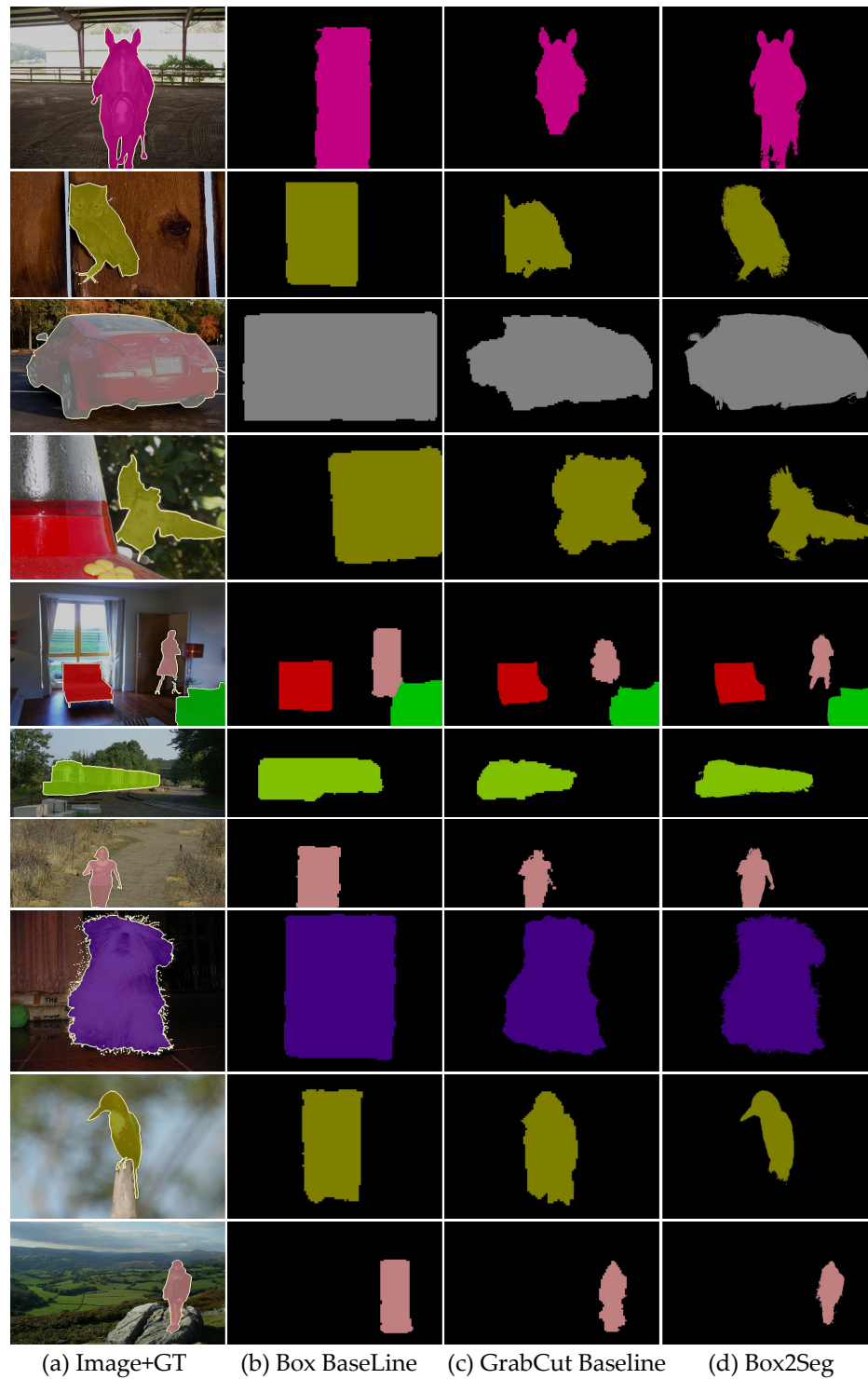


Figure B.1.1: Additional qualitative results: (a) Original Image with overlaid ground-truth (b) Results with Box baseline (c) Results with GrabCut baseline (d) Box2Seg (with CRF). Please refer to Table 4.4.3 in the main chapter for a description of the baselines.

B.2 GrabCut Baselines: Implementation Details and Quantitative Results

We use the GrabCut method (Rother et al. 2004) to generate the pseudo-ground-truth M (ref. to Sec. 4.3.1 in the main chapter for a more detailed description) for an input image I with bounding box annotations B_{box} . As described in Sec. 4.3.1 of the main chapter, $B_{box} \in \mathbb{R}^{K \times 5}$, where K is the number of bounding boxes in the image, each comprising of 4 coordinates and a class label. We use the *grabCut* function of the python library *cv2* as mentioned in the Algorithm 1 to generate the pseudo-ground-truth.

Algorithm 1: Python code to create pseudo-ground-truth using GrabCut.

Data: I, H, W, B_{box} : image, image height, width, and bounding-boxes

Result: `pseudo_ground_truth`: M

Denote by $sorted_bounding_boxes \leftarrow B_{box}$ sorted from large to small;

`pseudo_ground_truth` = $0_{H \times W}$;

for $bbox$ in $sorted_bounding_boxes$ **do**

$x, y, w, h, category_id = bbox$;

`mask` = $0_{h \times w}$;

`mask`[$y : y+h, x : x+w$] = 2;

`mask`[$y+int(h/10) : y+int(9*h/10), x+int(w/10) : x+int(9*w/10)$] = 3;

`mask`[$y+int(h/4) : y+int(3*h/4), x+int(w/4) : x+int(3*w/4)$] = 1;

`bgdModel` = $0_{1 \times 65}$;

`fgdModel` = $0_{1 \times 65}$;

`cv.grabCut(I, mask, None, bgdModel, fgdModel, 5, cv.GC_INIT_WITH_MASK)`;

`mask` = `numpy.where((mask==2)|| (mask==0), 0, 1).astype('uint8')`;

`mask`[$y, :$] = 0;

`mask`[$y+h, :$] = 0;

`mask`[$:, x$] = 0;

`mask`[$:, x+w$] = 0;

`pseudo_ground_truth`[`mask`>0] = $category_id$;

end

In Table B.2.1, we evaluate the accuracy of this GrabCut algorithm itself, without any training, against the segmentation ground-truth. Interestingly, GrabCut output on *ground-truth* bounding boxes (*GrabCut-NoTrain-GT*) results in a strong weakly-supervised baseline in itself with 71.6% mIoU. This is better compared to previous weakly supervised methods as shown in Table 4.4.1 of the main chapter. However, since the ground-truth bounding boxes are not available at inference, a more practical baseline is to obtain bounding boxes on the

Method	mIoU
GrabCut-NoTrain-GT	71.6
GrabCut-NoTrain-Det	68.5
GrabCut	72.7

Table B.2.1: Comparison of different methods using GrabCut pseudo-ground-truths obtained with our Algorithm. We consider two *evaluation-only* baselines, *GrabCut-NoTrain-GT* and *GrabCut-NoTrain-Det*, where we evaluate the accuracy of Grabcut-based Segmentation (refer to Algorithm 1) on Ground-Truth and Detected boxes respectively. We also consider a learning-based baseline, *Grabcut* which is a CNN baseline (using our ResNet-101 architecture described in Sec. 4.1 of the main chapter) trained with our loss $\mathcal{L}_{GC}$ described in Eqn. 4.2 in the main chapter.

validation set using an object detector (we used SNIPER (Singh et al. 2018)) and then run GrabCut on those. This baseline is referred to as *GrabCut-NoTrain-Det* and obtains 68.5% mIoU. Furthermore, training our segmentation network with *GrabCut* masks as supervision ($\mathcal{L}_{GC}$ loss only, refer to Eqn. 4.2 in the main chapter) without affinity or attention losses results in 72.7% mIOU. We expected the *GrabCut* baseline to be more accurate than the *NoTrain* baselines, as the *NoTrain* baselines do not harness the generalization capabilities of CNNs. We were quite surprised to find that previous SOTA papers (Khoreva et al. 2017a) neglected to compare their work with such a trivial, albeit strong baseline. Xu et al. (2017) has shown GrabCut results for comparison on the same dataset in its Figure 6 and they also seem quite competitive.

B.3 Erroneous Annotations in the OpenImages Dataset

We show examples of erroneous bounding box annotations present in the OpenImages dataset (Kuznetsova et al. 2018) in Fig. B.3.1 and B.3.2. We randomly sampled images from the dataset and picked a few examples demonstrating the three most common error modes: (i) bounding boxes that cover multiple object instances, (ii) non-exhaustive bounding boxes that miss objects of interest, and (iii) coarse bounding boxes which are not aligned with object boundaries, and either underestimate or overestimate object size. Case (i) causes inaccuracies in the outputs of the Grabcut algorithm (Rother et al. 2004) which is targeted for single object foreground extraction. Cases (ii) and (iii) invalidate our assumption that the pixels outside all bounding boxes can be considered *definite background*.

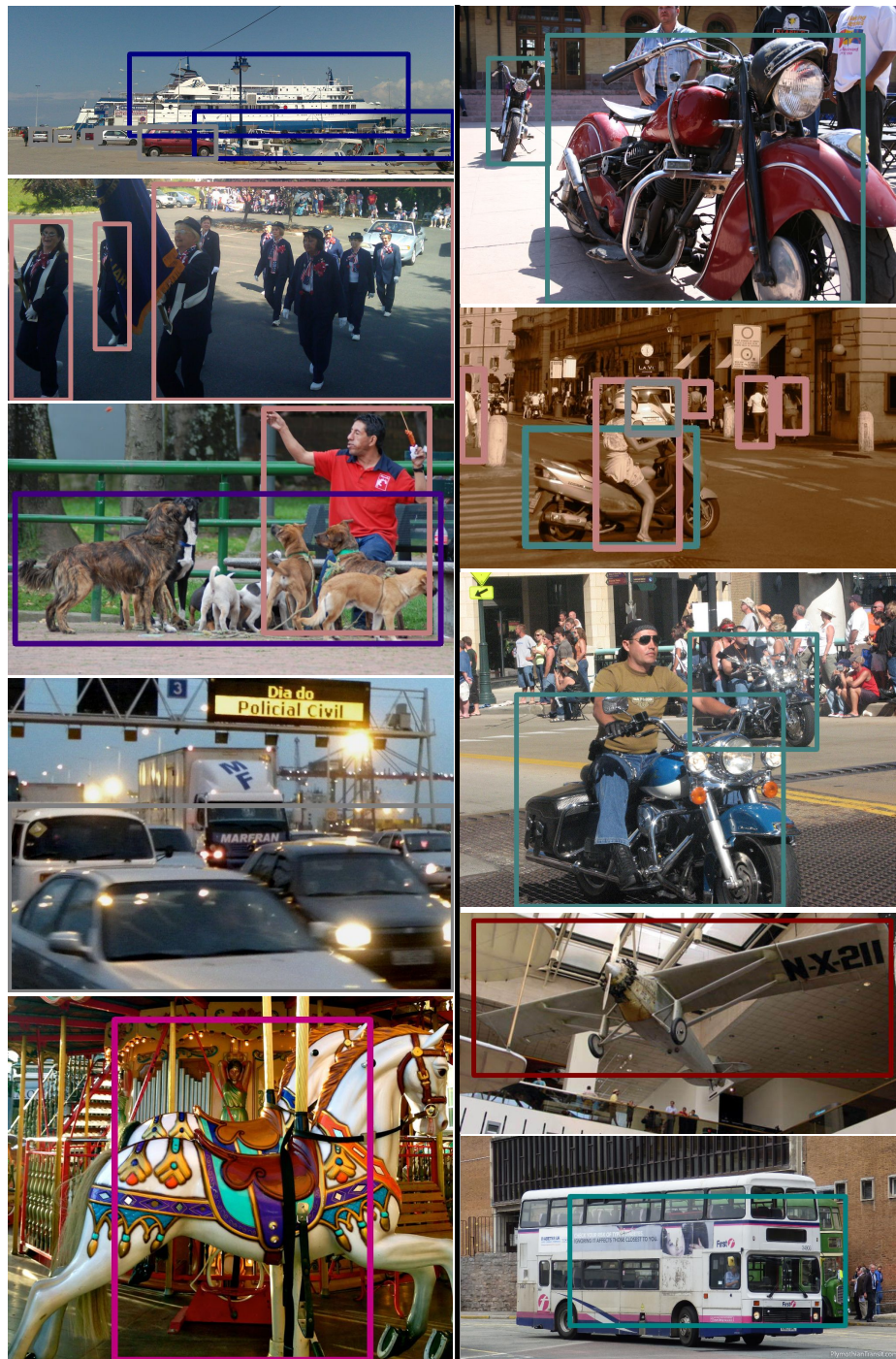


Figure B.3.1: Erroneous bounding box ground-truth in the OpenImages dataset: Col. 1 shows ground-truth bounding boxes that contain multiple instances inside. Col. 2 shows missing bounding boxes for the person class. Bounding-box annotations are not exhaustive in OpenImages.

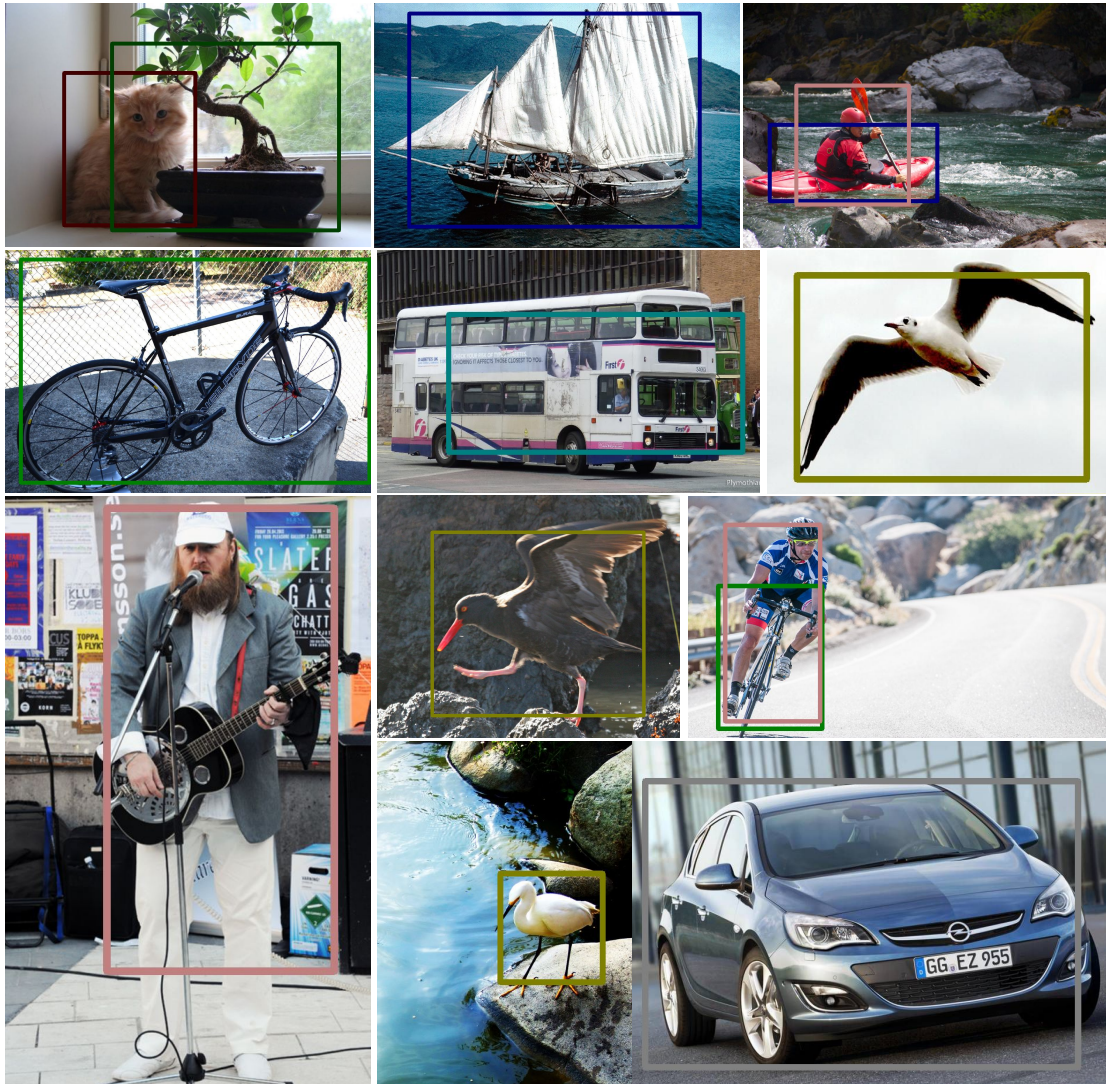


Figure B.3.2: Erroneous bounding box ground-truth in the OpenImages dataset: representative examples from the dataset where bounding boxes are not aligned with the object boundaries. Eg. the bounding box of the bicycle in row 2 col. 1 overestimates the object boundary on the upper-right corner, and the bounding box underestimates the legs of the person in row 3 col. 1.



Calibrating Deep Neural Networks using Focal Loss

In §C.1, we provide some empirical evidence for the observation made in §5.3 using reliability plots. In §C.2, we discuss the relationship between focal loss and a regularised KL divergence, where the regulariser is the entropy of the predicted distribution. In §C.3, we discuss the regularisation effect of focal loss on a simple setup, *i.e.*, a generalized linear model trained on simple data distribution. In §C.4, we show the proofs of the two propositions formulated in the main text. We then describe all the datasets and implementation details for our experiments in §C.5. In §C.6, we discuss additional approaches for training using focal loss, and also the results we get from these approaches. We also provide the Top-5 accuracies of several models as we speculate that calibrated models should have a higher softmax probability on the correct class even when they misclassify, as compared to models which are less calibrated. We further provide the results of evaluating our models using various metrics other than ECE (like AdaECE, Classwise-ECE, MCE, and NLL). Next, in §C.7, we provide additional results related to the confidence interval experiments performed in §5.5. In §C.8, we provide empirical and qualitative results to show that models trained using focal loss are calibrated, whilst maintaining their confidence in correct predictions. In §C.9, we provide a brief extension of our discussion about Figure 5.4.1(e), with a plot of L_2 norms of features obtained from the last ResNet block during training. In §C.10, we provide some empirical evidence to support the claims we make in §5.4 about early stopping. Finally, in §C.11, we

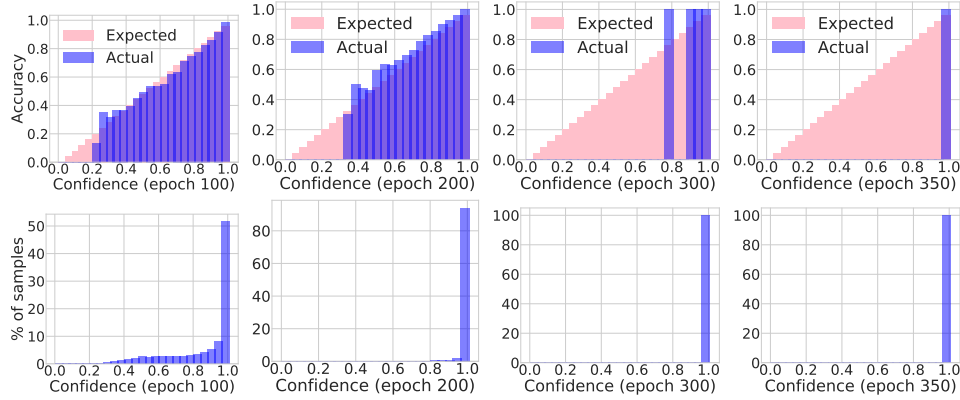


Figure C.1.1: The confidence values for training samples at different epochs during the NLL training of a ResNet-50 on CIFAR-10 (see §5.3). Top row: reliability plots using 25 confidence bins; bottom row: % of samples in each bin. As training progresses, the model gradually shifts all training samples to the highest confidence bin. Notably, it continues to do so even after achieving 100% training accuracy by the 300 epoch point.

discuss the performance of focal loss on the downstream task of machine translation with beam search. We choose machine translation as the downstream task because, in machine translation, softmax vectors from a model are directly fed into the beam search algorithm, and hence more calibrated probability vectors should intuitively produce better translations

C.1 Reliability Plots

In this section, we provide some empirical evidence to support the observation made in §5.3 that a model, even after attaining perfect training accuracy, can reduce the training NLL (loss) further by increasing the prediction confidences to match the ground-truth one-hot encoding. To empirically observe this, we use the ResNet-50 network used for the analysis in §5.3. We divide the confidence range $[0, 1]$ into 25 bins, and present reliability plots computed on the training set at training epochs 100, 200, 300 and 350 (see the top row of Figure C.1.1). In Figure C.1.1, we also show the percentage of samples in each confidence bin. It is quite clear from these plots that over time, the network gradually pushes all of the training samples towards the highest confidence bin. Furthermore, even though the network has achieved 100% accuracy on the training set by epoch 300, it still pushes some of the samples lying in lower confidence bins to the highest confidence bin by epoch 350.

C.2 Relation between Focal Loss and Entropy Regularised KL Divergence

Here we show why focal loss favours accurate but relatively less confident solutions. We show that it inherently provides a trade-off between minimizing the KL divergence and maximizing the entropy, depending on the strength of γ . We use $\mathcal{L}_f$ and $\mathcal{L}_c$ to denote the focal loss with parameter γ and cross entropy between $\hat{p}$ and q , respectively. K denotes the number of classes and q_y denotes the ground-truth probability assigned to the y^{th} class (similarly for $\hat{p}_y$). We consider the following simple extension of focal loss:

$$\begin{aligned}
 \mathcal{L}_f &= - \sum_{y=1}^K (1 - \hat{p}_y)^\gamma q_y \log \hat{p}_y \\
 &\geq - \sum_{y=1}^K (1 - \gamma \hat{p}_y) q_y \log \hat{p}_y && \text{By Bernoulli's inequality } \forall \gamma \geq 1, \text{ since } \hat{p}_y \in [0, 1] \\
 &= - \sum_{y=1}^K q_y \log \hat{p}_y - \gamma \left| \sum_{y=1}^K q_y \hat{p}_y \log \hat{p}_y \right| && \forall y, \log \hat{p}_y \leq 0 \\
 &\geq - \sum_{y=1}^K q_y \log \hat{p}_y - \gamma \max_j q_j \sum_{y=1}^K \hat{p}_y \log \hat{p}_y && \text{By Hölder's inequality } \|fg\|_1 \leq \|f\|_\infty \|g\|_1 \\
 &\geq - \sum_{y=1}^K q_y \log \hat{p}_y + \gamma \sum_{y=1}^K \hat{p}_y \log \hat{p}_y && \forall j, q_j \in [0, 1] \\
 &= \mathcal{L}_c - \gamma \mathbb{H}[\hat{p}].
 \end{aligned}$$

We know that $\mathcal{L}_c = \text{KL}(q||\hat{p}) + \mathbb{H}[q]$. Combining this equality with the above inequality leads to:

$$\mathcal{L}_f \geq \text{KL}(q||\hat{p}) + \underbrace{\mathbb{H}[q]}_{\text{constant}} - \gamma \mathbb{H}[\hat{p}].$$

In the case of one-hot encoding (Delta distribution for q), the focal loss will maximize $-\hat{p}_y \log \hat{p}_y$ (let y be the ground-truth class index), the component of the entropy of $\hat{p}$ corresponding to the ground-truth index. Thus, it will prefer learning $\hat{p}$ such that $\hat{p}_y$ is assigned a higher value (because of

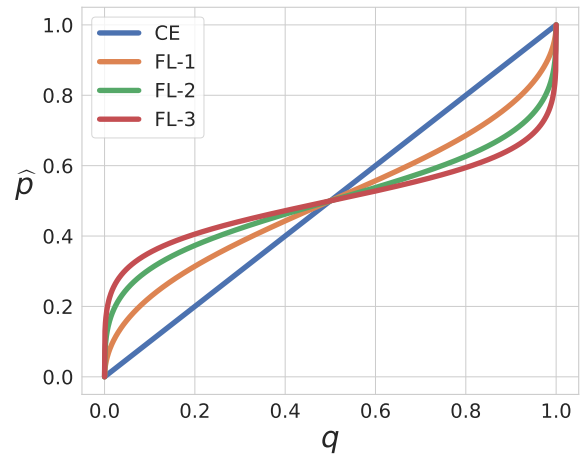


Figure C.2.1: Optimal $\hat{p}$ for various values of q .

the KL term), but not too high (because of the entropy term), and will ultimately avoid preferring overconfident models (by contrast to cross-entropy loss). Experimentally, we found the solution of the cross-entropy and focal loss equations, *i.e.*, the value of the predicted probability $\hat{p}$ which minimizes the loss, for various values of q in a binary classification problem (*i.e.*, $K = 2$), and plotted it in Figure C.2.1. As expected, focal loss favours a more entropic solution $\hat{p}$ that is closer to 0.5. In other words, as Figure C.2.1 shows, solutions to focal loss (Equation C.1) will always have higher entropy than those of cross-entropy, depending on the value of γ .

$$\hat{p} = \operatorname{argmin}_x - (1 - x)^\gamma q \log x - x^\gamma (1 - q) \log (1 - x) \quad 0 \leq x \leq 1 \quad (\text{C.1})$$

C.3 Focal Loss and Cross-Entropy on a Linear Model

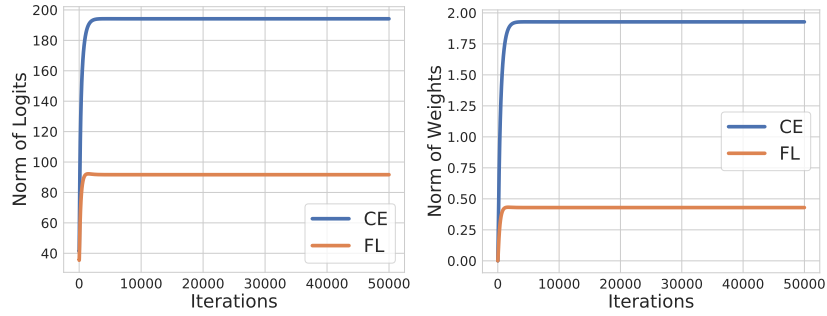


Figure C.3.1: (a): Norm of logits (b): Norm of weights.

The behaviour of deep neural networks is generally quite different from linear models and the problem of calibration is more pronounced in the case of deep neural networks, hence we focus on analyzing the calibration of deep networks in this chapter. However, weight norm analysis for the initial layers is complex due to batch norm and weight decay. Hence, to see the effect of weight magnification on miscalibration, here we use a simple network without batch norm or weight decay, which is a generalized linear model, and simple data distribution.

Setup We consider a binary classification problem. The data matrix $\mathbf{X} \in \mathbb{R}^{2 \times N}$ is created by assigning each class, two normally distributed clusters such that the mean of the clusters is linearly separable. The mean of the clusters is situated on the vertices of a two-dimensional hypercube of side length 4. The standard deviation for each cluster is 1 and the samples

are randomly linearly combined within each cluster to add covariance. Further, for 10% of the data points, the labels were flipped. 4000 samples are used for training and 1000 samples are used for testing. The model consists of a simple 2-parameter logistic regression model. $f_{\mathbf{w}}(\mathbf{x}) = \sigma(w_1x_1 + w_2x_2)$. We train this model using both cross-entropy and focal loss with $\gamma = 1$.

Weight Magnification We have argued that focal loss implicitly regularizes the weights of the model by providing smaller gradients as compared to cross-entropy. This helps in calibration as, if all the weights are large, the logits are large and thus the confidence of the network is large on all test points, even on the misclassified points. When the model misclassifies, it misclassifies with high confidence. Figure C.3.1 shows, for a generalized linear model, that the norm of the logits and the weights of a network blows for Cross-Entropy as compared to Focal Loss.

High Confidence for mistakes Figures C.3.2 (b) and (c) show that running gradient descent with cross-entropy (CE) and focal loss (FL) both gives the same decision regions, *i.e.*, the weight vector points in the same region for both FL and CE. However, as we have seen that the norm of the weights is much larger for CE as compared to FL, we would expect the confidence of misclassified test points to be large for CE as compared to FL. Figure C.3.2 (a) plots a histogram of the confidence of the misclassified points and it shows that CE misclassifies almost always with greater than 90% confidence whereas FL misclassifies with much lower confidence.

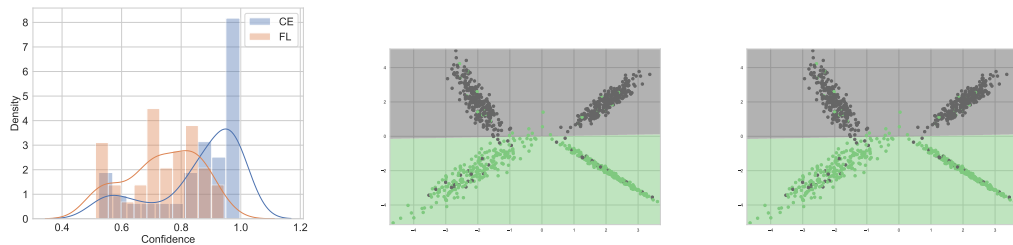


Figure C.3.2: (a): Confidence of mis-classifications (b): Decision boundary of a linear classifier trained using cross entropy (c): Decision boundary of a linear classifier trained using focal loss

C.4 Proofs

Here we provide the proofs of both propositions presented in the main text. While Proposition 1 helps us understand the regularization effect of focal loss, Proposition 2 provides us with the γ values in a principled way such that it is sample-dependent. Implementing the sample-dependent γ is very easy as the implementation of the Lambert-W function (Corless et al. 1996) is available in standard libraries (e.g., python scipy).

Proposition 1. *For focal loss $\mathcal{L}_f$ and cross-entropy $\mathcal{L}_c$, the gradients $\frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} g(\hat{p}_{i,y_i}, \gamma)$, where $g(p, \gamma) = (1-p)^\gamma - \gamma p(1-p)^{\gamma-1} \log(p)$, $\gamma \in \mathbb{R}^+$ is the focal loss hyperparameter, and $\mathbf{w}$ denotes the parameters of the last linear layer. Thus $\left\| \frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} \right\| \leq \left\| \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} \right\|$ if $g(\hat{p}_{i,y_i}, \gamma) \in [0, 1]$.*

Proof. Let $\mathbf{w}$ be the linear layer parameters connecting the feature map to the logit s . Then, using the chain rule, $\frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} = \left(\frac{\partial s}{\partial \mathbf{w}} \right) \left(\frac{\partial \hat{p}_{i,y_i}}{\partial s} \right) \left(\frac{\partial \mathcal{L}_f}{\partial \hat{p}_{i,y_i}} \right)$. Similarly, $\frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} = \left(\frac{\partial s}{\partial \mathbf{w}} \right) \left(\frac{\partial \hat{p}_{i,y_i}}{\partial s} \right) \left(\frac{\partial \mathcal{L}_c}{\partial \hat{p}_{i,y_i}} \right)$. The derivative of the focal loss with respect to $\hat{p}_{i,y_i}$, the softmax output of the network for the true class y_i , takes the form

$$\begin{aligned} \frac{\partial \mathcal{L}_f}{\partial \hat{p}_{i,y_i}} &= -\frac{1}{\hat{p}_{i,y_i}} \left((1 - \hat{p}_{i,y_i})^\gamma - \gamma \hat{p}_{i,y_i} (1 - \hat{p}_{i,y_i})^{\gamma-1} \log(\hat{p}_{i,y_i}) \right) \\ &= \frac{\partial \mathcal{L}_c}{\partial \hat{p}_{i,y_i}} g(\hat{p}_{i,y_i}, \gamma), \end{aligned}$$

in which $g(\hat{p}_{i,y_i}, \gamma) = (1 - \hat{p}_{i,y_i})^\gamma - \gamma \hat{p}_{i,y_i} (1 - \hat{p}_{i,y_i})^{\gamma-1} \log(\hat{p}_{i,y_i})$ and $\frac{\partial \mathcal{L}_c}{\partial \hat{p}_{i,y_i}} = -\frac{1}{\hat{p}_{i,y_i}}$. It is thus straightforward to verify that if $g(\hat{p}_{i,y_i}, \gamma) \in [0, 1]$, then $\left\| \frac{\partial \mathcal{L}_f}{\partial \hat{p}_{i,y_i}} \right\| \leq \left\| \frac{\partial \mathcal{L}_c}{\partial \hat{p}_{i,y_i}} \right\|$, which itself implies that $\left\| \frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} \right\| \leq \left\| \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} \right\|$. $\square$

Proposition 2. *Given a p_0 , for $1 \geq p \geq p_0 > 0$, $g(p, \gamma) \leq 1$ for all $\gamma \geq \gamma^* = \frac{a}{b} + \frac{1}{\log a} W_{-1} \left(-\frac{a^{(1-a/b)}}{b} \log a \right)$, where $a = 1 - p_0$, $b = p_0 \log p_0$, and W_{-1} is the Lambert-W function (Corless et al. 1996). Moreover, for $p \geq p_0 > 0$ and $\gamma \geq \gamma^*$, the equality $g(p, \gamma) = 1$ holds only for $p = p_0$ and $\gamma = \gamma^*$.*

Proof. We derive the value of $\gamma > 0$ for which $g(p_0, \gamma) = 1$ for a given $p_0 \in [0, 1]$. From Proposition 4.1, we already know that

$$\frac{\partial \mathcal{L}_f}{\partial \hat{p}_{i,y_i}} = \frac{\partial \mathcal{L}_c}{\partial \hat{p}_{i,y_i}} g(\hat{p}_{i,y_i}, \gamma), \quad (\text{C.2})$$

where $\mathcal{L}_f$ is focal loss, $\mathcal{L}_c$ is cross-entropy loss, $\hat{p}_{i,y_i}$ is the probability assigned by the model to the ground-truth correct class for the i^{th} sample, and

$$g(\hat{p}_{i,y_i}, \gamma) = (1 - \hat{p}_{i,y_i})^\gamma - \gamma \hat{p}_{i,y_i} (1 - \hat{p}_{i,y_i})^{\gamma-1} \log(\hat{p}_{i,y_i}). \quad (\text{C.3})$$

For $p \in [0, 1]$, if we look at the function $g(p, \gamma)$, then we can clearly see that $g(p, \gamma) \rightarrow 1$ as $p \rightarrow 0$, and that $g(p, \gamma) = 0$ when $p = 1$. To observe the behaviour of $g(p, \gamma)$ for intermediate values of p , we first take its derivative with respect to p :

$$\frac{\partial g(p, \gamma)}{\partial p} = \gamma(1-p)^{\gamma-2} \left[-2(1-p) - (1-p) \log p + (\gamma-1)p \log p \right] \quad (\text{C.4})$$

In Equation C.4, $\gamma(1-p)^{\gamma-2} > 0$ except when $p = 1$ (in which case $\gamma(1-p)^{\gamma-2} = 0$). Thus, to observe the sign of the gradient $\frac{\partial g(p, \gamma)}{\partial p}$, we focus on the term

$$-2(1-p) - (1-p) \log p + (\gamma-1)p \log p. \quad (\text{C.5})$$

Dividing Equation C.5 by $(-\log p)$, the sign remains unchanged and we get

$$k(p, \gamma) = \frac{2(1-p)}{\log p} + 1 - \gamma p. \quad (\text{C.6})$$

We can see that $k(p, \gamma) \rightarrow 1$ as $p \rightarrow 0$ and $k(p, \gamma) \rightarrow -(1+\gamma)$ as $p \rightarrow 1$ (using l'Hôpital's rule). Furthermore, $k(p, \gamma)$ is monotonically decreasing for $p \in [0, 1]$. Thus, as the gradient $\frac{\partial g(p, \gamma)}{\partial p}$ is positive initially starting from $p = 0$ and negative later till $p = 1$, we can say that $g(p, \gamma)$ first monotonically increases starting from 1 (as $p \rightarrow 0$) and then monotonically decreases down to 0 (at $p = 1$). Thus, if for some threshold $p_0 > 0$ and for some $\gamma > 0$, $g(p, \gamma) = 1$, then $\forall p > p_0$, $g(p, \gamma) < 1$. We now want to find a γ such that $\forall p \geq p_0$, $g(p, \gamma) \leq 1$. First, let $a = (1 - p_0)$ and $b = p_0 \log p_0$. Then:

$$\begin{aligned} g(p_0, \gamma) &= (1 - p_0)^\gamma - \gamma p_0 (1 - p_0)^{\gamma-1} \log p_0 \leq 1 \\ \implies (1 - p_0)^{\gamma-1} [(1 - p_0) - \gamma p_0 \log p_0] &\leq 1 \\ \implies a^{\gamma-1} (a - \gamma b) &\leq 1 \\ \implies (\gamma - 1) \log a + \log(a - \gamma b) &\leq 0 \\ \implies \left(\gamma - \frac{a}{b} \right) \log a + \log(a - \gamma b) &\leq \left(1 - \frac{a}{b} \right) \log a \\ \implies (a - \gamma b) e^{(\gamma - a/b) \log a} &\leq a^{(1 - a/b)} \\ \implies \left(\gamma - \frac{a}{b} \right) e^{(\gamma - a/b) \log a} &\leq -\frac{a^{(1 - a/b)}}{b} \\ \implies \left(\left(\gamma - \frac{a}{b} \right) \log a \right) e^{(\gamma - a/b) \log a} &\geq -\frac{a^{(1 - a/b)}}{b} \log a \end{aligned} \quad (\text{C.7})$$

where $a = (1 - p_0)$ and $b = p_0 \log p_0$. We know that the inverse of $y = xe^x$ is defined as $x = W(y)$, where W is the Lambert-W function (Corless et al. 1996). Furthermore, the r.h.s.

of the inequality in Equation C.7 is always negative, with a minimum possible value of $-1/e$ that occurs at $p_0 = 0.5$. Therefore, applying the Lambert-W function to the r.h.s. will yield two real solutions (corresponding to a principal branch denoted by W_0 and a negative branch denoted by W_{-1}). We first consider the solution corresponding to the negative branch (which is the smaller of the two solutions):

$$\begin{aligned} \left(\left(\gamma - \frac{a}{b} \right) \log a \right) &\leq W_{-1} \left(- \frac{a^{(1-a/b)}}{b} \log a \right) \\ \Rightarrow \gamma &\geq \frac{a}{b} + \frac{1}{\log a} W_{-1} \left(- \frac{a^{(1-a/b)}}{b} \log a \right) \end{aligned} \quad (\text{C.8})$$

If we consider the principal branch, the solution is

$$\gamma \leq \frac{a}{b} + \frac{1}{\log a} W_0 \left(- \frac{a^{(1-a/b)}}{b} \log a \right), \quad (\text{C.9})$$

which yields a negative value for γ that we discard. Thus Equation C.8 gives the values of γ for which if $p > p_0$, then $g(p, \gamma) < 1$. In other words, $g(p_0, \gamma) = 1$, and for any $p < p_0$, $g(p, \gamma) > 1$. $\square$

C.5 Dataset Description and Implementation Details

We use the following image and document classification datasets in our experiments:

1. **CIFAR-10** (Krizhevsky 2009): This dataset has 60,000 colour images of size 32×32 , divided equally into 10 classes. We use a train/validation/test split of 45,000/5,000/10,000 images.
2. **CIFAR-100** (Krizhevsky 2009): This dataset has 60,000 colour images of size 32×32 , divided equally into 100 classes. (Note that the images in this dataset are not the same images as in CIFAR-10.) We again use a train/validation/test split of 45,000/5,000/10,000 images.
3. **Tiny-ImageNet** (Deng et al. 2009): Tiny-ImageNet is a subset of ImageNet with 64 x 64 dimensional images, 200 classes and 500 images per class in the training set and 50 images per class in the validation set. The image dimensions of Tiny-ImageNet are twice that of CIFAR-10/100 images.

4. **20 Newsgroups** (Lang 1995): This dataset contains 20,000 news articles, categorised evenly into 20 different newsgroups based on their content. It is a popular dataset for text classification. Whilst some of the newsgroups are very related (*e.g.*, *rec.motorcycles* and *rec.autos*), others are quite unrelated (*e.g.*, *sci.space* and *misc.forsale*). We use a train/validation/test split of 15,098/900/3,999 documents.
5. **Stanford Sentiment Treebank (SST)** (Socher et al. 2013): This dataset contains movie reviews in the form of sentence parse trees, where each node is annotated by sentiment. We use the dataset version with binary labels, for which 6,920/872/1,821 documents are used as the training/validation/test split. In the training set, each node of a parse tree is annotated as positive, neutral, or negative. At test time, the evaluation is done based on the model classification at the root node, *i.e.*, considering the whole sentence, which contains only positive or negative sentiment.

All our experiments required a single 12 GB TITAN Xp GPU. For training networks on CIFAR-10 and CIFAR-100, we use SGD with a momentum of 0.9 as our optimizer, and train the networks for 350 epochs, with a learning rate of 0.1 for the first 150 epochs, 0.01 for the next 100 epochs, and 0.001 for the last 100 epochs. We use a training batch size of 128. Furthermore, we augment the training images by applying random crops and random horizontal flips. For Tiny-ImageNet, we train for 100 epochs with a learning rate of 0.1 for the first 40 epochs, 0.01 for the next 20 epochs, and 0.001 for the last 40 epochs. We use a training batch size of 64. It should be noted that for Tiny-ImageNet, we saved 50 samples per class (*i.e.*, a total of 10000 samples) from the training set as our own validation set to fine-tune the temperature parameter (hence, we trained on 90000 images) and we use the Tiny-ImageNet validation set as our test set.

For 20 Newsgroups, we train the Global Pooling Convolutional Network (Lin et al. 2014a) using the Adam optimizer, with a learning rate of 0.001, and betas 0.9 and 0.999. The code is a PyTorch adaptation of *20 Newsgroups* (2018). We used Glove word embeddings (Pennington et al. 2014) to train the network. We trained all the models for 50 epochs and used the models with the best validation accuracy.

For the SST Binary dataset, we train the Tree-LSTM (Tai et al. 2015) using the AdaGrad optimizer with a learning rate of 0.05 and a weight decay of 10^{-4} , as suggested by the authors. We used the constituency model, which considers binary parse trees of the data and trains a binary Tree-LSTM on them. The Glove word embeddings (Pennington et al. 2014) were also

tuned for best results. The code framework we used is inspired by *TreeLSTM* (2015). We trained these models for 25 epochs and used the models with the best validation accuracy.

For all our models, we use the PyTorch framework, setting any hyperparameters not explicitly mentioned to the default values used in the standard models. For MMCE, we used $\lambda = 2$ for all our experiments as we found it to perform better across all the values we tried. A calibrated model which does not generalize well to an unseen test set is not very useful. Hence, for all the experiments, we set the training parameters in a way such that we get the best test set accuracies on all datasets for each model.

C.6 Additional Results

In addition to the sample-dependent γ approach, we try the following focal loss approaches as well:

Focal Loss (Fixed γ): We trained models on focal loss with γ fixed to 1, 2 and 3. We found $\gamma = 3$ to produce the best ECE among models trained using a fixed γ . This corroborates the observation we made in § 5.4 that $\gamma = 3$ should produce better results than $\gamma = 1$ or $\gamma = 2$, as the regularising effect for $\gamma = 3$ is higher.

Focal Loss (Scheduled γ): As a simplification to the sample-dependent γ approach, we also tried using a schedule for γ during training, as we expect the value of $\hat{p}_{i,y_i}$ to increase in general for all samples over time. In particular, we report results for two different schedules: (a) Focal Loss (scheduled γ 5,3,2): $\gamma = 5$ for the first 100 epochs, $\gamma = 3$ for the next 150 epochs, and $\gamma = 2$ for the last 100 epochs, and (b) Focal Loss (scheduled γ 5,3,1): $\gamma = 5$ for the first 100 epochs, $\gamma = 3$ for the next 150 epochs, and $\gamma = 1$ for the last 100 epochs. We also tried various other schedules but found these two to produce the best results on the validation sets.

Finally, for the sample-dependent γ approach, we also found the policy: Focal Loss (sample-dependent γ 5,3,2) with $\gamma = 5$ for $\hat{p}_{i,y_i} \in [0, 0.2)$, $\gamma = 3$ for $\hat{p}_{i,y_i} \in [0.2, 0.5)$ and $\gamma = 2$ for $\hat{p}_{i,y_i} \in [0.5, 1]$ to produce competitive results.

In Tables C.6.1 and C.6.2, we present the AdaECE and Classwise-ECE scores for all the baselines discussed in Table 5.5.1.

In Table 5.5.2, we present the classification errors on the test datasets for all the major loss functions we considered. Here we also report the classification errors for the different focal loss approaches in Table C.6.6. We also report the ECE, Ada-ECE and Classwise-ECE for all the focal loss approaches in Table C.6.3, Table C.6.4 and Table C.6.5 respectively.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FL-3 (Ours)		FLSD-53 (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	17.52	3.42(2.1)	6.52	3.64(1.1)	15.32	2.38(1.8)	7.81	4.01(1.1)	5.08	2.02(1.1)	4.5	2.0(1.1)
	ResNet-110	19.05	5.86(2.3)	7.73	4.53(1.2)	19.14	4.85(2.3)	11.12	8.59(1.1)	8.64	4.14(1.2)	8.55	3.96(1.2)
	Wide-ResNet-26-10	15.33	2.89(2.2)	4.22	2.81(1.1)	13.16	4.25(1.9)	5.1	5.1(1)	2.08	2.08(1)	2.75	1.63(1.1)
	DenseNet-121	20.98	5.09(2.3)	5.04	2.56(1.1)	19.13	3.07(2.1)	12.83	8.92(1.2)	4.15	1.23(1.1)	3.55	1.24(1.1)
CIFAR-10	ResNet-50	4.33	2.14(2.5)	1.74	1.23(1.1)	4.55	2.16(2.6)	3.89	2.92(0.9)	1.95	1.83(1.1)	1.56	1.26(1.1)
	ResNet-110	4.4	1.99(2.8)	2.6	1.7(1.2)	5.06	2.52(2.8)	4.44	4.44(1)	1.62	1.44(1.1)	2.07	1.67(1.1)
	Wide-ResNet-26-10	3.23	1.69(2.2)	1.7	1.7(1)	3.29	1.6(2.2)	4.27	2.44(0.8)	1.84	1.54(0.9)	1.52	1.38(0.9)
	DenseNet-121	4.51	2.13(2.4)	2.03	2.03(1)	5.1	2.29(2.5)	4.42	3.33(0.9)	1.22	1.48(0.9)	1.42	1.42(1)
Tiny-ImageNet	ResNet-50	15.23	5.41(1.4)	4.37	4.07(0.9)	13.0	5.56(1.3)	15.28	6.29(0.7)	1.88	1.88(1)	1.42	1.42(1)
20 Newsgroups	Global Pooling CNN	17.91	2.23(3.4)	13.57	3.11(2.3)	15.21	6.47(2.2)	4.39	2.63(1.1)	8.65	3.78(1.5)	6.92	2.35(1.5)
SST Binary	Tree-LSTM	7.27	3.39(1.8)	8.12	2.84(2.5)	5.01	4.32(1.5)	5.14	4.23(1.2)	16.01	2.16(0.5)	9.15	1.92(0.7)

Table C.6.1: Adaptive ECE (%) computed for different approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FL-3 (Ours)		FLSD-53 (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	0.38	0.22(2.1)	0.22	0.20(1.1)	0.34	0.21(1.8)	0.23	0.21(1.1)	0.20	0.20(1.1)	0.20	0.20(1.1)
	ResNet-110	0.41	0.21(2.3)	0.24	0.23(1.2)	0.42	0.22(2.3)	0.26	0.22(1.1)	0.24	0.22(1.2)	0.24	0.21(1.2)
	Wide-ResNet-26-10	0.34	0.20(2.2)	0.19	0.19(1.1)	0.31	0.20(1.9)	0.21	0.21(1)	0.18	0.18(1)	0.18	0.19(1.1)
	DenseNet-121	0.45	0.23(2.3)	0.20	0.21(1.1)	0.42	0.24(2.1)	0.29	0.24(1.2)	0.20	0.20(1.1)	0.19	0.20(1.1)
CIFAR-10	ResNet-50	0.91	0.45(2.5)	0.46	0.42(1.1)	0.94	0.52(2.6)	0.71	0.51(0.9)	0.43	0.48(1.1)	0.42	0.42(1.1)
	ResNet-110	0.91	0.50(2.8)	0.59	0.50(1.2)	1.04	0.55(2.8)	0.66	0.66(1)	0.44	0.41(1.1)	0.48	0.44(1.1)
	Wide-ResNet-26-10	0.68	0.37(2.2)	0.44	0.44(1)	0.70	0.35(2.2)	0.80	0.45(0.8)	0.44	0.36(0.9)	0.41	0.31(0.9)
	DenseNet-121	0.92	0.47(2.4)	0.46	0.46(1)	1.04	0.57(2.5)	0.60	0.50(0.9)	0.43	0.41(0.9)	0.41	0.41(1)
Tiny-ImageNet	ResNet-50	0.22	0.16(1.4)	0.16	0.16(0.9)	0.21	0.16(1.3)	0.21	0.17(0.7)	0.16	0.16(1)	0.16	0.16(1)
20 Newsgroups	Global Pooling CNN	1.95	0.83(3.4)	1.56	0.82(2.3)	1.77	1.10(2.2)	0.93	0.91(1.1)	1.31	1.05(1.5)	1.40	1.19(1.5)
SST Binary	Tree-LSTM	5.81	3.76(1.8)	6.38	2.48(2.5)	3.82	2.70(1.5)	3.99	3.20(1.2)	6.35	2.81(0.5)	4.84	3.24(0.7)

Table C.6.2: Classwise-ECE (%) computed for different approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.

Finally, calibrated models should have a higher logit score (or softmax probability) on the correct class even when they misclassify, as compared to models which are less calibrated. Thus, intuitively, such models should have higher Top-5 accuracy. In Table C.6.7, we report the Top-5 accuracies for all our models on datasets where the number of classes is relatively high (*i.e.*, on CIFAR-100 with 100 classes and Tiny-ImageNet with 200 classes). We observe focal loss with sample-dependent γ to produce the highest top-5 accuracies on all models trained on CIFAR-100 and the second-best top-5 accuracy (only marginally below the highest accuracy) on Tiny-ImageNet.

In addition to ECE, Ada-ECE and Classwise-ECE, we use various other metrics to compare the proposed methods with the baselines (*i.e.*, cross-entropy, Brier loss, MMCE and Label Smoothing). We present the test NLL % before and after temperature scaling in Tables C.6.8 and C.6.9, respectively. We report the test set MCE % before and after temperature scaling in Tables C.6.10 and C.6.11, respectively.

We use the following abbreviation to report results on different varieties of Focal Loss.

FL-1 refers to Focal Loss (fixed γ 1), FL-2 refers to Focal Loss (fixed γ 2), FL-3 refers to Focal Loss (fixed γ 3), FLSc-531 refers to Focal Loss (scheduled γ 5,3,1), FLSc-532 refers to Focal Loss (scheduled γ 5,3,2), FLSD-532 refers to Focal Loss (sample-dependent γ 5,3,2) and FLSD-53 refers to Focal Loss (sample-dependent γ 5,3).

Dataset	Model	FL-1		FL-2		FL-3		FLSc-531		FLSc-532		FLSD-532		FLSD-53	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	12.86	2.3(1.5)	8.61	2.24(1.3)	5.13	1.97(1.1)	11.63	2.09(1.4)	8.47	2.13(1.3)	9.09	1.61(1.3)	4.5	2.(1.1)
	ResNet-110	15.08	4.55(1.5)	11.57	3.73(1.3)	8.64	3.95(1.2)	14.99	4.56(1.5)	11.2	3.43(1.3)	11.74	3.64(1.3)	8.56	4.12(1.2)
	Wide-ResNet-26-10	8.93	2.53(1.4)	4.64	2.93(1.2)	2.13	2.13(1)	9.36	2.48(1.4)	4.98	1.94(1.2)	4.98	2.55(1.2)	3.03	1.64(1.1)
	DenseNet-121	14.24	2.8(1.5)	7.9	2.33(1.2)	4.15	1.25(1.1)	13.05	2.08(1.5)	7.63	1.96(1.2)	8.14	2.35(1.3)	3.73	1.31(1.1)
CIFAR-10	ResNet-50	3.42	1.08(1.6)	2.36	0.91(1.2)	1.48	1.42(1.1)	4.06	1.53(1.6)	2.97	1.53(1.2)	2.52	0.88(1.3)	1.55	0.95(1.1)
	ResNet-110	3.46	1.2(1.6)	2.7	0.89(1.3)	1.55	1.02(1.1)	4.92	1.5(1.7)	3.33	1.36(1.3)	2.82	0.97(1.3)	1.87	1.07(1.1)
	Wide-ResNet-26-10	2.69	1.46(1.3)	1.42	1.03(1.1)	1.69	0.97(0.9)	2.81	0.96(1.4)	1.82	1.45(1.1)	1.31	0.87(1.1)	1.56	0.84(0.9)
	DenseNet-121	3.44	1.63(1.4)	1.93	1.04(1.1)	1.32	1.26(0.9)	4.12	1.65(1.5)	2.22	1.34(1.1)	2.45	1.31(1.2)	1.22	1.22(1)
Tiny-ImageNet	ResNet-50	7.61	3.29(1.2)	3.02	3.02(1)	1.87	1.87(1)	7.77	3.07(1.2)	3.62	2.54(1.1)	2.81	2.57(1.1)	1.76	1.76(1)
20 Newsgroups	Global Pooling CNN	15.06	2.14(2.6)	12.1	3.22(1.6)	8.67	3.51(1.5)	13.55	4.32(1.7)	12.13	2.47(1.8)	12.2	2.39(2)	6.92	2.19(1.5)
SST Binary	Tree-LSTM	6.78	3.29(1.6)	3.05	3.05(1)	16.05	1.78(0.5)	4.66	3.36(1.4)	3.91	2.64(0.9)	4.47	2.77(0.9)	9.19	1.83(0.7)

Table C.6.3: ECE (%) computed for different focal loss approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.

Dataset	Model	FL-1		FL-2		FL-3		FLSc-531		FLSc-532		FLSD-532		FLSD-53	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	12.86	2.54(1.5)	8.55	2.44(1.3)	5.08	2.02(1.1)	11.58	2.01(1.4)	8.41	2.25(1.3)	9.08	1.94(1.3)	4.39	2.33(1.1)
	ResNet-110	15.08	5.16(1.5)	11.57	4.46(1.3)	8.64	4.14(1.2)	14.98	4.97(1.5)	11.18	3.68(1.3)	11.74	4.21(1.3)	8.55	3.96(1.2)
	Wide-ResNet-26-10	8.93	2.74(1.4)	4.65	2.96(1.2)	2.08	2.08(1)	9.2	2.52(1.4)	5	2.11(1.2)	5	2.58(1.2)	2.75	1.63(1.1)
	DenseNet-121	14.24	2.71(1.5)	7.9	2.36(1.2)	4.15	1.23(1.1)	13.01	2.18(1.5)	7.61	2.04(1.2)	8.04	2.1(1.3)	3.55	1.24(1.1)
CIFAR-10	ResNet-50	3.42	1.51(1.6)	2.37	1.69(1.2)	1.95	1.83(1.1)	4.06	2.43(1.6)	2.95	2.18(1.2)	2.5	1.23(1.3)	1.56	1.26(1.1)
	ResNet-110	3.42	1.57(1.6)	2.69	1.29(1.3)	1.62	1.44(1.1)	4.91	2.61(1.7)	3.32	1.92(1.3)	2.78	1.58(1.3)	2.07	1.67(1.1)
	Wide-ResNet-26-10	2.7	1.71(1.3)	1.64	1.47(1.1)	1.84	1.54(0.9)	2.75	1.85(1.4)	2.04	1.9(1.1)	1.68	1.49(1.1)	1.52	1.38(0.9)
	DenseNet-121	3.44	1.85(1.4)	1.8	1.39(1.1)	1.22	1.48(0.9)	4.11	2.2(1.5)	2.19	1.64(1.1)	2.44	1.6(1.2)	1.42	1.42(1)
Tiny-ImageNet	ResNet-50	7.56	2.95(1.2)	3.15	3.15(1)	1.88	1.88(1)	7.7	2.9(1.2)	3.76	2.4(1.1)	2.81	2.6(1.1)	1.42	1.42(1)
20 Newsgroups	Global Pooling CNN	15.06	2.22(2.6)	12.1	3.33(1.6)	8.65	3.78(1.5)	13.55	4.58(1.7)	12.13	2.49(1.8)	12.19	2.37(2)	6.92	2.35(1.5)
SST Binary	Tree-LSTM	6.27	4.59(1.6)	3.69	3.69(1)	16.01	2.16(0.5)	4.43	3.57(1.4)	3.37	2.46(0.9)	4.42	2.96(0.9)	9.15	1.92(0.7)

Table C.6.4: AdaECE (%) computed for different focal loss approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.

Dataset	Model	FL-1		FL-2		FL-3		FLSc-531		FLSc-532		FLSD-532		FLSD-53	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	0.30	0.21(1.5)	0.24	0.22(1.3)	0.20	0.20(1.1)	0.28	0.21(1.4)	0.24	0.21(1.3)	0.25	0.21(1.3)	0.20	0.20(1.1)
	ResNet-110	0.34	0.21(1.5)	0.28	0.20(1.3)	0.24	0.22(1.2)	0.34	0.22(1.5)	0.28	0.21(1.3)	0.28	0.21(1.3)	0.24	0.21(1.2)
	Wide-ResNet-26-10	0.23	0.19(1.4)	0.18	0.21(1.2)	0.18	0.18(1)	0.24	0.19(1.4)	0.19	0.20(1.2)	0.19	0.19(1.2)	0.18	0.19(1.1)
	DenseNet-121	0.33	0.22(1.5)	0.23	0.21(1.2)	0.20	0.20(1.1)	0.31	0.22(1.5)	0.23	0.20(1.2)	0.23	0.22(1.3)	0.19	0.20(1.1)
CIFAR-10	ResNet-50	0.73	0.40(1.6)	0.54	0.45(1.2)	0.43	0.48(1.1)	0.85	0.51(1.6)	0.64	0.48(1.2)	0.56	0.42(1.3)	0.42	0.42(1.1)
	ResNet-110	0.74	0.44(1.6)	0.61	0.42(1.3)	0.44	0.41(1.1)	1.02	0.62(1.7)	0.73	0.51(1.3)	0.63	0.44(1.3)	0.48	0.44(1.1)
	Wide-ResNet-26-10	0.59	0.39(1.3)	0.37	0.36(1.1)	0.44	0.36(0.9)	0.61	0.37(1.4)	0.42	0.42(1.1)	0.37	0.33(1.1)	0.41	0.31(0.9)
	DenseNet-121	0.74	0.44(1.4)	0.45	0.39(1.1)	0.43	0.41(0.9)	0.88	0.47(1.5)	0.52	0.43(1.1)	0.56	0.47(1.2)	0.41	0.41(1)
Tiny-ImageNet	ResNet-50	0.18	0.17(1.2)	0.16	0.16(1)	0.16	0.16(1)	0.18	0.17(1.2)	0.16	0.16(1.1)	0.16	0.17(1.1)	0.16	0.16(1)
20 Newsgroups	Global Pooling CNN	1.73	0.98(2.6)	1.51	1.00(1.6)	1.31	1.05(1.5)	1.59	1.15(1.7)	1.54	1.10(1.8)	1.50	0.99(2)	1.40	1.19(1.5)
SST Binary	Tree-LSTM	5.46	3.63(1.6)	3.34	3.34(1)	6.35	2.81(0.5)	4.32	3.60(1.4)	3.95	3.90(0.9)	3.52	3.39(0.9)	4.84	3.24(0.7)

Table C.6.5: Classwise-ECE (%) computed for different focal loss approaches both pre and post-temperature scaling (cross-validating T on ECE). The optimal temperature for each method is indicated in brackets.

Dataset	Model	FL-1	FL-2	FL-3	FLSc-531	FLSc-532	FLSD-532	FLSD-53
CIFAR-100	ResNet-50	22.8	23.15	22.75	23.49	23.24	23.55	23.22
	ResNet-110	22.36	22.53	22.92	22.81	22.96	22.93	22.51
	Wide-ResNet-26-10	19.61	20.01	19.69	20.13	20.13	19.71	20.11
	DenseNet-121	23.82	23.19	23.25	23.69	23.72	22.41	22.67
CIFAR-10	ResNet-50	4.93	4.98	5.25	5.66	5.63	5.24	4.98
	ResNet-110	4.78	5.06	5.08	6.13	5.71	5.19	5.42
	Wide-ResNet-26-10	4.27	4.27	4.13	4.11	4.46	4.14	4.01
	DenseNet-121	5.09	4.84	5.33	5.46	5.65	5.46	5.46
Tiny-ImageNet	ResNet-50	50.06	47.7	49.69	50.49	49.83	48.95	49.06
20 Newsgroups	Global Pooling CNN	26.13	28.23	29.26	29.16	28.16	27.26	27.98
SST Binary	Tree-LSTM	12.63	12.3	12.19	12.36	13.07	12.3	12.8

Table C.6.6: Error (%) computed for different focal loss approaches.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FLSD-53 (Ours)	
		Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
CIFAR-100	ResNet-50	76.7	93.77	76.61	93.24	76.8	93.69	76.57	92.86	76.78	94.44
	ResNet-110	77.27	93.79	74.9	92.44	76.93	93.78	76.57	92.27	77.49	94.78
	Wide-ResNet-26-10	79.3	93.96	79.41	94.56	79.27	94.11	78.81	93.18	79.89	95.2
	DenseNet-121	75.48	91.33	76.25	92.76	76	91.96	75.95	89.51	77.33	94.49
Tiny-ImageNet	ResNet-50	50.19	74.24	46.8	70.34	48.69	73.52	52.88	76.15	50.94	76.07

Table C.6.7: Top-1 and Top-5 accuracies computed for different approaches.

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	FL-1	FL-2	FL-3	FLSc-531	FLSc-532	FLSD-532	FLSD-53
CIFAR-100	ResNet-50	153.67	99.63	125.28	121.02	105.61	92.82	87.52	100.09	92.66	94.1	88.03
	ResNet-110	179.21	110.72	180.54	133.11	114.18	96.74	90.9	112.46	95.85	97.97	89.92
	Wide-ResNet-26-10	140.1	84.62	119.58	108.06	87.56	77.8	74.66	88.61	78.52	78.86	76.92
	DenseNet-121	205.61	98.31	166.65	142.04	115.5	93.11	87.13	107.91	93.12	91.14	85.47
CIFAR-10	ResNet-50	41.21	18.67	44.83	27.68	22.67	18.6	18.43	25.32	20.5	18.69	17.55
	ResNet-110	47.51	20.44	55.71	29.88	22.54	19.19	17.8	32.77	22.48	19.39	18.54
	Wide-ResNet-26-10	26.75	15.85	28.47	21.71	17.66	14.96	15.2	18.5	15.57	14.78	14.55
	DenseNet-121	42.93	19.11	52.14	28.7	22.5	17.56	18.02	27.41	19.5	20.14	18.39
Tiny-ImageNet	ResNet-50	232.85	240.32	234.29	235.04	219.07	202.92	207.2	217.52	211.42	204.71	204.97
20 Newsgroups	Global Pooling CNN	176.57	130.41	158.7	90.95	140.4	115.97	109.62	128.75	123.72	124.03	109.17
SST Binary	Tree-LSTM	50.2	54.96	37.28	44.34	53.9	47.72	50.29	50.25	53.13	45.08	49.23

Table C.6.8: NLL (%) computed for different approaches pre temperature scaling.

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	FL-1	FL-2	FL-3	FLSc-531	FLSc-532	FLSD-532	FLSD-53
CIFAR-100	ResNet-50	106.83	99.57	101.92	120.19	94.58	91.80	87.37	92.77	91.58	92.83	88.27
	ResNet-110	104.63	111.81	106.73	129.76	94.65	91.24	89.92	93.73	91.30	92.29	88.93
	Wide-ResNet-26-10	97.10	85.77	95.92	108.06	83.68	80.44	74.66	84.11	80.01	80.40	78.14
	DenseNet-121	119.23	98.74	113.24	136.28	100.81	91.35	87.55	98.16	91.55	90.57	86.06
CIFAR-10	ResNet-50	20.38	18.36	21.58	27.69	17.56	17.67	18.34	19.93	19.25	17.28	17.37
	ResNet-110	21.52	19.60	24.61	29.88	17.32	17.53	17.62	23.79	20.21	17.78	18.24
	Wide-ResNet-26-10	15.33	15.85	16.16	21.19	15.48	14.85	15.06	15.81	15.38	14.69	14.23
	DenseNet-121	21.77	19.11	24.88	28.95	18.71	17.21	18.10	21.65	19.04	19.27	18.39
Tiny-ImageNet	ResNet-50	220.98	238.98	226.29	214.95	217.51	202.92	207.20	215.37	211.57	205.42	204.97
20 Newsgroups	Global Pooling CNN	87.95	93.11	99.74	90.42	87.24	93.60	94.69	97.89	93.66	91.73	93.98
SST Binary	Tree-LSTM	41.05	38.27	36.37	43.45	45.67	47.72	45.96	45.82	54.52	45.36	49.69

Table C.6.9: NLL (%) computed for different approaches post temperature scaling (cross-validating T on ECE).

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	FL-1	FL-2	FL-3	FLSc-531	FLSc-532	FLSD-532	FLSD-53
CIFAR-100	ResNet-50	44.34	36.75	39.53	26.11	33.22	21.03	13.02	26.76	23.56	22.4	16.12
	ResNet-110	55.92	24.85	50.69	36.23	40.49	32.57	26	37.24	29.56	34.73	22.57
	Wide-ResNet-26-10	49.36	14.68	40.13	23.79	27	15.14	9.96	27.81	17.59	13.64	10.17
	DenseNet-121	56.28	15.47	49.97	43.59	35.45	21.7	11.61	38.68	18.91	21.34	9.68
CIFAR-10	ResNet-50	38.65	31.54	60.06	35.61	31.75	25	21.83	30.54	23.57	25.45	14.89
	ResNet-110	44.25	25.18	67.52	45.72	73.35	25.92	25.15	34.18	30.38	30.8	18.95
	Wide-ResNet-26-10	48.17	77.15	36.82	24.89	29.17	30.17	23.86	37.57	30.65	18.51	74.07
	DenseNet-121	45.19	19.39	43.92	45.5	38.03	29.59	77.08	33.5	16.47	17.85	13.36
Tiny-ImageNet	ResNet-50	30.83	8.41	26.48	25.48	20.7	8.47	6.11	16.03	9.28	8.97	3.76
20 Newsgroups	Global Pooling CNN	36.91	31.35	34.72	8.93	34.28	24.1	18.85	26.02	25.02	24.29	17.44
SST Binary	Tree-LSTM	71.08	92.62	68.43	39.39	95.48	86.21	22.32	76.28	86.93	80.85	73.7

Table C.6.10: MCE (%) computed for different approaches pre temperature scaling.

C.7 Bar plots

In this section, we present additional results in reference to Figure 5.5.1. In particular, we compute 90% confidence intervals for ECE, AdaECE and Classwise-ECE using 1000

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	FL-1	FL-2	FL-3	FLSc-531	FLSc-532	FLSD-532	FLSD-53
CIFAR-100	ResNet-50	12.75	21.61	11.99	18.58	8.92	8.86	6.76	7.46	6.76	5.24	27.18
	ResNet-110	22.65	13.56	19.23	30.46	20.13	12	13.06	18.28	13.72	15.89	10.94
	Wide-ResNet-26-10	14.18	13.42	16.5	23.79	10.28	18.32	9.96	13.18	11.01	12.5	9.73
	DenseNet-121	21.63	8.55	13.02	29.95	10.49	11.63	6.17	6.21	6.48	9.41	5.68
CIFAR-10	ResNet-50	20.6	22.46	23.6	40.51	25.86	28.17	15.76	22.05	23.85	24.76	26.37
	ResNet-110	29.98	22.73	31.87	45.72	29.74	23.82	37.61	26.25	25.94	11.59	17.35
	Wide-ResNet-26-10	26.63	77.15	32.33	37.53	74.58	29.58	25.64	28.63	20.23	19.68	36.56
	DenseNet-121	32.52	19.39	27.03	53.57	19.68	22.71	76.27	21.05	32.76	35.06	13.36
Tiny-ImageNet	ResNet-50	13.33	12.82	12.52	17.2	6.5	8.47	6.11	5.97	7.01	5.73	3.76
20 Newsgroups	Global Pooling CNN	36.91	31.35	34.72	8.93	34.28	24.1	18.85	26.02	25.02	24.29	17.44
SST Binary	Tree-LSTM	88.48	91.86	32.92	35.72	87.77	86.21	74.52	54.27	88.85	82.42	76.71

Table C.6.11: MCE (%) computed for different approaches post temperature scaling (cross-validating T on ECE).

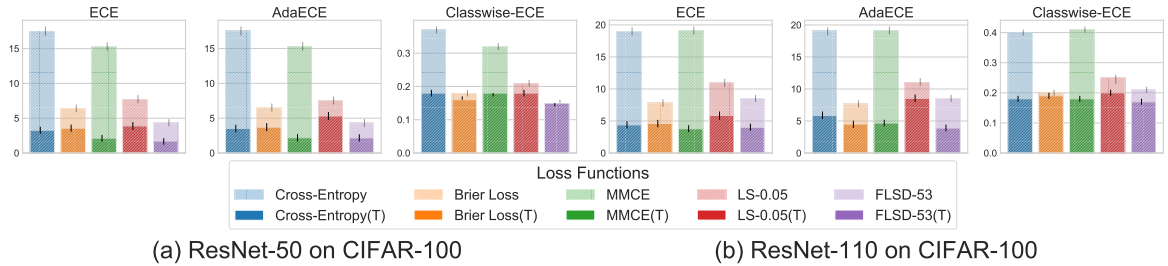


Figure C.7.1: Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for ResNet-50 (first 3 figures) and ResNet-110 (last 3 figures) on CIFAR-100.

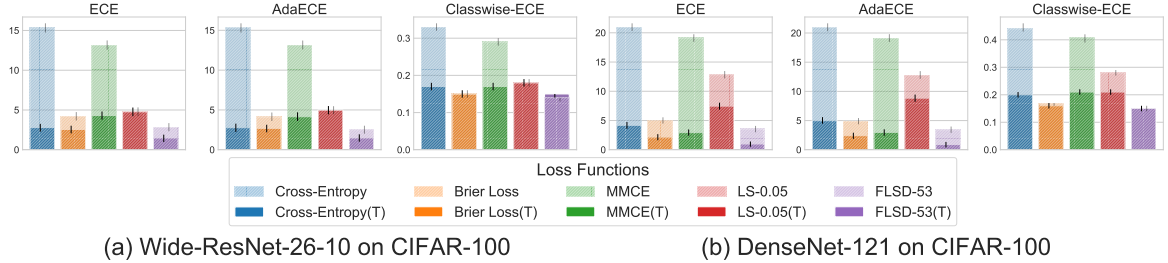


Figure C.7.2: Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for Wide-ResNet-26-10 (first 3 figures) and DenseNet-121 (last 3 figures) on CIFAR-100.

bootstrap samples following Kumar et al. (2019) and present the resulting confidence intervals as bar plots in Figures C.7.1, C.7.2, C.7.3 and C.7.4. These plots further corroborate the observations made in Section 5.5. We find that FLSD-53 broadly produces the lowest calibration errors, and in quite a few cases (especially before temperature scaling) the differences in calibration errors between cross-entropy and focal loss are statistically significant.

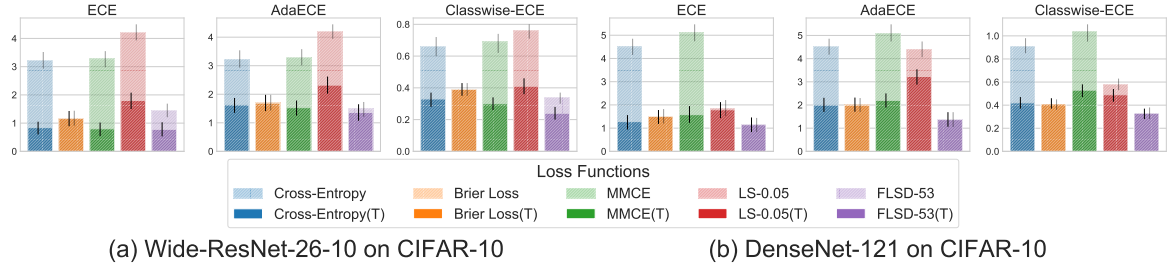


Figure C.7.3: Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for Wide-ResNet-26-10 (first 3 figures) and DenseNet-121 (last 3 figures) on CIFAR-10.

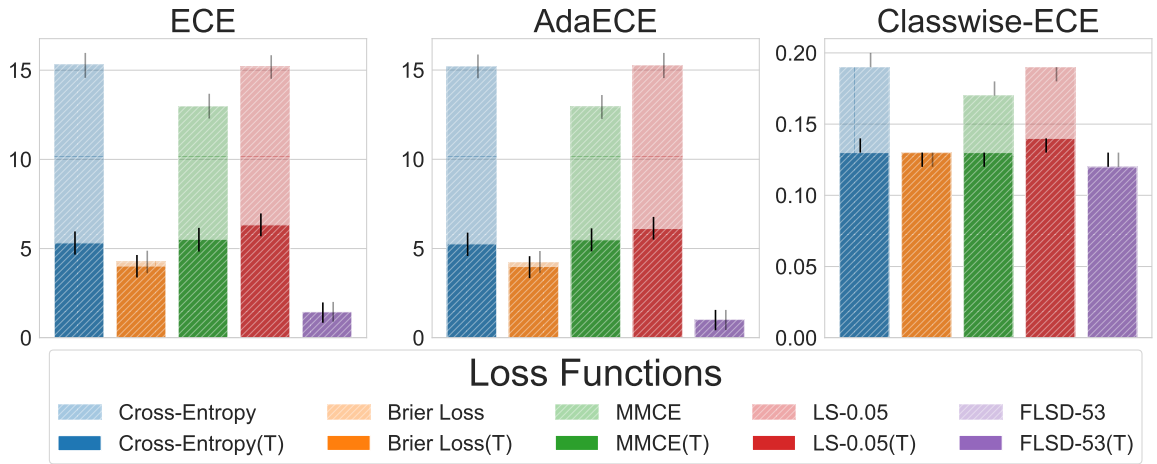


Figure C.7.4: Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE computed for ResNet-50 on Tiny-ImageNet.

Dataset	Model	Cross-Entropy (Pre T)		Cross-Entropy (Post T)		MMCE (Pre T)		MMCE (Post T)		Focal Loss (Pre T)		Focal Loss (Post T)	
		IS99%	Accuracy	IS99%	Accuracy	IS99%	Accuracy	IS99%	Accuracy	IS99%	Accuracy	IS99%	Accuracy
CIFAR-10	ResNet-110	97.11	96.33	11.5	97.39	97.65	96.72	10.62	99.83	61.41	99.51	31.10	99.68
CIFAR-10	ResNet-50	95.93	96.72	7.33	99.73	92.33	98.24	4.21	100	46.31	99.57	14.27	99.93

Table C.8.1: Percentage of test samples predicted with confidence higher than 99% and the corresponding accuracy for Cross Entropy, MMCE and Focal loss computed both pre and post-temperature scaling (represented in the table as pre-T and post-T respectively).

C.8 Focal Loss is Confident and Calibrated

In extension to what we present in § 5.5, we also follow the approach adopted in Kumar et al. (2018) and measure the percentage of test samples that are predicted with a confidence of 0.99 or more (we call this set of test samples S_{99}). In Table C.8.1, we report $|S_{99}|$ as a percentage of the total number of test samples, along with the accuracy of the samples in S_{99} for ResNet-50 and ResNet-110 trained on CIFAR-10, using cross-entropy loss, MMCE

loss, and focal loss. We observe that $|S_{99}|$ for the focal loss model is much lower than for the cross-entropy or MMCE models before temperature scaling. However, after temperature scaling, $|S_{99}|$ for focal loss is significantly higher than for both MMCE and cross-entropy. The reason is that with an optimal temperature of 1.1, the confidence of the temperature-scaled model for focal loss does not reduce as much as those of the models for cross-entropy

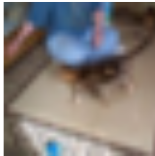
				
Cross Entropy	Actual Label: Cat Predicted Label: Cat Confidence (Pre T): 0.9999 Confidence (Post T): 0.9880	Actual Label: Ship Predicted Label: Ship Confidence (Pre T): 0.9999 Confidence (Post T): 0.9919	Actual Label: Airplane Predicted Label: Airplane Confidence (Pre T): 0.9999 Confidence (Post T): 0.9893	Actual Label: Frog Predicted Label: Frog Confidence (Pre T): 0.9999 Confidence (Post T): 0.9863
Brier Score	Actual Label: Cat Predicted Label: Cat Confidence (Pre T): 0.9977 Confidence (Post T): 0.9953	Actual Label: Ship Predicted Label: Ship Confidence (Pre T): 0.9989 Confidence (Post T): 0.9976	Actual Label: Airplane Predicted Label: Airplane Confidence (Pre T): 0.9961 Confidence (Post T): 0.9923	Actual Label: Frog Predicted Label: Frog Confidence (Pre T): 0.9955 Confidence (Post T): 0.9911
MMCE	Actual Label: Cat Predicted Label: Cat Confidence (Pre T): 0.9999 Confidence (Post T): 0.9796	Actual Label: Ship Predicted Label: Ship Confidence (Pre T): 0.9999 Confidence (Post T): 0.9894	Actual Label: Airplane Predicted Label: Airplane Confidence (Pre T): 0.9999 Confidence (Post T): 0.9694	Actual Label: Frog Predicted Label: Frog Confidence (Pre T): 0.9999 Confidence (Post T): 0.9732
Focal Loss	Actual Label: Cat Predicted Label: Cat Confidence (Pre T): 0.9984 Confidence (Post T): 0.9967	Actual Label: Ship Predicted Label: Ship Confidence (Pre T): 0.9977 Confidence (Post T): 0.9956	Actual Label: Airplane Predicted Label: Airplane Confidence (Pre T): 0.9937 Confidence (Post T): 0.9884	Actual Label: Frog Predicted Label: Frog Confidence (Pre T): 0.9954 Confidence (Post T): 0.9912
				
Cross Entropy	Actual Label: Dog Predicted Label: Bird Confidence (Pre T): 0.9872 Confidence (Post T): 0.7293	Actual Label: Airplane Predicted Label: Ship Confidence (Pre T): 0.9973 Confidence (Post T): 0.8703	Actual Label: Automobile Predicted Label: Truck Confidence (Pre T): 0.9999 Confidence (Post T): 0.9611	Actual Label: Cat Predicted Label: Airplane Confidence (Pre T): 0.9966 Confidence (Post T): 0.8019
Brier Score	Actual Label: Dog Predicted Label: Cat Confidence (Pre T): 0.7758 Confidence (Post T): 0.7396	Actual Label: Airplane Predicted Label: Ship Confidence (Pre T): 0.9750 Confidence (Post T): 0.9614	Actual Label: Automobile Predicted Label: Truck Confidence (Pre T): 0.9519 Confidence (Post T): 0.9338	Actual Label: Cat Predicted Label: Airplane Confidence (Pre T): 0.3753 Confidence (Post T): 0.3572
MMCE	Actual Label: Dog Predicted Label: Cat Confidence (Pre T): 0.9409 Confidence (Post T): 0.6406	Actual Label: Airplane Predicted Label: Ship Confidence (Pre T): 0.9999 Confidence (Post T): 0.9756	Actual Label: Automobile Predicted Label: Truck Confidence (Pre T): 0.9995 Confidence (Post T): 0.9124	Actual Label: Cat Predicted Label: Airplane Confidence (Pre T): 0.9936 Confidence (Post T): 0.7074
Focal Loss	Actual Label: Dog Predicted Label: Bird Confidence (Pre T): 0.6714 Confidence (Post T): 0.6468	Actual Label: Airplane Predicted Label: Ship Confidence (Pre T): 0.7975 Confidence (Post T): 0.7740	Actual Label: Automobile Predicted Label: Truck Confidence (Pre T): 0.5789 Confidence (Post T): 0.5689	Actual Label: Cat Predicted Label: Airplane Confidence (Pre T): 0.4178 Confidence (Post T): 0.3937

Figure C.8.1: Qualitative results showing the performance of Cross-Entropy, Brier loss, MMCE, and Focal Loss (sample-dependent γ 5,3) for a ResNet-50 trained on CIFAR-10. The first row of images has been correctly classified by networks trained on all of the four loss functions and the second row of images has all been incorrectly classified. For each image, we present the actual label, the predicted label, and the confidence of the prediction both before and after temperature scaling.

and MMCE, for which the optimal temperatures lie between 2.5 to 2.8. We thus conclude that models trained on focal loss are not only more calibrated, but also better preserve their confidence in predictions, even after being post-processed with temperature scaling.

In Figure C.8.1, we present some qualitative results to support this claim and show the improvement in the confidence estimates of focal loss in comparison to other baselines (*i.e.*, cross-entropy, MMCE, and Brier loss). For this, we take ResNet-50 networks trained on CIFAR-10 using the four loss functions (cross-entropy, MMCE, Brier loss, and Focal loss with sample-dependent γ 5,3) and measure the confidence of their predictions for four correctly and four incorrectly classified test samples. We report these confidences both before and after temperature scaling. It is clear from Figure C.8.1 that for all the correctly classified samples, the model trained using focal loss has very confident predictions in both pre and post-temperature scaling. However, on misclassified samples, we observe very low confidence for the focal loss model. The ResNet-50 network trained using cross-entropy is very confident even on the misclassified samples, particularly before temperature scaling. Apart from focal loss, the only model which has relatively low confidence in misclassified test samples is the one trained using Brier loss. These observations support our claim that focal loss produces not only a calibrated model but also one which is confident in its correct predictions.

C.9 Ordering of Feature Norms

As an extension to the discussion related to Figure 5.4.1(e), we plot the L_2 norm of the features/activations obtained from the last ResNet block (right before the linear layer is applied to these features to get the logits). We plot these norms throughout the training period for networks trained on cross-entropy and focal loss with γ set to 1, 2 and 3 in Figure C.9.1. We observe that there is a distinct ordering of feature norms for the four models: cross-entropy has the highest feature norm, followed by focal loss with $\gamma = 1$, followed by focal loss with $\gamma = 2$ and finally focal loss with $\gamma = 3$. Furthermore, this ordering is preserved throughout training. As we saw from Figure 5.4.1(e), from epoch 150 onwards (*i.e.*, the epoch from which the networks start getting miscalibrated), there is a flip in the ordering of weight norms of the last linear layer. From epoch 150 onwards, the weight norms also follow the same ordering that we observe from Figure C.9.1 here. This shows that throughout training the initial layer weights (before the last linear layer) of the network trained using focal loss are also regularized to favour the lower norm of the output features, thus possibly

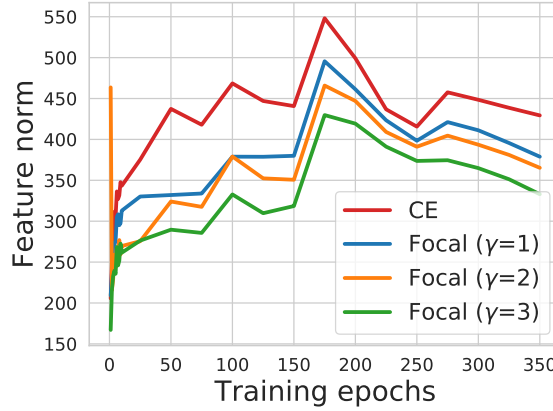


Figure C.9.1: L_2 norm of features obtained from the last ResNet block (before the linear layer) of ResNet-50 averaged over the entire training dataset of CIFAR-10 using a batch size of 128.

Criterion	Loss	Epoch	Error	ECE %
ECE	CE	151	7.34	1.69
ECE	FLSD-53	257	5.52	0.85
NLL	CE	153	6.69	2.28
NLL	FLSD-53	266	5.34	1.33
Error	CE	344	5.0	4.46
Error	FLSD-53	343	4.99	1.43
Full	CE	350	4.95	4.35
Full	FLSD-53	350	4.98	1.55

Table C.10.1: Classification errors and ECE scores obtained from ResNet-50 models trained using cross-entropy and focal loss with different early stopping criteria (best in hindsight ECE, NLL and classification error on the validation set) applied during training. In the table CE and FL stand for cross-entropy and focal loss respectively and the Full Criterion indicates models where early stopping has not been applied.

leading to less peakiness in the final prediction as compared to that of cross-entropy loss (see the ‘Peak at the wrong place’ paragraph of § 5.3).

C.10 Early stopping

From Figure 5.4.1(a), one may think that intermediate models (using early stopping) might provide better accuracy and calibration. However, there is no ideal approach for early stopping. For a fair comparison, we train ResNet50 on CIFAR-10 using cross-entropy and focal loss with the best (in hindsight) possible early stopping. We train each model for 350 epochs and choose the 3 intermediate models with the best validation set ECE, NLL, and classification error, respectively. We present the test set performance in Table C.10.1.

Metrics	CE ($\alpha = 0.0$)	LS ($\alpha = 0.1$)	FL ($\gamma = 1.0$)
ECE% Pre T / Post T / T	10.16/2.59/1.2	3.25/3.25/1.0	1.69/1.69/1.0
BLEU Pre T / Post T	26.31/26.21	26.33/26.33	26.39/26.39

Table C.11.1: Test set ECE and BLEU score both pre and post-temperature scaling for cross-entropy (CE) with hard targets, cross-entropy with label smoothing (LS) ($\alpha = 0.1$) and focal loss (FL) ($\gamma = 1$).

From the table, we can observe that: 1) On every early stopping criterion, the model trained on focal loss outperforms the one trained on cross-entropy in both error and ECE, 2) ECE as a stopping criterion provides a better test set ECE, but increases the test error significantly, 3) even without early stopping, focal loss achieves consistently better error and ECE compared to cross-entropy using any stopping criterion.

C.11 Machine Translation: A Downstream Task

In this section, we explore machine translation with beam search as a relevant downstream task for calibration. Following the setup in Müller et al. (2019), we train the Transformer architecture (Vaswani et al. 2017) on the WMT 2014 English-to-German translation dataset. The training settings (like the optimizer, LR schedule, *etc.*) are the same as Vaswani et al. (2017). We chose machine translation as a relevant task because the softmax vectors produced by the transformer model are directly fed into the beam search algorithm, and hence softmax outputs from a calibrated model should intuitively produce better translations and a better BLEU score.

We train three transformer models, one on cross-entropy with hard target labels, the second on cross-entropy with label smoothing (with smoothing factor $\alpha = 0.1$), and the third on focal loss with $\gamma = 1$. To compare these models in terms of calibration, we report the test set ECE (%) both before and after temperature scaling in the first row of Table C.11.1. Furthermore, to evaluate their performance on the English-to-German translation task, we also report the test set BLEU score of these models in the second row of Table C.11.1. Finally, to study the variation of test set BLEU score and validation set ECE with temperature, we plot them against temperature for all three models in Figure C.11.1.

We observe from Table C.11.1 that the model trained on focal loss outperforms its competitors on ECE and also has a competitive edge over other methods on BLEU score as well. The focal loss model also has an optimal temperature of 1, just like the model trained on cross-entropy with label smoothing. From Figure C.11.1, we can see that the

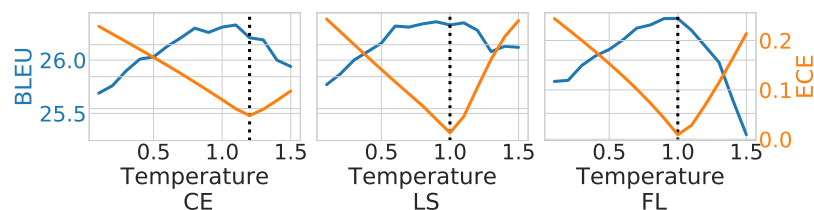


Figure C.11.1: Changes in *test set* BLEU score and *validation set* ECE with temperature, for models trained using (a) cross-entropy with hard targets (CE) (b) cross-entropy with label smoothing (LS) ($\alpha = 0.1$), and (c) focal loss (FL) ($\gamma = 1$).

models obtain the highest BLEU scores at around the same temperatures at which they obtain low ECEs, thereby confirming our initial notion that a more calibrated model provides better translations. However, since the optimal temperatures are tuned on the validation set, they don't often correspond to the best BLEU scores on the test set. On the test set, the highest BLEU scores we observe are 26.33 for cross-entropy, 26.36 for cross-entropy with label smoothing, and 26.39 for focal loss. Thus, the performance of focal loss on machine translation (a downstream task related to calibration) is also very encouraging.

D

Publications

D.1 Multi-Agent Diverse Generative Adversarial Networks

Contributions:

- Arnab Ghosh and Viveka Kulharia came up with the ideas and conducted the experiments.
- Initial supervision and support was provided by Vinay Namboodiri, and later Philip Torr and Puneet Dokania also fully supervised the project. Puneet extensively participated in the writing of the manuscript.

Multi-Agent Diverse Generative Adversarial Networks

Arnab Ghosh*
University of Oxford, UK
arnabg@robots.ox.ac.uk

Viveka Kulharia*
University of Oxford, UK
viveka@robots.ox.ac.uk

Vinay Namboodiri
IIT Kanpur, India
vinaypn@iitk.ac.in

Philip H.S. Torr
University of Oxford, UK
philip.torr@eng.ox.ac.uk

Puneet K. Dokania
University of Oxford, UK
puneet@robots.ox.ac.uk

Abstract

We propose MAD-GAN, an intuitive generalization to the Generative Adversarial Networks (GANs) and its conditional variants to address the well known problem of mode collapse. First, MAD-GAN is a multi-agent GAN architecture incorporating multiple generators and one discriminator. Second, to enforce that different generators capture diverse high probability modes, the discriminator of MAD-GAN is designed such that along with finding the real and fake samples, it is also required to identify the generator that generated the given fake sample. Intuitively, to succeed in this task, the discriminator must learn to push different generators towards different identifiable modes. We perform extensive experiments on synthetic and real datasets and compare MAD-GAN with different variants of GAN. We show high quality diverse sample generations for challenging tasks such as image-to-image translation and face generation. In addition, we also show that MAD-GAN is able to disentangle different modalities when trained using highly challenging diverse-class dataset (e.g. dataset with images of forests, icebergs, and bedrooms). In the end, we show its efficacy on the unsupervised feature representation task.

1. Introduction

Generative models have attracted considerable attention recently. The underlying idea behind such models is to attempt to capture the distribution of high-dimensional data such as images and texts. Though these models are highly useful in various applications, it is computationally expensive to train them as they require intractable integration in a very high-dimensional space. This drastically limits their applicability. However, recently there has been considerable progress in *deep generative models* – conglom-



Figure 1: Diverse-class data generation using MAD-GAN. Diverse-class dataset contains images from different classes/modalities (in this case, forests, icebergs, and bedrooms). Each row represents generations by a particular generator and each column represents generations for a given random noise input z . As shown, once trained using this dataset, generators of MAD-GAN are able to disentangle different modalities, hence, each generator is able to generate images from a particular modality.

erate of deep neural networks and generative models – as they do not explicitly require the intractable integration, and can be efficiently trained using back-propagation algorithm. Two such famous examples are Generative Adversarial Networks (GANs) [11] and Variational Autoencoders [14].

In this paper we focus on GANs as they are known to produce sharp and plausible images. Briefly, GANs employ a generator and a discriminator where both are involved in a minimax game. The task of the discriminator is to learn the difference between *real* samples (from true data distribution p_d) and *fake* samples (from generator distribution p_g). Whereas, the task of the generator is to maximize the mistakes of the discriminator. At convergence, the generator learns to produce real looking images. A few successful applications of GANs are video generation [26], image inpainting [22], image manipulation [29], 3D object generation [27], interactive image generation using few brush strokes [29], image super-resolution [17], diagrammatic abstract reasoning [15] and conditional GANs [20, 24].

Despite the remarkable success of GAN, it suffers from the major problem of *mode collapse* [2, 7, 8, 19, 25].

*Joint first author

Though, theoretically, convergence guarantees the generator learning the true data distribution. However, practically, reaching the true equilibrium is difficult and not guaranteed, which potentially leads to the aforementioned problem of mode collapse. Broadly speaking, there are two schools of thought to address the issue: (1) improving the learning of GANs to reach better optima [2, 19, 25]; and (2) explicitly enforcing GANs to capture diverse modes [7, 8, 18]. Here we focus on the latter.

Borrowing from the multi-agent algorithm [1] and coupled GAN [18], we propose to use multiple generators with one discriminator. We call this framework the Multi-Agent GAN architecture, as shown in Fig. 2. In detail, similar to the standard GAN, the objective of each generator here is to maximize the mistakes of the *common* discriminator. Depending on the task, it might be useful for different generators to share information. This is done using the initial layer parameters of generators. Another reason behind sharing these parameters is the fact that initial layers capture low-frequency structures which are almost the same for a particular type of dataset (for example, faces), therefore, sharing them reduces redundant computations. However, when the dataset contains images from completely different modalities, one can avoid sharing these parameters. Naively using multiple generators may lead to the *trivial solution* where all the generators learn to generate *similar* samples. To resolve this issue and generate different visually plausible samples capturing diverse high probability modes, we propose to modify the objective function of the discriminator. In the modified objective, along with finding the real and the fake samples, the discriminator also has to correctly identify the generator that generated the given fake sample. Intuitively, in order to succeed in this task, the discriminator must learn to push generations corresponding to different generators towards different identifiable modes. Combining the Multi-Agent GAN architecture with the diversity enforcing term allows us to generate diverse plausible samples, thus the name Multi-Agent Diverse GAN (MAD-GAN).

As an example, an intuitive setting where mode collapse occurs is when a GAN is trained on a dataset containing images from different modalities/classes. For example, a diverse-class dataset containing images such as forests, iceberg, and bedrooms. This is of particular interest as it not only requires the model to disentangle intra-class variations, it also requires inter-class disentanglement. Fig. 1 demonstrates the surprising effectiveness of MAD-GAN in this challenging setting. Generators among themselves are able to disentangle inter-class variations, and each generator is also able to capture intra-class variations.

In addition, we analyze MAD-GAN through extensive experiments and compare it with several variants of GAN. First, for the proof of concept, we perform experiments in controlled settings using synthetic dataset (mix-

ture of Gaussians), and complicated Stacked/Compositional MNIST datasets with hand engineered modes. In these settings, we empirically show that our approach outperforms all other GAN variants we compare with, and is able to generate high quality samples while capturing large number of modes. In a more realistic setting, we show high quality diverse sample generations for the challenging tasks of *image-to-image translation* [12] (conditional GAN) and *face generation* [8, 23]. Using the SVHN dataset [21], we also show the efficacy of our framework for learning the feature representation in an unsupervised setting.

We also provide theoretical analysis of this approach and show that the proposed modification in the objective of discriminator allows generators to learn together as a mixture model where each generator represents a mixture component. We show that at convergence, the global optimum value of $-(k+1)\log(k+1) + k\log k$ is achieved, where k is the number of generators.

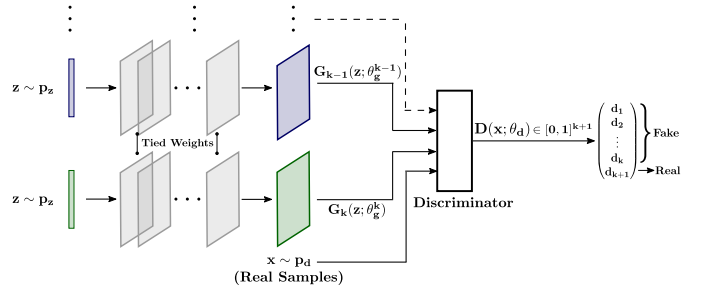


Figure 2: Multi-Agent Diverse GAN (MAD-GAN). The discriminator outputs $k+1$ softmax scores signifying the probability of its input sample being from either one of the k generators or the real distribution.

2. Related Work

The recent work called InfoGAN [8] proposed an information-theoretic extension to GANs in order to address the problem of mode collapse. Briefly, InfoGAN disentangles the latent representation by assuming a factored representation of the latent variables. In order to enforce that the generator learns factor specific generations, InfoGAN maximizes the mutual information between the factored latents and the generator distribution. Che *et al.* [7] proposed a mode regularized GAN (ModeGAN) which uses an encoder-decoder paradigm. The basic idea behind ModeGAN is that if a sample from the true data distribution p_d belongs to a particular mode, then the sample generated by the generator (fake sample) when the true sample is passed through the encoder-decoder is likely to belong to the same mode. ModeGAN assumes that there exists enough true samples from a mode for the generator to be able to capture it. Another work by Metz *et al.* [19] proposed a surrogate objective for the update of the generator with respect to the

unrolled optimization of the discriminator (UnrolledGAN) to address the issue of convergence of the training process of GANs. This improves the training process of the generator which in turn allow the generators to explore better coverage to true data distribution.

Liu *et al.* [18] presented Coupled GAN, a method for training two generators with shared parameters to learn the joint distribution of the data. The shared parameters guide both the generators towards similar subspaces but since they are trained independently on two domains, they promote diverse generations. Durugkar *et al.* [9] proposed a model with multiple discriminators whereby an ensemble of multiple discriminators have been shown to stabilize the training of the generator by guiding it to produce better samples.

W-GAN [3] is a recent technique which employs integral probability metrics based on the earth mover distance rather than the JS-divergences that the original GAN uses. BE-GAN [5] builds upon W-GAN using an autoencoder based equilibrium enforcing technique alongside the Wasserstein distance. DCGAN [23] was a seminal technique which used a fully convolutional generator and discriminator for the first time along with the introduction of batch normalization thus stabilizing the training procedure, and was able to generate compelling generations. GoGAN [13] introduced a training procedure for the training of the discriminator using a maximum margin formulation alongside the earth mover distance based on the Wasserstein-1 metric. [4] introduced a technique and theoretical formulation stating the importance of multiple generators and discriminators in order to completely model the data distribution. In terms of employing multiple generators, our work is closest to [4, 18, 10]. However, while using multiple generators, our method explicitly enforces them to capture diverse modes.

3. Preliminaries

Here we present a brief review of GANs [11]. Given a set of samples $\mathcal{D} = (x_i)_{i=1}^n$ from the true data distribution p_d , the GAN learning problem is to obtain the optimal parameters θ_g of a generator $G(z; \theta_g)$ that can sample from an approximate data distribution p_g , where $z \sim p_z$ is the prior input noise (*e.g.* samples from a normal distribution). In order to learn the optimal θ_g , the GAN objective (Eq. (1)) employs a discriminator $D(x; \theta_d)$ that learns to differentiate between a *real* (from p_d) and a *fake* (from p_g) sample x . The overall GAN objective is:

$$\min_{\theta_g} \max_{\theta_d} V(\theta_d, \theta_g) := \mathbb{E}_{x \sim p_d} \log D(x; \theta_d) + \mathbb{E}_{z \sim p_z} \log (1 - D(G(z; \theta_g); \theta_d)) \quad (1)$$

The above objective is optimized in a block-wise manner where θ_d and θ_g are optimized one at a time while fixing the other. For a given sample x (either from p_d or p_g)

and the parameter θ_d , the function $D(x; \theta_d) \in [0, 1]$ produces a score that represents the probability of x belonging to the true data distribution p_d (or probability of it being real). The objective of the discriminator is to learn parameters θ_d that maximizes this score for the true samples (from p_d) while minimizing it for the fake ones $\tilde{x} = D(z; \theta_g)$ (from p_g). In the case of generator, the objective is to minimize $\mathbb{E}_{z \sim p_z} \log (1 - D(G(z; \theta_g); \theta_d))$, equivalently maximize $\mathbb{E}_{z \sim p_z} \log D(G(z; \theta_g); \theta_d)$. Thus, the generator learns to maximize the scores for the fake samples (from p_g), which is exactly the opposite to what discriminator is trying to achieve. In this manner, the generator and the discriminator are involved in a minimax game where the task of the generator is to maximize the mistakes of the discriminator. Theoretically, at equilibrium, the generator learns to generate real samples, which means $p_g = p_d$.

4. Multi-Agent Diverse GAN

In the GAN objective, one can argue that the task of a generator is much harder than that of the discriminator as it has to produce real looking images to maximize the mistakes of the discriminator. This, along with the minimax nature of the objective raise several challenges for GANs [2, 7, 8, 19, 25]: (1) mode collapse; (2) difficult optimization; and (3) trivial solution. In this work we propose a new framework to address the first challenge of *mode collapse* by increasing the capacity of the generator while using well known tricks to partially avoid other challenges [2].

Briefly, we propose a *Multi-Agent GAN architecture* that employs multiple generators and one discriminator in order to generate different samples from high probability regions of the true data distribution. In addition, theoretically, we show that our formulation allows generators to act as a mixture model with each generator capturing one component.

4.1. Multi-Agent GAN Architecture

Here we describe our proposed architecture (Fig. 2). It involves k generators and one discriminator. In the case of homogeneous data (all the images belong to same class, *e.g.* faces or birds), we allow all the generators to share information by tying most of the initial layer parameters. This is essential to avoid redundant computations as initial layers of a generator capture low-frequency structures which are almost the same for a particular type of dataset. This also allows different generators to converge faster. However, in the case of *diverse-class* data (*e.g.* dataset with a mixture of different classes such as forests, icebergs *etc.*), it is necessary to avoid sharing these parameters to allow each generator to capture content specific structures. Thus, the extent to which one should share these parameters depends on the task at hand.

More specifically, given $z \sim p_z$ for the i -th generator, similar to the standard GAN, the first step involves gen-

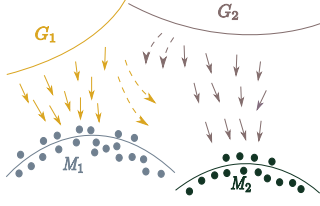


Figure 3: Visualization of different generators getting pushed towards different modes. Here, M_1 and M_2 could be a cluster of modes where each cluster itself contains many modes. The arrows abstractly represent generator specific gradients for the purpose of building intuition.

erating a sample (for example, an image) $\tilde{x}_i$. Since each generator receives the same latent input sampled from the same distribution, naively using this simple approach may lead to the *trivial solution* where all the generators learn to generate similar samples. In what follows, we propose an intuitive solution to avoid this issue and allow the generators to capture diverse modes.

4.2. Enforcing Diverse Modes

Inspired by the discriminator formulation for the semi-supervised learning [25], we use a generator identification based objective function that, along with minimizing the score $D(\tilde{x}; \theta_d)$, requires the discriminator to identify the generator that generated the given fake sample $\tilde{x}$. In order to do so, as opposed to the standard GAN objective function where the discriminator outputs a scalar value, we modify it to output $k+1$ soft-max scores. In more detail, given the set of k generators, the discriminator produces a soft-max probability distribution over $k+1$ classes. The score at $(k+1)$ -th index ($D_{k+1}(\cdot)$) represents the probability that the sample belongs to the true data distribution and the score at $j \in \{1, \dots, k\}$ -th index represents the probability of it being generated by the j -th generator. Under this setting, while learning θ_d , we optimize the cross-entropy between the soft-max output of the discriminator and the Dirac delta distribution $\delta \in \{0, 1\}^{k+1}$, where for $j \in \{1, \dots, k\}$, $\delta(j) = 1$ if the sample belongs to the j -th generator, otherwise $\delta(k+1) = 1$. Thus, the objective of the discriminator, which is optimizing θ_d while keeping θ_g constant (refer Eq. (1)), is modified to:

$$\max_{\theta_d} \mathbb{E}_{x \sim p} H(\delta, D(x; \theta_d))$$

where, $Supp(p) = \cup_{i=1}^k Supp(p_{g_i}) \cup Supp(p_d)$ and $H(\cdot, \cdot)$ is the negative of the cross entropy function. Intuitively, in order to correctly identify the generator that produced a given fake sample, the discriminator must learn to push different generators towards different identifiable modes. However, the objective of each generator remains the same as in the standard GAN. Thus, for the i -th generator, the

objective is to minimize the following:

$$\mathbb{E}_{x \sim p_d} \log D_{k+1}(x; \theta_d) + \mathbb{E}_{z \sim p_z} \log(1 - D_{k+1}(G_i(z; \theta_g^i); \theta_d))$$

To update the parameters, the gradient for each generator is simply computed as $\nabla_{\theta_g^i} \log(1 - D_{k+1}(G_i(z; \theta_g^i); \theta_d))$. Notice that all the generators in this case can be updated in parallel. For the discriminator, given $x \sim p$ (can be real or fake) and corresponding δ , the gradient is $\nabla_{\theta_d} \log D_j(x; \theta_d)$, where $D_j(x; \theta_d)$ is the j -th index of $D(x; \theta_d)$ for which $\delta(j) = 1$. Therefore, using this approach requires *very minor modifications to the standard GAN optimization algorithm* and can be easily used with different variants of GAN. An intuitive visualization is shown in Fig. 3.

Theorem 1 shows that the above objective function actually allows generators to form a mixture model where each generator represents a mixture component and the global optimum of $-(k+1) \log(k+1) + k \log k$ is achieved when $p_d = \frac{1}{k} \sum_{i=1}^k p_{g_i}$. Notice that, at $k = 1$, which is the case with one generator, we obtain exactly the same Jensen-Shannon divergence based objective function as shown in [11] with the optimal value of $-\log 4$.

Theorem 1. *Given the optimal discriminator, the objective for training the generators boils down to minimizing*

$$KL(p_d(x) || p_{avg}(x)) + k KL\left(\frac{1}{k} \sum_{i=1}^k p_{g_i}(x) || p_{avg}(x)\right) - (k+1) \log(k+1) + k \log k \quad (2)$$

where, $p_{avg}(x) = \frac{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}{k+1}$. The above objective function obtains its global minimum if $p_d = \frac{1}{k} \sum_{i=1}^k p_{g_i}$ with the objective value of $-(k+1) \log(k+1) + k \log k$.

Proof. The joint objective of all the generators is to minimize the following:

$$\mathbb{E}_{x \sim p_d} \log D_{k+1}(x) + \sum_{i=1}^k \mathbb{E}_{x \sim p_{g_i}} \log(1 - D_{k+1}(x))$$

Using Corollary 1, we substitute the optimal discriminator in the above equation and obtain:

$$\begin{aligned} & \mathbb{E}_{x \sim p_d} \log \left[\frac{p_d(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)} \right] + \\ & \sum_{i=1}^k \mathbb{E}_{x \sim p_{g_i}} \log \left[\frac{\sum_{i=1}^k p_{g_i}(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)} \right] \\ & = \mathbb{E}_{x \sim p_d} \log \left[\frac{p_d(x)}{p_{avg}(x)} \right] + k \mathbb{E}_{x \sim p_g} \log \left[\frac{p_g(x)}{p_{avg}(x)} \right] \\ & - (k+1) \log(k+1) + k \log k \end{aligned} \quad (3)$$

where, $p_g = \frac{\sum_{i=1}^k p_{g_i}}{k}$ and $p_{avg}(x) = \frac{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}{k+1}$. Note that, Eq. (3) is exactly the same as Eq. (2). When $p_d = \frac{\sum_{i=1}^k p_{g_i}}{k}$, both the KL terms become zero and the global minimum is achieved. $\square$

Corollary 1. *For fixed generators, the optimal distribution learned by the discriminator D has the following form:*

$$D_{k+1}(x) = \frac{p_d(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)},$$

$$D_i(x) = \frac{p_{g_i}(x)}{p_d(x) + \sum_{i=1}^k p_{g_i}(x)}, \forall i \in \{1, \dots, k\}.$$

where, $D_i(x)$ represents the i -th index of $D(x; \theta_d)$, p_d the true data distribution, and p_{g_i} the distribution learned by the i -th generator.

Proof. For fixed generators, the objective function of the discriminator is to maximize

$$\mathbb{E}_{x \sim p_d} \log D_{k+1}(x) + \sum_{i=1}^k \mathbb{E}_{x_i \sim p_{g_i}} \log D_i(x_i)$$

where, $\sum_{i=1}^{k+1} D_i(x) = 1$ and $D_i(x) \in [0, 1], \forall i$. The above equation can be written as:

$$\begin{aligned} & \int_x p_d(x) \log D_{k+1}(x) dx + \sum_{i=1}^k \int_x p_{g_i}(x) \log D_i(x) dx \\ &= \int_{x \in p} \sum_{i=1}^{k+1} p_i(x) \log D_i(x) dx \end{aligned} \quad (4)$$

where, $p_{k+1}(x) := p_d(x)$, $p_i(x) := p_{g_i}(x), \forall i \in \{1, \dots, k\}$, and $Supp(p) = \bigcup_{i=1}^k Supp(p_{g_i}) \cup Supp(p_d)$. Therefore, for a given x , the optimum of objective function defined in Eq. (4) with constraints defined above can be obtained using Proposition 1. $\square$

Proposition 1. *Given $\mathbf{y} = (y_1, \dots, y_n)$, $y_i \geq 0$, and $a_i \in \mathbb{R}$, the optimal solution for the objective function defined below is achieved at $y_i^* = \frac{a_i}{\sum_{i=1}^n a_i}, \forall i$*

$$\max_{\mathbf{y}} \sum_{i=1}^n a_i \log y_i, \quad s.t. \quad \sum_{i=1}^n y_i = 1$$

Proof. The Lagrangian of the above problem is:

$$L(\mathbf{y}, \lambda) = \sum_{i=1}^n a_i \log y_i + \lambda \left(\sum_{i=1}^n y_i - 1 \right)$$

Differentiating w.r.t y_i and λ , and equating to zero,

$$\frac{a_i}{y_i} + \lambda = 0, \quad \sum_{i=1}^n y_i - 1 = 0$$

Solving the above two equations, we obtain $y_i^* = \frac{a_i}{\sum_{i=1}^n a_i}$. $\square$

GAN Variants	Chi-square ($\times 10^5$)	KL-Div
DCGAN [23]	0.90	0.322
WGAN [3]	1.32	0.614
BEGAN [5]	1.06	0.944
GoGAN [13]	2.52	0.652
Unrolled GAN [19]	3.98	1.321
Mode-Reg DCGAN [7]	1.02	0.927
InfoGAN [8]	0.83	0.21
MA-GAN	1.39	0.526
MAD-GAN (Our)	0.24	0.145

Table 1: Synthetic experiment on 1D GMM (Fig. 4).

5. Experiments

We present an extensive quantitative and qualitative analysis of MAD-GAN on various synthetic and real-world datasets. First, we use a simple 1D mixture of Gaussians and also Stacked/Compositional MNIST dataset (1000 modes) to compare MAD-GAN with several known variants of GANs, such as DCGAN [23], WGAN [3], BEGAN [5], GoGAN [13], Unrolled GAN [19], Mode-Reg GAN [7] and InfoGAN [8]. Furthermore, we created another baseline, called *MA-GAN (Multi-Agent GAN)*, which is a trivial extension of GAN with multiple generators and one discriminator. As opposed to MAD-GAN, MA-GAN has a simple Multi-Agent architecture without modifications to the objective of the discriminator. This comparison allows us to understand the effect of explicitly enforcing diversity in the objective of the MAD-GAN. We use KL-divergence [16] and number of modes recovered [7] as the criterion for comparisons and show superior results compared to all the other methods. Additionally, we show diverse generations for the challenging tasks of *image-to-image translation* [12], *diverse-class data generation*, and face generation. It is non-trivial to devise a metric to evaluate diversity on these high quality generation tasks, so we perform qualitative assessment. Note that, the image-to-image translation objective is known to learn the delta distribution, thus, it is agnostic to the input noise vector. However, we show that MAD-GAN is able to produce highly plausible diverse generations for this task. In the end, we show the efficacy of MAD-GAN in unsupervised feature representation learning task. We provide detailed overview of the architectures, datasets, and the parameters used in our experiments in the supplementary.

In the case of InfoGAN [8], we varied the dimension of the categorical variable, depicting the number of modes, to obtain the best cross-validated results.

5.1. Non-Parametric Density Estimation

In order to understand the behavior of MAD-GAN and different state-of-the-art GAN models, we first perform a

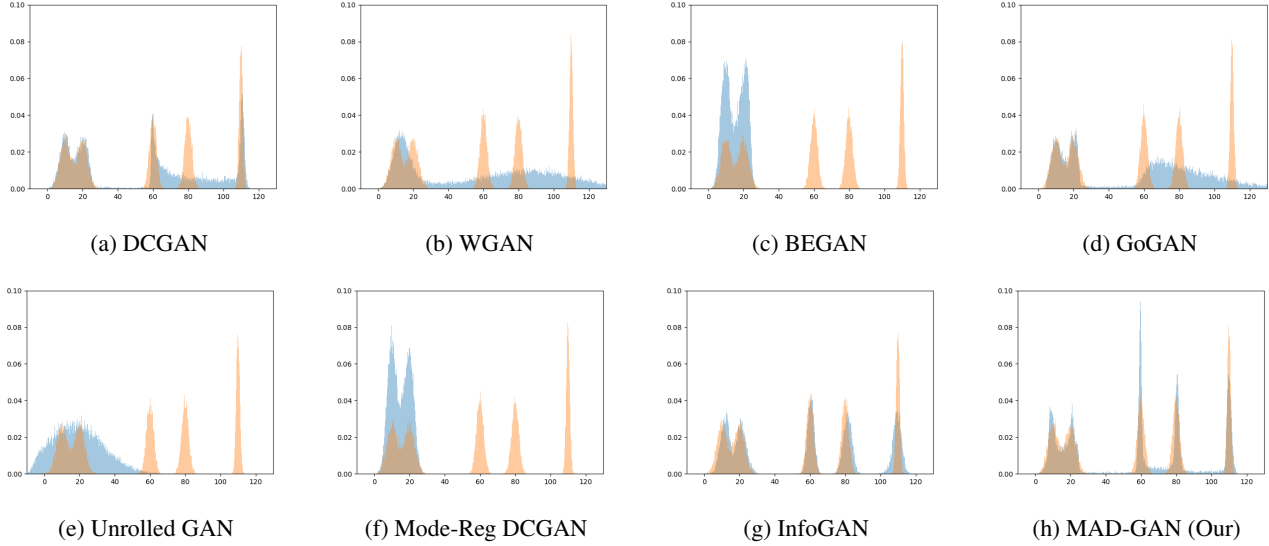


Figure 4: A toy example to understand the behaviour of different GAN variants in order to compare with MAD-GAN (each method was trained for 198000 iterations). The orange bars show the density estimate of the training data and the blue ones for the generated data points. After careful cross-validation, we chose the bin size of 0.1.

very simple synthetic experiment, much easier than generating high-dimensional complex images. We consider a distribution of 1D GMM [6] having five mixture components with modes at 10, 20, 60, 80 and 110, and standard deviations of 3, 3, 2, 2 and 1, respectively. While the first two modes overlap significantly, the fifth mode stands isolated as shown in Fig. 4. We train different GAN models using 200,000 samples from this distribution and generate 65,536 data points from each model. In order to compare the learned distribution with the ground truth distributions, we first estimate them using bins over the data points and create the histograms. These histograms are carefully created using different bin sizes and the best bin (found to be 0.1) is chosen. Then, we use Chi-square distance and the KL-divergence to compute distance between the two histograms. From Fig. 4 and Tab. 1 it is evident that MAD-GAN is able to capture all the clustered modes which includes significantly overlapped modes as well. MAD-GAN obtains the minimum value in terms of both Chi-square distance and the KL-divergence. In this experiment, both MAD-GAN and MA-GAN used four generators. In the case of InfoGAN, we used 5 dimensional categorical variable, which provides the best result.

5.2. Stacked and Compositional MNIST

We now perform experiments on a more challenging setup, similar to [7, 19], in order to examine and compare MAD-GAN with other GAN variants. [19] created a Stacked-MNIST dataset with 25,600 samples where each sample has three channels stacked together with a random MNIST digit in each of them. Thus, it creates 1000 distinct

GAN Variants	KL Div	# Modes Covered
DCGAN [23]	2.15	712
WGAN [3]	1.02	868
BEGAN [5]	1.89	819
GoGAN [13]	2.89	672
Unrolled GAN [19]	1.29	842
Mode-Reg DCGAN [7]	1.79	827
InfoGAN [8]	2.75	840
MA-GAN	3.4	700
MAD-GAN (Our)	0.91	890

Table 2: Stacked-MNIST experiments and comparisons. Note that three generators are used for MAD-GAN.

GAN Variants	KL Div	# Modes Covered
DCGAN [23]	0.18	980
WGAN [3]	0.25	1000
BEGAN [5]	0.19	999
GoGAN [13]	0.87	972
Unrolled GAN [19]	0.091	1000
Mode-Reg DCGAN [7]	0.12	992
InfoGAN [8]	0.47	990
MA-GAN	1.62	997
MAD-GAN (Our)	0.074	1000

Table 3: Compositional-MNIST experiments and comparisons. Note that three generators are used for MAD-GAN.



Figure 6: Diverse generations for *edges-to-handbags* generation task. In each sub-figure, the first column represents the input, columns 2-4 represents generations by MAD-GAN (using three generators), and columns 5-7 are generations by InfoGAN (using three categorical codes). It is evident that different generators of MAD-GAN are able to produce diverse results capturing different colors, textures, design patterns, among others. However, InfoGAN generations are visually almost the same, indicating mode collapse.

modes in the data distribution. [19] used a stripped down version of the generator and discriminator pair to reduce the modeling capacity. We do the same for fair comparisons and used the same architecture as mentioned in their paper. Similarly, [7] created Compositional-MNIST whereby they took 3 random MNIST digits and place them at the 3 quadrants of a 64×64 dimensional image. This also resulted in a data distribution with 1000 hand-designed modes. The distribution of the resulting generated samples was estimated using a pretrained MNIST classifier to classify each of the digits either in the channels or the quadrants to decide the mode it belongs to.

Tables 2 and 3 provide comparison of our method with variants of GAN in terms of KL divergence and the number of modes recovered for the Stacked and Compositional MNIST datasets, respectively. In Stacked-MNIST, as evident from the Tab. 2, MAD-GAN outperforms all other variants of GAN in both the criteria. Interestingly, in the case of Compositional-MNIST, as shown in Tab. 3, MAD-GAN, WGAN and Unrolled GAN were able to recover all the 1000 modes. However, in terms of KL divergence, the distribution generated by MAD-GAN is the closest to the true data distribution.

5.3. Diverse Samples for Image-to-Image Translation and Comparison to InfoGAN

Here we present experimental results on the challenging task of image-to-image translation [12] which uses conditional variant of GANs [20]. Conditional GAN for this task is known to learn the delta distribution, thus, generates the same image irrespective of the variations in the input noise vector. Generating diverse samples in this setting in itself is an open problem. We show that MAD-GAN is able to



Figure 8: Diverse generations for *night-to-day* image generation task. First column in each sub-figure represents the input. The remaining three columns show the diverse generations of three different generators of MAD-GAN (Our).

generate diverse samples in these experiments as well. We use three generators for MAD-GAN experiments and show three diverse generations. Note that, we do not claim to capture all the possible modes present in the data distribution because firstly we cannot estimate the number of modes a priori, and secondly, even if we could, we do not know how diverse the generations would be after using certain number of generators. We follow the same approach as [12] and employ patch based conditional GAN.

We compare MAD-GAN with InfoGAN [8] in these experiments as it is closest to our approach and can be used in image-to-image translation task. Theoretically, latent codes in InfoGAN should enable diverse generations. However, InfoGAN can only be used when the bias introduced by the categorical variables have significant impact on the generator network. For image-to-image translation and high resolution generations, the categorical variable does not have



Figure 9: Face generations using MAD-GAN. Each generator employed is DCGAN. Each row represents a generator. Each column represents generations for a given random noise input z . Note that, the first generator is generating faces pointing to the left. The second generator is generating female faces with long hair, while the third generator generates images with light background.

sufficient impact on the generations. As will be seen shortly, we validate this hypothesis by comparing our method with InfoGAN for this task. For the InfoGAN generator, to capture three kinds of distinct modes, the categorical code is chosen to take three values. Since we are dealing with images, in this case, the categorical code is a 2D matrix in which we set one third of the entries to 1 and remaining to 0 for each category. The generator is fed input image along with categorical code appended channel wise to the image. Architecture of the Q network is same as that of the pix2pix discriminator [12], except that the output is a vector of size 3 for the prediction of the categorical codes. The discriminator and the Q network parameters are unshared. We observed that sharing them and even adding noise besides categorical code did not make any difference. Note that, we tried different variations of the categorical codes but did not observe any significant variation in the generations.

Fig. 6 shows generations by MAD-GAN and InfoGAN for the *edges-to-handbags* task, where given the edges of handbags, the objective is to generate real looking handbags. Clearly, each MAD-GAN generator is able to produce meaningful images but different from remaining generators in terms of color, texture, and patterns. However, InfoGAN generations are almost the same for all the three categorical codes. In addition, in Fig. 8, we show diverse generations for the *night-to-day* task, where given night images of places, the objective is to generate their corresponding day images. As can be seen, the generated day images in Fig. 8 differ in terms of lighting conditions, sky patterns, weather conditions, and many other minute yet useful cues.

5.4. Diverse-Class Data Generation

To further explore the mode capturing capacity of MAD-GAN, we experimented with a much more challenging task of diverse-class data generation. In detail, we trained MAD-GAN (three generators) on a combined dataset consisting of various highly diverse images such as *islets*, *icebergs*, *broadleaf-forest*, *bamboo-forest*, and *bedroom*, obtained from the Places dataset [28]. Images were randomly

selected from each of them, creating a training dataset of 24,000 images. The generators have the same architecture as that of DCGAN. In this case, as the images in the dataset belong to different classes, we did not share the generator parameters. As shown in Fig. 1, to our surprise, we found that even in this highly challenging setting, the generations from different generators belong to different classes. This clearly indicates that the generators in MAD-GAN are able to disentangle inter-class variations. In addition, each generator for different noise input is able to generate diverse samples, indicating intra-class diversity.

5.5. Diverse Face Generation

Here we show diverse face generations (CelebA dataset) using MAD-GAN where we use DCGAN [23] as each of our three generators. Again, we use the same setting as provided in DCGAN. The high quality face generations are shown in the Fig. 9.

5.6. Unsupervised Representation Learning

Similar to DCGAN [23], we train our framework using SVHN dataset [21]. The trained discriminator is used to extract features. Using these features, we train an SVM for the classification task. For the MAD-GAN, with three generators, we obtained misclassification error of 17.5% which is almost 5% better than the results reported by DCGAN (22.48%). This clearly indicates that our framework is able to learn a better feature space in an unsupervised setting.

6. Conclusion

We presented a very simple and effective framework, Multi-Agent Diverse GAN (MAD-GAN), for generating diverse and meaningful samples. We showed the efficacy of our approach and compared it with various variants of GAN that it captures diverse modes while producing high quality samples. We presented a theoretical analysis of MAD-GAN with conditions for global optimality. Looking forward, an interesting future direction would be to estimate a priori the number of generators needed for a particular dataset. It is not clear how to do that given that we do not have access to the true data distribution. In addition, we would also like to theoretically understand the limiting cases that depend on the relationship between the number of generators and the complexity of the data distribution. Another interesting direction would be to exploit different generators such that their combinations can be used to capture diverse modes.

7. Acknowledgements

This work was supported by the EPSRC, ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1.

References

- [1] M. Abadi and D. Andersen. Learning to protect communications with adversarial neural cryptography. *arXiv preprint arXiv:1610.06918*, 2016.
- [2] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017.
- [3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. In *International Conference on Machine Learning*, 2017.
- [4] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang. Generalization and equilibrium in generative adversarial nets (gans). In *International Conference on Machine Learning*, 2017.
- [5] D. Berthelot, T. Schumm, and L. Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.
- [6] C. Bishop. Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn. *Springer, New York*, 2007.
- [7] T. Che, Y. Li, A. Jacob, Y. Bengio, and W. Li. Mode regularized generative adversarial networks. In *International Conference on Learning Representations*, 2017.
- [8] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2016.
- [9] I. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. In *International Conference on Learning Representations*, 2017.
- [10] A. Ghosh, V. Kulharia, and V. Namboodiri. Message passing multi-agent gans. *arXiv preprint arXiv:1612.01294*, 2016.
- [11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.
- [12] P. Isola, J.-Y. Zhu, T. Zhou, and A. Efros. Image-to-image translation with conditional adversarial networks. In *Computer Vision and Pattern Recognition*, 2017.
- [13] F. Juefei-Xu, V. N. Boddeti, and M. Savvides. Gang of gans: Generative adversarial networks with maximum margin ranking. *arXiv preprint arXiv:1704.04865*, 2017.
- [14] D. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.
- [15] V. Kulharia, A. Ghosh, A. Mukerjee, V. Namboodiri, and M. Bansal. Contextual rnn-gans for abstract reasoning diagram generation. In *AAAI Conference on Artificial Intelligence*, 2017.
- [16] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 1951.
- [17] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *Computer Vision and Pattern Recognition*, 2017.
- [18] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, 2016.
- [19] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. In *International Conference on Learning Representations*, 2017.
- [20] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [21] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.
- [22] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Computer Vision and Pattern Recognition*, 2016.
- [23] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [24] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, 2016.
- [25] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, 2016.
- [26] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, 2016.
- [27] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, 2016.
- [28] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [29] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, 2016.

D.2 Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation

Contributions:

- Viveka Kulharia, Siddhartha Chandra, Amit Agrawal, and Ambrish Tyagi came up with the ideas during weekly discussions at Amazon, and Viveka conducted the experiments.
- Siddhartha Chandra took over the project and completed any remaining experiments of the paper after Viveka Kulharia returned from the internship.
- Siddhartha Chandra, Amit Agrawal, Philip H.S. Torr, and Ambrish Tyagi helped in the writing.



Box2Seg: Attention Weighted Loss and Discriminative Feature Learning for Weakly Supervised Segmentation

Viveka Kulharia² , Siddhartha Chandra¹, Amit Agrawal¹, Philip Torr²,
and Ambrish Tyagi¹

¹ Amazon Lab126, Sunnyvale, USA

² University of Oxford, Oxford, UK
vivekakulharia@gmail.com

Abstract. We propose a weakly supervised approach to semantic segmentation using bounding box annotations. Bounding boxes are treated as noisy labels for the foreground objects. We predict a per-class attention map that saliently guides the per-pixel cross entropy loss to focus on foreground pixels and refines the segmentation boundaries. This avoids propagating erroneous gradients due to incorrect foreground labels on the background. Additionally, we learn pixel embeddings to simultaneously optimize for high intra-class feature affinity while increasing discrimination between features across different classes. Our method, Box2Seg, achieves state-of-the-art segmentation accuracy on PASCAL VOC 2012 by significantly improving the mIOU metric by 2.1% compared to previous weakly supervised approaches. Our weakly supervised approach is comparable to the recent fully supervised methods when fine-tuned with limited amount of pixel-level annotations. Qualitative results and ablation studies show the benefit of different loss terms on the overall performance.

1 Introduction

The accuracy of semantic segmentation approaches has improved significantly in recent years [8, 10, 32, 35, 53, 54, 56]. The mean Intersection-over-Union (mIoU) metric on the PASCAL VOC semantic segmentation benchmark has improved by over 20% in the last five years. The success of these efforts can be broadly attributed to (i) advancements in deep neural network architectures and loss functions (ii) efficient processing (better GPUs), and (iii) the availability of large

V. Kulharia, S.Chandra — Equally Contributed.

V. Kulharia was an intern at Amazon Lab126.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58583-9_18) contains supplementary material, which is available to authorized users.

© Springer Nature Switzerland AG 2020

A. Vedaldi et al. (Eds.): ECCV 2020, LNCS 12372, pp. 290–308, 2020.

https://doi.org/10.1007/978-3-030-58583-9_18

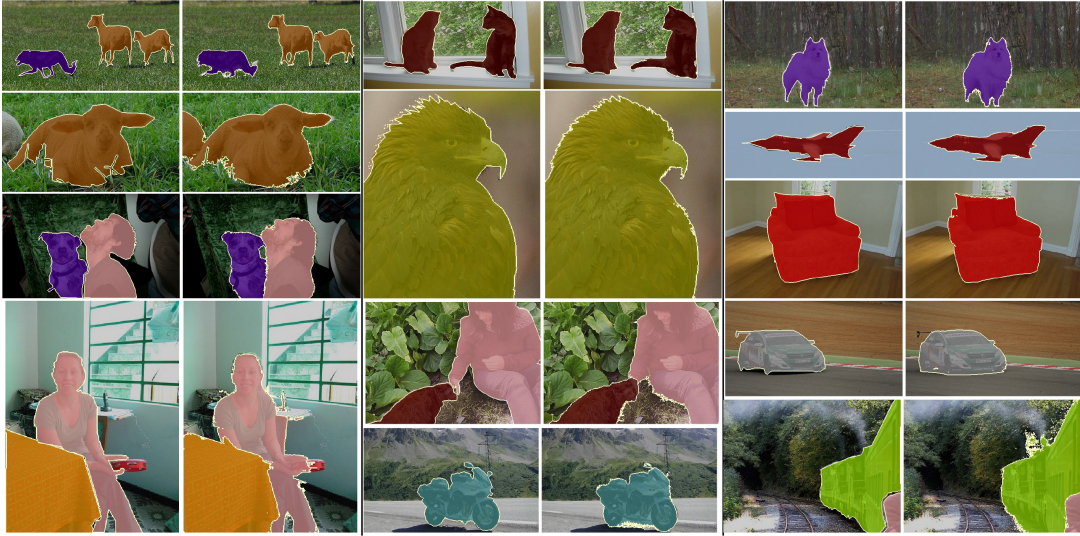


Fig. 1. Our Box2Seg model is able to produce high quality semantic segmentation using only bounding box annotations. Each image pair shows ground-truth (left) and predicted segmentation (right).

datasets of images with human labeled per-pixel annotations [13, 30]. Improvements in network architectures and hardware capabilities benefit all deep learning tasks. However, large datasets with per-pixel semantic labels are both expensive and slow to obtain [4, 13] (typically 4–10 minutes per image), making it challenging to scale to a large number of object categories. Consequently, even the largest semantic segmentation datasets [30, 58] include less than a couple of hundred object categories.

To address the scarcity of labeled data, some previous works have used synthetic datasets [43]. While labeling synthetically generated datasets involves little annotation effort, models trained on them do not always generalize well to the real world due to the domain gap between the real and the synthetic images. Alternative training strategies such as semi-, self-, or weak-supervision that require simpler/fewer labels (eg. image-level labels or bounding box annotations) have also been proposed. In this work, we show how to leverage real images with weak supervision to advance the state-of-the-art (SOTA) for semantic segmentation (refer to Fig. 1 for sample results).

Bounding box annotations yield high quality ground truth at a small cost. According to [4, 13], per-pixel labeling takes over 4 minutes per image compared to ~ 7 s (35x faster) for annotating bounding boxes [36]. Furthermore, large datasets with bounding box annotations containing over 9 million images are publicly available [25]. Weakly supervised approaches using bounding box annotations have been shown to be more accurate when compared to methods that use only image-level labels. SOTA segmentation result on VOC using bounding box annotations [46] outperforms the SOTA methods using image-level labels [27] by $\sim 4\%$. Our work, *Box2Seg*, builds upon previous approaches that use bounding box annotations.

A key intuition of our paper is to consider bounding box annotations as containing *label noise* for the foreground object. Since bounding box annotation is a super-set of the actual object segmentation, this label noise is one-sided. In other words, some foreground labels are incorrectly assigned to background pixels within the bounding box. However, all foreground pixels inside the box and background pixels outside all boxes are correctly labeled. Typical fully supervised segmentation training considers the label for every pixel as correct and gradients are back-propagated from all pixels. This would be an issue for a weakly supervised segmentation algorithm. To handle this, our algorithm predicts a novel per-pixel class-specific attention map and pixel embeddings in addition to the per-pixel segmentation output. The attention map is used to modulate the per-pixel cross entropy loss to handle label noise and reduces propagation of incorrect gradients. Thus, the attention map allows us to automatically discover salient regions of the object within the bounding box. The attention map is regularized using a soft filling-rate constraint for each training image.

We learn discriminative feature embeddings to capture long-range pairwise relationships between pixels across an image. We pretrain these embeddings to maximize the affinity between pixels belonging to same classes, while at the same time increasing the distance between features corresponding to different classes. During training, we define a novel loss function on pairs of pixels such that it encourages the pixel affinities to align with the predicted segmentation probabilities. A few methods have also proposed using feature affinity as an explicit measure to improve segmentation in fully-supervised and co-segmentation settings [5, 18, 22, 31, 33]. Affinities have also been used in some weakly supervised approaches [1, 2], but they are trained in a fully-supervised manner using pseudo ground truth derived from class activation maps, as opposed to our embeddings which are trained by minimizing disagreements between them and estimated segmentation output (Sect. 3.4). We show how discriminative feature learning obtained using the model predictions can also be incorporated and is helpful in the context of weakly supervised semantic segmentation.

The remainder of this paper is organized as follows. We discuss related work in Sect. 2. Our learning algorithm and loss functions are described in Sect. 3. We demonstrate SOTA results on the PASCAL VOC 2012 segmentation benchmark (Sect. 4), outperforming previous weakly-supervised approaches by 2.1% on the mIoU metric. We also show that our weakly supervised model serves as a good starting point for semi-supervised learning tasks, surpassing fully supervised baselines pre-trained with ImageNet [12] with only a fraction of pixel-level segmentation annotations.

2 Related Work

Previous works on weakly-supervised semantic segmentation have used image-level annotations [17, 20, 26, 27, 42, 50, 52], points/clicks [4], scribbles [29, 47–49], bounding box annotations [11, 19, 37, 41, 46, 55] and adversarial training [3, 21]. We take a closer look at some of these methods and categorize them based on the labels required and their methodology.

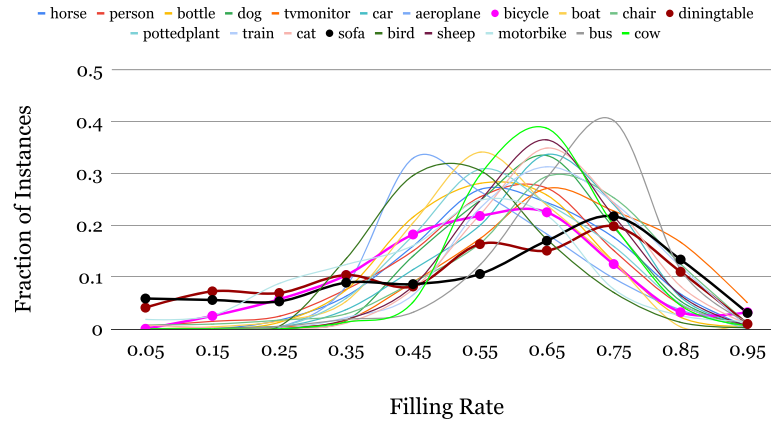


Fig. 2. Distribution of filling rate in PASCAL VOC 2012 training dataset. The per-class filling rates vary widely, especially for the categories such as sofa, dining table and bicycle. Unlike [46] that proposes using 1-3 filling rates per class, we use a per-image class specific filling rate.

Image Level Labels: Deep learning approaches that use image-level labels typically train a classification model first to recover coarse class activation maps [57], which describe class-discriminative image regions. The predicted class activation maps are then used as ‘seeds’ for optimization methods that grow the coarse activation maps to larger pseudo segmentation maps. A number of optimization methods such as super-pixelization [26], deep seeded region growing [20], conditional random fields [42, 52], and combinatorial grouping [40] have been proposed. Finally, the pseudo segmentation maps are used as ground truth to train the segmentation model [28]. Some approaches additionally employ a class-agnostic saliency estimation model [6, 50, 52] to capture *objectness* of pixels, and others employ Expectation-Maximization (EM) [17] to iteratively refine the pseudo ground truth and the parameters of the segmentation model.

Bounding Box Labels: The availability of bounding box annotations alleviates the need to estimate class activation maps for localizing objects of interest. The bounding boxes serve as crude segmentation masks which are refined using heuristic cues [23] and graph based optimization algorithms such as Grab-Cut [44] or mean-field inference [24] on a densely-connected conditional random field [37, 41, 46]. Previous works have used the refined masks for training the segmentation model [28]. Some of these approaches additionally use EM [11, 23, 37, 55] for iterative refinement of the ground truth and model parameters. However, training models with EM is time-consuming and is prone to propagation of errors over iterations; In contrast, our proposed approach does not require iterative refinement and outperforms these methods by a large margin (Sect. 4).

Multi-tasking: Hu *et al.* [19] train an object detector and segmentation model simultaneously using the Mask-RCNN framework [15] assuming a closed form relationship between parameters of the detection and segmentation branches. While [19] is able to benefit from the advantages of multi-tasking, it requires

(a) (limited) training data with per-pixel annotations (b) a region of interest (RoI) proposal method, and (c) uses the RoI warping module which captures local context alone. Our approach does not require any per-pixel annotations and uses a fully-convolutional network. We capture global image context via long-range interactions by training pixel embeddings.

Label Noise: We consider bounding boxes as noisy labels for foreground objects. Some fully-supervised approaches [34, 39] have tackled noise in the segmentation ground truth by consolidating predictions from two or more independent classifiers and discarding pixels with ambiguous predictions from the training data. Among weakly supervised methods, Song *et al.* [46] propose the idea of using the filling-rate as a cue to supervise training with bounding box ground truth. The filling-rate of a class [46] is defined as the average proportion of pixels in a bounding box that instances of the class occupy, and is estimated by applying dense-CRF [24] for foreground extraction within bounding boxes. For example, suppose in the PASCAL VOC 2012 dataset, instances of ‘sheep’ occupy roughly 60% of pixels in each bounding-box. During training, their approach ignores gradients from the 40% pixels with the lowest confidences in each bounding box containing ‘sheep’. In contrast, we advocate estimating a *per-image class-specific* spatial attention map, to allow for large intra-class variations in object appearance (see Fig. 2). Instead of ignoring loss on pixels with low-confidence as in [46], which is a hard constraint, we use the filling rate to regularize the attention map as a soft constraint. Therefore, our attention map offers a continuous modulation of cross-entropy loss, rather than a binary decision. Attention maps have also been employed in [1, 59] where they are trained in a fully-supervised manner using pseudo segmentation ground truth. In contrast, our attention modulated loss offers a principled way of handling label noise present in segmentation ground truth derived from bounding box annotations. We validate our approach with quantitative and qualitative comparisons with [46] in Sect. 4.

3 Proposed Approach

In this section, we describe our Box2Seg algorithm in detail. We discuss our pipeline and loss functions that allow us to learn a per-image, class-specific attention map as well as pixel embeddings.

3.1 Feed Forward Network Architecture

Let $I \in \mathbb{R}^{n \times 3}$ denote a 3-channel input color image with n pixels. Our segmentation model $\mathcal{S}$ is a fully-Convolutional Neural Network (CNN) which takes I as input and produces three outputs: (i) the segmentation output probabilities $y \in \mathbb{P}^{m \times (L+1)}$, $\mathbb{P} \in [0, 1]$ (ii) the attention map $\alpha \in \mathbb{P}^{m \times L}$, and (iii) the pixel embeddings $\beta \in \mathbb{R}^{m \times d}$. Here d denotes the size of the pixel embeddings and m denotes the spatial resolution of the outputs. Each pixel in the output image can assume one of the $L + 1$ labels (L object categories and the background class). The ground truth bounding boxes are denoted by $B_{box} \in \mathbb{R}^{K \times 5}$, where K is

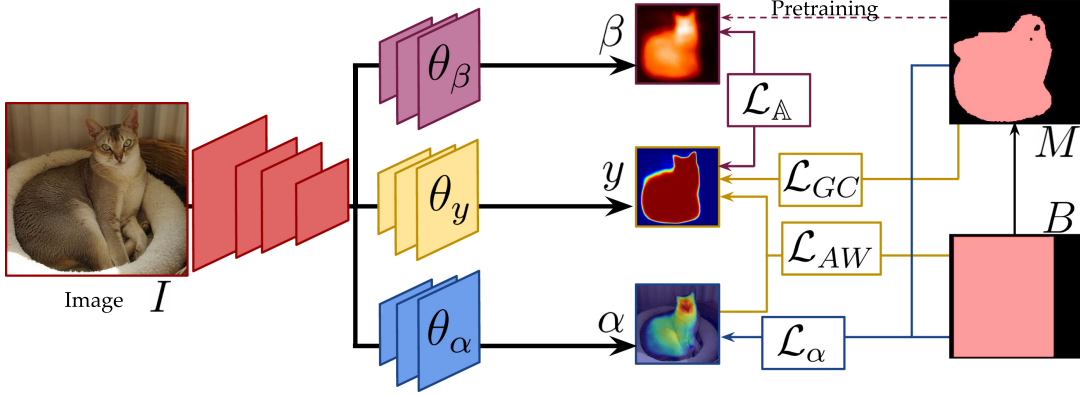


Fig. 3. Overview of our weakly supervised Box2Seg training pipeline. Our segmentation model is a three branch CNN which outputs segmentation probabilities y , per-class spatial attention maps α , and pixel embeddings β . B denotes segmentation masks defined using bounding box annotations. GrabCut is applied to B to obtain refined segmentation masks M . We use B and M to supervise the three outputs of the model. Pixel embeddings capture long-range pairwise relationships and the attention map refines the segmentation output y by reducing the effect of label noise. At test time we only use the branch producing the segmentation probabilities y , and discard the other branches that output α, β .

the number of bounding boxes comprising of 4 coordinates and a class label. To simplify the notation, we denote by $B \in \mathbb{R}^{m \times (L+1)}$ the box-segmentation tensor obtained by setting all pixels inside a bounding box to 1 for the corresponding class label (channel). In the case two boxes overlap, we assign 1 to the class corresponding to the smaller bounding box (assuming that the smaller box is in the front). Note that this assumption may not always be true and can result in incorrect label assignments for B (See eg. Fig. 8).

Similar to prior works [37, 41, 46], we generate pseudo ground-truth segmentation maps $M \in \mathbb{R}^{m \times (L+1)}$ by applying classical graph-based unsupervised segmentation approach (Grabcut [44]) on each bounding box in our training dataset. The segmentation masks obtained by classical methods, albeit noisy and imprecise, provide a good prior for training deep learning models.

Figure 3 gives an overview of our approach. Training involves passing the input image I through a common feature encoder that feeds into the three branches of network $\mathcal{S}$ to produce y, α , and β , as follows:

$$y = \text{Softmax}(\mathcal{S}(I, \theta_y)); \quad \alpha = \sigma(\mathcal{S}(I, \theta_\alpha)); \quad \beta = \mathcal{S}(I, \theta_\beta),$$

where $\theta_y, \theta_\alpha, \theta_\beta$ denote the parameters of the model $\mathcal{S}$ for the respective branches. Softmax denotes the softmax over all the classes and σ denotes the sigmoid activation function. Note that the segmentation output probabilities, y , sums up to 1 for each pixel across classes due to softmax. However, the activation maps, α , use sigmoid output, making them independent for each class. Note

that our model does not use additional parameters compared to any baselines we compare against in Sect. 4 as we discard the branches producing α, β at test time.

3.2 Box and GrabCut Based Losses

We use the box-segmentation tensor B to train a simple baseline by minimizing the following cross-entropy loss:

$$\mathcal{L}_{box} = -\frac{1}{m} \sum_{c=0}^L \sum_{i=1}^m B(i, c) \log(y(i, c)). \quad (1)$$

Similar to previous works, we use the GrabCut outputs M obtained from the bounding boxes to define another baseline by minimizing the following cross-entropy loss:

$$\mathcal{L}_{GC} = -\frac{1}{m} \sum_{c=0}^L \sum_{i=1}^m M(i, c) \log(y(i, c)). \quad (2)$$

Since the GrabCut algorithm provides reasonable segmentation outputs (Sect. 4), we use $\mathcal{L}_{GC}$ in addition to our loss functions described in the following sections.

3.3 Attention Weighted Segmentation Loss

Our novel attention modulated cross-entropy loss considers bounding box annotations as *noisy labels* for the foreground object. Note that since the bounding box is a super-set of the actual object segmentation mask, the label noise is one-sided: foreground labels are incorrectly assigned to background pixels, but no true foreground labels are missing. Additionally, pixels outside all the bounding boxes can be considered *definite background* and do not have any label noise. Since supervised segmentation training typically considers all labels as correct, in the presence of label noise, erroneous gradients can be back propagated during training. At pixels close to bounding box and object boundaries, the network gets conflicting information about the foreground/background labels at similar pixels. This is the reason for the worse performance of baseline trained with box annotations only (Table 3).

Attention on Foreground Objects: We propose to modulate the per-pixel cross-entropy loss using the predicted attention map from the network by minimizing

$$\mathcal{L}_{fg} = \frac{-1}{\sum_i^m B(i, c)} \sum_{c=1}^L \sum_{i=1}^m \alpha(i, c) B(i, c) \log(y(i, c)). \quad (3)$$

Note that the attention map has same spatial resolution as the segmentation output and is class-specific. The attention weighted loss is only defined for the

L foreground classes. In addition, the loss is normalized with the size of the bounding box, to give similar weighting to each class.

Background Loss: Since the pixels outside all the bounding boxes can be considered as definite background, the background loss is defined as

$$\mathcal{L}_{bg} = -\frac{\sum_i^m B(i, 0) \log(y(i, 0))}{\sum_i^m B(i, 0)}, \quad (4)$$

where 0 denotes the background class. We define the **Attention Weighted Loss** (AWL) as

$$\mathcal{L}_{AW} = \mathcal{L}_{fg} + \mathcal{L}_{bg}. \quad (5)$$

Attention Map Regularization: Without any regularization on the attention maps, the network can minimize $\mathcal{L}_{fg}$ by predicting all $\alpha(i, c) = 0$. To prevent this, we compared two approaches to regularize the attention map. The first approach regularizes the attention mask using ground truth bounding boxes (similar to [46]), by minimizing the $\mathcal{L}_2$ loss between the attention maps and the bounding boxes.

$$\mathcal{L}_\alpha^{bbox} = \sum_{i=1}^m \sum_{c=1}^L \| B(i, c) - \alpha(i, c) \|^2. \quad (6)$$

The second approach regularizes the attention mask using fill-ratios obtained from GrabCut outputs M as follows. Let η_c denote the per class, per image, filling rate defined as the proportion of pixels in the pseudo ground-truth M compared to its corresponding bounding box.

$$\eta_c = \frac{\sum_i^m M(i, c)}{\sum_i^m B(i, c)}. \quad (7)$$

Similarly, the predicted fill rate of the attention map is computed as

$$\eta'_c = \frac{\sum_i^m B(i, c) \alpha(i, c)}{\sum_i^m B(i, c)}. \quad (8)$$

We use a margin loss to ensure that the predicted fill rate is at least a factor γ of the fill rate obtained using M . $\mathcal{L}_\alpha^{fr} = \max(0, \gamma\eta_c - \eta'_c)$, where $\gamma \in [0, 1]$ is a hyper-parameter which is set to 0.7 in our experiments. Thus, the predicted fill rate (Eqn. 8) is allowed to vary between $\gamma\eta_c$ and 1. This enforces a soft constraint that the attention map should allow propagation of loss from at least 70% of η_c pixels inside the bounding box. Using $\mathcal{L}_\alpha^{bbox}$ for regularization forces the attention mask to take the shape of the bounding box. Thus, it is prone to include background pixels in attention map. Using $\mathcal{L}_\alpha^{fr}$ provides a softer constraint and gives better results in our experiments. Figure 4 shows some qualitative examples where attention map is able to focus on foreground pixels despite errors in the underlying GrabCut segmentations, allowing Box2Seg to be robust to label noise.

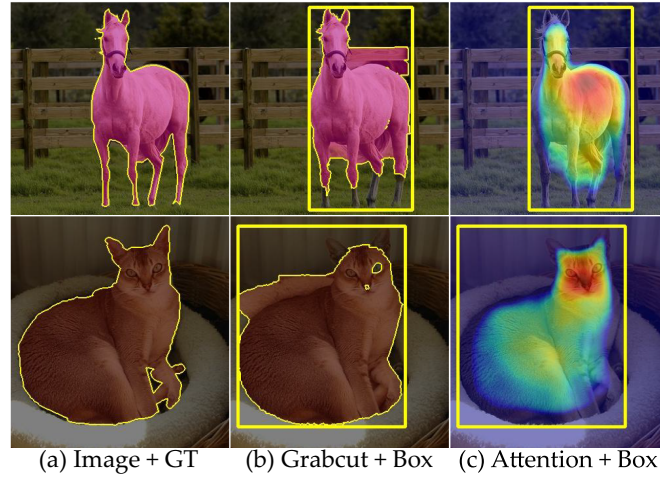


Fig. 4. Visualizing attention maps produced by our network. (a) Input images with overlaid ground truth masks. (b) Grabcut output with ground-truth bounding box. (c) Predicted attention map. Note that while Grabcut output is erroneous and often includes background pixels, our attention maps are concentrated on the objects of interest.

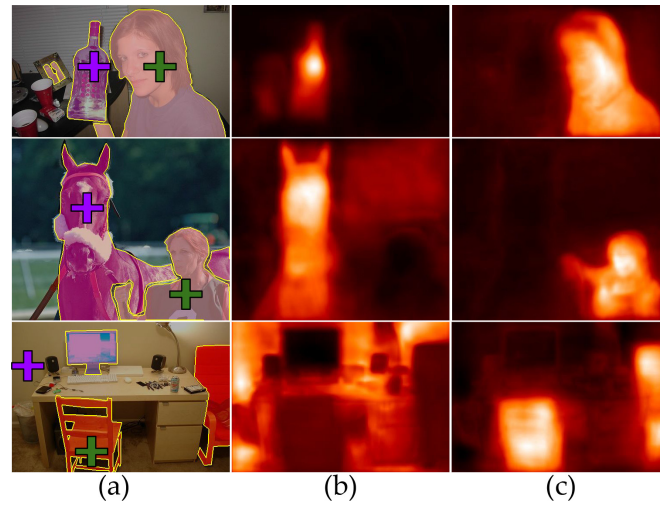


Fig. 5. Visualization of affinities produced by our network. (a) Input image with overlaid ground truth masks and two pixels marked with **purple** and **green** + signs. (b,c) Affinity heatmaps (Eqn. 9) w.r.t. the **purple** and **green** pixels, resp. (Color figure online)

3.4 Discriminative Feature Learning

We now describe training of our pixel embeddings β which capture long-range pairwise relationships between different pixels. Pixel embeddings can be denoted as $\beta = \{\beta_i\}$, where β_i is the d -dimensional feature for the i^{th} pixel. Affinity between embeddings at pixel i and j is given by its normalized dot product

$$\mathbb{A}(i, j) = \beta_i \cdot \beta_j = \frac{\beta_j^T \beta_i}{\|\beta_j\| \|\beta_i\|}. \quad (9)$$

Intuitively, we want to achieve high affinity between feature vectors of two pixels that belong to the same class, while ensuring low affinity between features of two different classes. Similarly, a background pixel should have low affinity with respect to another pixel that belongs to one of the L foreground classes. To achieve this, we define a novel loss function on pairs of pixels (i, j) , such that it encourages the pixel affinities to align with the predicted segmentation probabilities, as follows,

$$\mathcal{L}_{\mathbb{A}} = \sum_{i, j} (\mathbb{A}(i, j) - y_j^T y_i)^2. \quad (10)$$

However, training affinity matrices requires large amount of memory. To avoid creating large affinity matrices of size $m \times m$, we randomly sample a small fraction of pixel embeddings equally from each class to compute this loss. Figure 5 shows the affinity maps computed from our class discriminative embeddings.

3.5 Training Box2Seg

Our approach optimizes the following loss function:

$$\mathcal{L} = \mathcal{L}_{\mathcal{GC}} + \lambda_{AW} \mathcal{L}_{AW} + \lambda_{\alpha} \mathcal{L}_{\alpha} + \lambda_{\mathbb{A}} \mathcal{L}_{\mathbb{A}}, \quad (11)$$

where $\mathcal{L}_{\alpha}$ equals either $\mathcal{L}_{\alpha}^{bbx}$ or $\mathcal{L}_{\alpha}^{fr}$. λ_{AW} , λ_{α} and $\lambda_{\mathbb{A}}$ are weights applied to the individual losses.

4 Experiments and Results

We evaluate the performance of our approach on PASCAL VOC 2012 [13] dataset. Our ablation studies provide insights into our design choices. Finally, we demonstrate that our method provides a better initialization than Imagenet pretraining for the task of semi-supervised segmentation.

4.1 Implementation Details

Our segmentation network architecture is similar to UPerNet [51] where the encoder backbone is ResNet-101 [16], and decoders consist of 2 convolutional layers. We employ the ResNet-101 backbone to ensure fair comparison with the three most recent SOTA works, SDI [23], Li *et al.* [28], and BCM [46] as well as 4 other recent methods [27, 47–49] in Table 1. We have three decoders, one each for the y , α , and β branches. The final results are spatially down-sampled by a factor of 4, i.e. $m = n/16$. We start with ImageNet pretrained [12] weights to initialize our encoder.

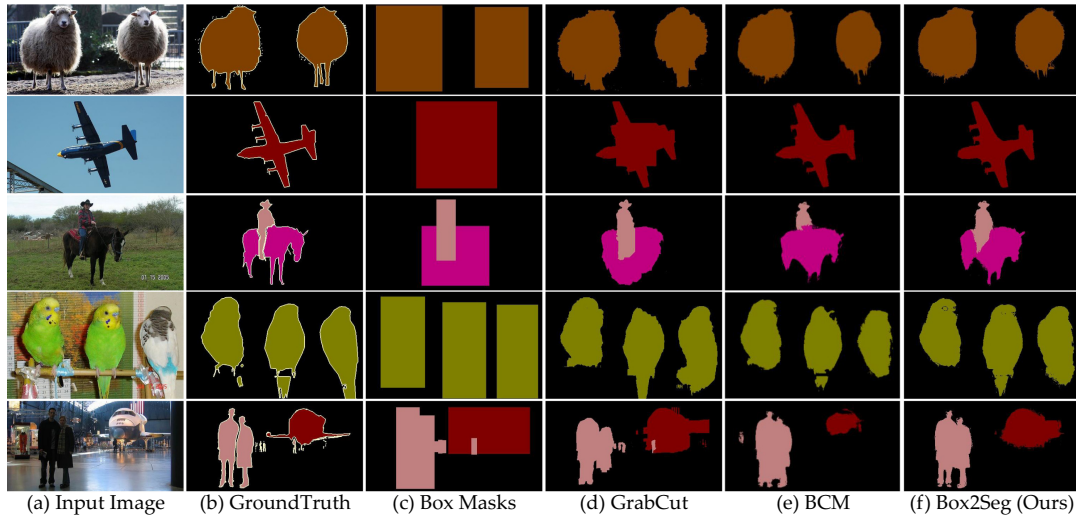


Fig. 6. Comparison of our segmentation results with those of BCM [46].

We train our network in two stages. First, we pre-train the feature representations using the affinity loss by randomly sampling 10% of pixel pairs using the Grabcut outputs M . This is done by minimizing Eqn. 10 using M_i in place of y_i . Note that the pretraining phase is meant to only serve the task of weight initialization for our discriminative feature learning. Our final feature representations are robust to noise in the Grabcut outputs since they are trained eventually to agree with our predictions using Eqn. 10. After the pretraining phase, we enable the decoder branch to also output y and α and train the entire network end-to-end to optimize the loss function in Eqn. 11. After doing a grid search of hyper-parameters on a held out validation set, λ_{AW} , λ_α and $\lambda_{\mathbb{A}}$ are set to 10, 1, and 1 respectively. We use Stochastic Gradient Descent to train our models for 40 epochs with an initial learning rate = $1e-4$, momentum = 0.9, weight decay = $5e^{-4}$ and a polynomially decaying learning rate as in [7]. Our implementation uses PyTorch [38] and is trained on Nvidia’s TitanX GPUs. Note that at test time we discard the decoders yielding α, β , therefore our method uses no additional parameters compared to any of our baselines in the following sections. Our implementation will be available at www.github.com/vivkul/Box2Seg.

4.2 Quantitative and Qualitative Evaluation

PASCAL VOC 2012 is one of the gold standard benchmarks for semantic segmentation. Following [9, 11, 32], we use the augmented annotation set [14] consisting of 10582 training and 1449 validation images. Performance is measured using the mIoU metric on the validation set.

We compare the accuracy of our algorithm to the SOTA weakly supervised segmentation methods on VOC validation set in Table 1. Our approach shows a large improvement over previous methods ([11, 23, 28, 37, 46]), some of which have also used VGG-16 backbones for feature representation. Li *et al.* [28] report

Table 1. Comparison of Box2Seg to previous weakly supervised semantic segmentation methods on PASCAL VOC validation set. C=dense-CRF post processing.

Method	Annotations	Backbone	MIoU
GrabCut-NoTrain-GT	Box	-	71.6
GrabCut-NoTrain-Det	Box	-	68.5
SSNet [52]	Image-level	DenseNet-169	63.3
F2FA [27]	Image-level	ResNet-101	66.5
ScribbleSup (C) [29]	Scribble	VGG-16	63.1
NormalCut [47]	Scribble	ResNet-101	72.8
BPG [49]	Scribble	ResNet-101	73.2
KernelCut [48]	Scribble	ResNet-101	73.0
WSSL (C) [37]	Box	VGG-16	60.6
BoxSup (C) [11]	Box	VGG-16	62.0
SDI (C) [23]	Box	ResNet-101	69.4
BCM (C) [46]	Box	ResNet-101	70.2
Li <i>et al.</i> (C) [28]	Box	ResNet-101	74.3
Box2Seg	Box	ResNet-101	74.9
Box2Seg (C)	Box	ResNet-101	76.4

an mIoU of 74.3%, using bounding boxes as supervision and dense CRF post-processing. Our method yields an mIoU of 74.9% without dense CRF, and 76.4% mIoU after dense CRF post processing resulting in an improvement of 2.1%. Box2Seg also outperforms recent weakly supervised methods that use image level labels [27, 52] or scribbles [29, 47–49]. Per-category results comparing Box2Seg against the SOTA methods are reported in Table 2.

Qualitative comparison of our results with BCM [46] is shown in Fig. 6. Our method is able to produce higher-quality object segmentations compared to BCM. It’s also robust to false segmentations in some cases, e.g. false detection closer to the left edge of the BCM result on the last row are suppressed by our method. Please refer to Fig. 1 for additional examples of segmentations using our approach. We also show some failure cases of our approach in Fig. 8.

Table 2. Per-class results on PASCAL VOC 2012 Validation set. We compare our *Box2Seg* (with and without denseCRF) with those of the previous state-of-the-art methods. Please note that Li *et al.* (CRF) results require COCO annotations. Per-class results from [46] for their ResNet-101 model are not available.

Method	Bkg	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Mbike	Person	Plant	Sheep	Sofa	Train	Tv	MIoU
BCM VGG (CRF) [46]	89.8	68.3	27.1	73.7	56.4	72.6	84.2	75.6	79.9	35.2	78.3	53.2	77.6	66.4	68.1	73.1	56.8	80.1	45.1	74.7	54.6	66.8
BCM ResNet (CRF) [46]	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	70.2
Li <i>et al.</i> (CRF) [28]	93.3	85.0	35.9	88.6	70.3	77.9	91.9	83.6	90.5	39.2	84.5	59.4	86.5	82.4	81.5	84.3	57.0	85.9	55.8	85.8	70.4	75.7
Box2Seg	92.5	66.5	31.7	78.9	65.5	83.4	90.4	86.7	86.0	55.1	81.8	59.9	80.5	74.1	76.0	75.7	65.3	85.1	72.5	87.8	77.7	74.9
Box2Seg (CRF)	93.3	72.4	33.0	84.2	64.9	83.5	90.9	86.7	88.7	57.2	83.6	62.5	82.6	76.8	77.0	77.8	63.3	87.2	75.1	88.3	74.1	76.4

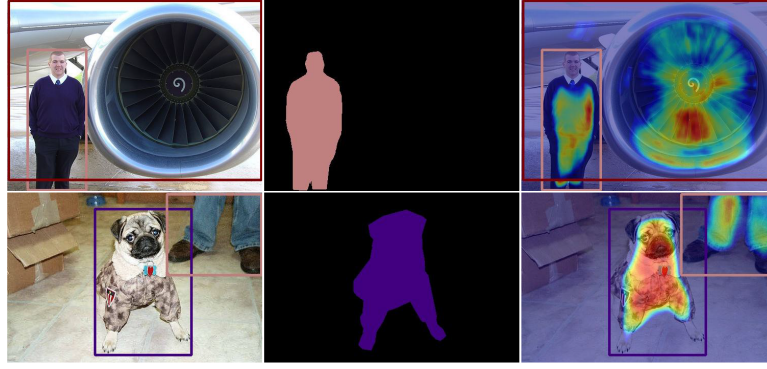


Fig. 7. (Left) Input images with overlaid bounding box labels. (Middle) Missing pixel-level annotations on foreground objects such as *Aeroplane* and *Person*. Note that the bounding box annotations are correct. (Right) Predicted attention map with boxes. Please refer to Sect. 4.4 for discussion on how our AWL can help in cases with conflicting ground truth during fully-supervised training.

Table 3. Ablation study showing the effect of our loss functions in improving the performance over baseline methods.

Method	$\mathcal{L}_{box}$	$\mathcal{L}_{GC}$	$\mathcal{L}_{AW}$	$\mathcal{L}_{\alpha}$	$\mathcal{L}_{\mathbb{A}}$	mIoU
Box	✓					59.3
GrabCut		✓				72.7
Affinity		✓			✓	73.9
AW-box		✓	✓	$\mathcal{L}_{\alpha}^{bbox}$		74.1
AW-fr		✓	✓	$\mathcal{L}_{\alpha}^{fr}$		74.6
Box2Seg		✓	✓	$\mathcal{L}_{\alpha}^{fr}$	✓	74.9

Accuracy of GrabCut (no training): We also evaluated the accuracy of GrabCut algorithm itself, without any training, against the segmentation ground-truth. Interestingly, GrabCut output on *ground truth* bounding boxes (*GrabCut-NoTrain-GT*) results in a strong weakly-supervised baseline with 71.6% mIoU. However, since the ground truth bounding boxes are not available at inference, a more practical baseline is to obtain bounding boxes on the validation set using an object detector (we used SNIPER [45]) and then run GrabCut on those. This baseline (*GrabCut-NoTrain-Det*) obtains 68.5% mIoU.

4.3 Ablation Studies

Table 3 shows the efficacy of our loss functions in improving the performance of our approach. The trivial *Box* baseline obtained by training the segmentation network with bounding box supervision ($\mathcal{L}_{box}$ loss only) results in a low mIoU of 59.3%, as expected. Training the segmentation network with *GrabCut* masks ($\mathcal{L}_{GC}$ loss only) without affinity or attention losses resulted in 72.7% mIoU. Introducing our feature embeddings to the pipeline (Affinity) improves the mIoU

to 73.9%. Our novel AWL terms ($\lambda_{AW}\mathcal{L}_{AW} + \lambda_{\alpha}\mathcal{L}_{\alpha}$) significantly improve the mIoU to 74.6% with filling-rate regularization (*AW-fr*) (more about AWL in Sect. 4.4). Finally, combining all losses (Eqn 11), *Box2Seg* obtains 74.9% mIoU.

4.4 Attention Weighted Loss in the Fully-Supervised Setting

We demonstrate that AWL boosts segmentation accuracy in fully-supervised case if we have disagreements between bounding-box and per-pixel annotations.

During our analysis, we found that roughly 10% of training images in PASCAL VOC dataset [13, 14] have disagreements between the bounding box and pixel-level annotations. Figure 7 shows few examples, where pixel-level annotation for object categories are missing, but their corresponding bounding box labels are correctly provided. Due to incorrect pixel-level annotations, fully supervised training would back-propagate erroneous gradients on these pixels. Since our novel AWL ($\mathcal{L}_{AW}$ in Sect. 3.3) is effective in dealing with label noise for weakly supervised networks, we analyze if it can further improve the performance of a fully supervised network also. In cases where conflicting sources of ground truth exist (as in Fig. 7), AWL can allow correct gradients to propagate back due to correct bounding box annotations (see the predicted attention maps in Fig. 7), thereby reducing the effect of incorrect pixel-level annotations.

Table 4. Improvements in segmentation accuracy on the PASCAL VOC 2012 validation set in the fully supervised setting using AWL.

Method	CE Loss	+ $\mathcal{L}_{AW}$	Δ
Supervised baseline	73.6	75.1	+1.5

Table 4 shows the improvement obtained by adding AWL to the fully *Supervised baseline*. Adding the AWL (+ $\mathcal{L}_{AW}$) to the fully-*Supervised* baseline improved the segmentation mIoU by 1.5%. Thus, AWL is effective at improving segmentation accuracy, both in the weakly- and fully- supervised settings.

4.5 Semi-supervised Semantic Segmentation

Weakly supervised trained method can naturally serve as a starting point for semi-supervised segmentation. To study this, we fine tuned our Box2Seg model using different amount of pixel-level annotations. We observe significant improvements in accuracy, even with small amount of supervision as shown in Table 5. For comparison, we show the result of semi-supervised fine tuned BCM [46] model trained with 1464 images (13.8% of the data) and another fully supervised baseline (*DeepLab*) using ImageNet as the initialization. Our weakly supervised baseline results improved from 74.9% mIoU to 83.1% with just 10% of supervised data. Therefore, our Box2Seg model can serve as a good starting point and provides better results compared to ImageNet based initialization.

Table 5. Semi-supervised segmentation using Box2Seg model as initialization. ($S=\tau\%$) implies $\tau\%$ images with pixel annotations are used for semi-supervised fine-tuning. C=dense-CRF post processing. †: ResNet-101 backbone (refer [46]).

Method	MIoU
BCM [46] † ($S=0\%$) (C)	70.2
BCM [46] † ($S=13.8\%$) (C)	71.6
DeepLab † ($S=100\%$) (C)	74.5
Box2Seg ($S=0\%$)	74.9
Box2Seg ($S=5\%$)	78.7
Box2Seg ($S=10\%$)	83.1
Box2Seg ($S=100\%$)	86.4

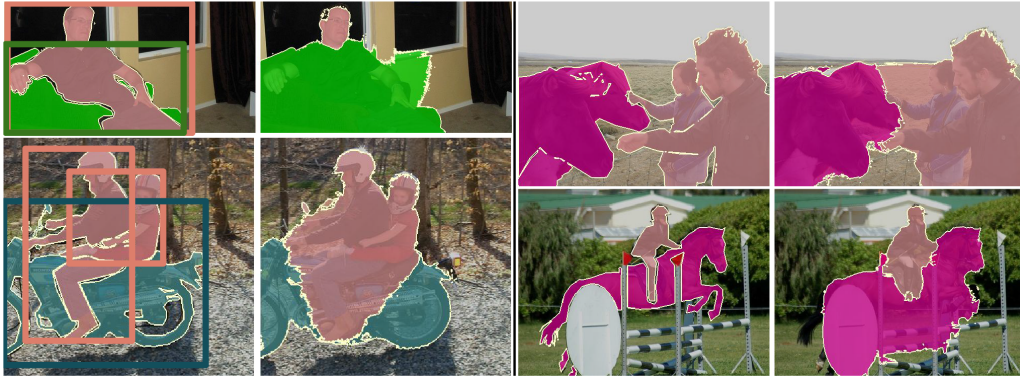


Fig. 8. Failure Cases. (Left) Incorrect label assignment on box annotations B can happen when the smaller bounding box is physically behind the larger bounding box. (Right) Predicted segmentation bleeds into background.

5 Conclusions

In this work, we proposed a pipeline for training a weakly supervised semantic segmentation method from bounding box annotations. We showed that bounding box annotations can be treated as noisy labels for foreground objects and proposed a novel attention weighted loss to reduce the effect of erroneous gradients due to incorrect labels. We also proposed pixel embeddings to capture global context via long-range pairwise interactions. We showed qualitative improvements over the previous SOTA on the PASCAL VOC semantic segmentation benchmark and pushed the mIoU metric forward by 2.1%. Interestingly, fully supervised methods can also benefit from attention weighted loss in the presence of exhaustive bounding box annotations but missing pixel-level annotations. Future work would involve using an edge detector as a cue to learn class boundaries, and also extending our method to benefit from image-level supervision.

Acknowledgements. V. Kulharia worked as an intern at Amazon Lab126 and continued this work at Oxford University. V Kulharia’s DPhil is funded by the Toyota Research Institute grant.

References

1. Ahn, J., Cho, S., Kwak, S.: Weakly supervised learning of instance segmentation with inter-pixel relations. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2209–2218 (2019)
2. Ahn, J., Kwak, S.: Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018)
3. Arandjelović, R., Zisserman, A.: Object discovery with a copy-pasting gan. *arXiv preprint [arXiv:1905.11369](https://arxiv.org/abs/1905.11369)* (2019)
4. Bearman, A., Russakovsky, O., Ferrari, V., Fei-Fei, L.: What’s the point: semantic segmentation with point supervision. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016*. LNCS, vol. 9911, pp. 549–565. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_34
5. Chandra, S., Usunier, N., Kokkinos, I.: Dense and low-rank gaussian crfs using deep embeddings. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5103–5112 (2017)
6. Chaudhry, A., Dokania, P.K., Torr, P.H.: Discovering class-specific pixels for weakly-supervised semantic segmentation. In: *British Machine Vision Conference (BMVC)* (2017)
7. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint [arXiv:1412.7062](https://arxiv.org/abs/1412.7062)* (2014)
8. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. [arXiv:1606.00915](https://arxiv.org/abs/1606.00915) (2016)
9. Chen, L., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. *CoRR* abs/1706.05587 (2017). <http://arxiv.org/abs/1706.05587>
10. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: *European Conference on Computer Vision (ECCV)*, pp. 801–818 (2018)
11. Dai, J., He, K., Sun, J.: Boxsup: exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: *International Conference on Computer Vision (ICCV)*, pp. 1635–1643 (2015)
12. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009)
13. Everingham, M., Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis. (IJCV)* **88**(2), 303–338 (2010). <https://doi.org/10.1007/s11263-009-0275-4>
14. Hariharan, B., Arbelaez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: *International Conference on Computer Vision (ICCV)* (2011)
15. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: *International Conference on Computer Vision (ICCV)* (2017)

16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
17. Hou, Q., Massiceti, D., Dokania, P.K., Wei, Y., Cheng, M.M., Torr, P.H.S.: Bottom-up top-down cues for weakly-supervised semantic segmentation. In: Pelillo, M., Hancock, E. (eds.) EMMCVPR 2017. LNCS, vol. 10746, pp. 263–277. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78199-0_18
18. Hsu, K.J., Lin, Y.Y., Chuang, Y.Y.: Deepco3: deep instance co-segmentation by co-peak search and co-saliency detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
19. Hu, R., Dollár, P., He, K., Darrell, T., Girshick, R.: Learning to segment every thing. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4233–4241 (2018)
20. Huang, Z., Wang, X., Wang, J., Liu, W., Wang, J.: Weakly-supervised semantic segmentation network with deep seeded region growing. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 7014–7023 (2018)
21. Hung, W.C., Tsai, Y.H., Liou, Y.T., Lin, Y.Y., Yang, M.H.: Adversarial learning for semi-supervised semantic segmentation. In: British Machine Vision Conference (BMVC) (2018)
22. Ke, T.W., Hwang, J.J., Liu, Z., Yu, S.X.: Adaptive affinity fields for semantic segmentation. In: European Conference on Computer Vision (ECCV) (2018)
23. Khoreva, A., Benenson, R., Hosang, J., Hein, M., Schiele, B.: Simple does it: weakly supervised instance and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 876–885 (2017)
24. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. In: Neural Information Processing Systems (NIPS) (2011)
25. Kuznetsova, A., et al.: The open images dataset v4: unified image classification, object detection, and visual relationship detection at scale. arXiv preprint [arXiv:1811.00982](https://arxiv.org/abs/1811.00982) (2018)
26. Kwak, S., Hong, S., Han, B.: Weakly supervised semantic segmentation using superpixel pooling network. In: AAAI Conference on Artificial Intelligence (2017)
27. Lee, J., Kim, E., Lee, S., Lee, J., Yoon, S.: Frame-to-frame aggregation of active regions in web videos for weakly supervised semantic segmentation. In: International Conference on Computer Vision (ICCV), October 2019
28. Li, Q., Arnab, A., Torr, P.H.: Weakly-and semi-supervised panoptic segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV 18), pp. 102–118 (2018)
29. Lin, D., Dai, J., Jia, J., He, K., Sun, J.: Scribblesup: scribble-supervised convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3159–3167 (2016)
30. Lin, T.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
31. Liu, S., De Mello, S., Gu, J., Zhong, G., Yang, M.H., Kautz, J.: Learning affinity via spatial propagation networks. In: Neural Information Processing Systems (NIPS) (2017)
32. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440 (2015)
33. Maire, M., Narihira, T., Yu, S.X.: Affinity cnn: learning pixel-centric pairwise relations for figure/ground embedding. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

34. Min, S., Chen, X., Zha, Z.J., Wu, F., Zhang, Y.: A two-stream mutual attention network for semi-supervised biomedical segmentation with noisy labels (2018)
35. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: arXiv preprint [arXiv:1505.04366](https://arxiv.org/abs/1505.04366) (2015)
36. Papadopoulos, D.P., Uijlings, J.R., Keller, F., Ferrari, V.: Extreme clicking for efficient object annotation. In: IEEE International Conference on Computer Vision (CVPR), pp. 4930–4939 (2017)
37. Papandreou, G., Chen, L.C., Murphy, K.P., Yuille, A.L.: Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1742–1750 (2015)
38. Paszke, A., et al.: Automatic differentiation in pytorch (2017)
39. Petit, O., Thome, N., Charnoz, A., Hostettler, A., Soler, L.: Handling missing annotations for semantic segmentation with deep convNets. In: Stoyanov, D., et al. (eds.) DLMIA/ML-CDS -2018. LNCS, vol. 11045, pp. 20–28. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00889-5_3
40. Pont-Tuset, J., Arbelaez, P., Barron, J.T., Marques, F., Malik, J.: Multiscale combinatorial grouping for image segmentation and object proposal generation. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **39**(1), 128–140 (2016)
41. Rajchl, M., et al.: Deepcut: object segmentation from bounding box annotations using convolutional neural networks. IEEE Trans. Med. Imaging **36**(2), 674–683 (2016)
42. Redondo-Cabrera, C., Baptista-Ríos, M., López-Sastre, R.J.: Learning to exploit the prior network knowledge for weakly supervised semantic segmentation. IEEE Trans. Image Process. **28**(7), 3649–3661 (2019)
43. Richter, S.R., Hayder, Z., Koltun, V.: Playing for benchmarks. In: International Conference on Computer Vision (ICCV) (2017)
44. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. (TOG) **23**(3), 309–314 (2004)
45. Singh, B., Najibi, M., Davis, L.S.: SNIPER: efficient multi-scale training. In: Neural Information Processing Systems (NIPS) (2018)
46. Song, C., Huang, Y., Ouyang, W., Wang, L.: Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3136–3145 (2019)
47. Tang, M., Djelouah, A., Perazzi, F., Boykov, Y., Schroers, C.: Normalized cut loss for weakly-supervised cnn segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1818–1827 (2018)
48. Tang, M., Perazzi, F., Djelouah, A., Ben Ayed, I., Schroers, C., Boykov, Y.: On regularized losses for weakly-supervised cnn segmentation. In: European Conference on Computer Vision (ECCV), pp. 507–522 (2018)
49. Wang, B., et al.: Boundary perception guidance: a scribble-supervised semantic segmentation approach. In: International Joint Conference on Artificial Intelligence (IJCAI) (2019)
50. Wang, X., You, S., Li, X., Ma, H.: Weakly-supervised semantic segmentation by iteratively mining common object features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1354–1362 (2018)
51. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: European Conference on Computer Vision (ECCV), pp. 418–434 (2018)

52. Zeng, Y., Zhuge, Y., Lu, H., Zhang, L.: Joint learning of saliency detection and weakly supervised semantic segmentation. *Int. Conf. Comput. Vis. (ICCV)* **3**(11), 12 (2019)
53. Zhang, H., et al.: Context encoding for semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018
54. Zhang, H., Zhang, H., Wang, C., Xie, J.: Co-occurrent features in semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019)
55. Zhao, X., Liang, S., Wei, Y.: Pseudo mask augmented object detection. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4061–4070 (2018)
56. Zheng, S., et al.: Conditional random fields as recurrent neural networks. In: *International Conference on Computer Vision (ICCV)* (2015)
57. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2921–2929 (2016)
58. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017)
59. Zhu, Y., Zhou, Y., Xu, H., Ye, Q., Doermann, D., Jiao, J.: Learning instance activation maps for weakly supervised instance segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3116–3125 (2019)

D.3 Calibrating Deep Neural Networks using Focal Loss

Contributions:

- The idea of using focal loss was first suggested by Puneet, which was then extensively discussed and studied by Viveka, Jishnu, and Amartya.
- Jishnu and Viveka worked equally on almost all the experiment design and analysis.
- Viveka Kulharia led the theoretical analysis with Amartya Sanyal and Jishnu Mukhoti.
- Puneet led the project, the weekly discussions, and supervised it along with active participation from Stuart Golodetz and Philip H. S. Torr.

Calibrating Deep Neural Networks using Focal Loss

Jishnu Mukhoti*
University of Oxford
FiveAI Ltd.

Viveka Kulharia*
University of Oxford

Amartya Sanyal
University of Oxford
The Alan Turing Institute

Stuart Golodetz
FiveAI Ltd.

Philip H. S. Torr
University of Oxford
FiveAI Ltd.

Puneet K. Dokania
University of Oxford
FiveAI Ltd.

Abstract

Miscalibration – a mismatch between a model’s confidence and its correctness – of Deep Neural Networks (DNNs) makes their predictions hard to rely on. Ideally, we want networks to be accurate, calibrated and confident. We show that, as opposed to the standard cross-entropy loss, focal loss [19] allows us to learn models that are already very well calibrated. When combined with temperature scaling, whilst preserving accuracy, it yields state-of-the-art calibrated models. We provide a thorough analysis of the factors causing miscalibration, and use the insights we glean from this to justify the empirically excellent performance of focal loss. To facilitate the use of focal loss in practice, we also provide a principled approach to automatically select the hyperparameter involved in the loss function. We perform extensive experiments on a variety of computer vision and NLP datasets, and with a wide variety of network architectures, and show that our approach achieves state-of-the-art calibration without compromising on accuracy in almost all cases. Code is available at https://github.com/torrvision/focal_calibration.

1 Introduction

Deep neural networks have dominated computer vision and machine learning in recent years, and this has led to their widespread deployment in real-world systems [2, 3, 11, 12, 38]. However, many current multi-class classification networks in particular are poorly calibrated, in the sense that the probability values that they associate with the class labels they predict overestimate the likelihoods of those class labels being correct in the real world. This is a major problem, since if networks are routinely overconfident, then downstream components cannot trust their predictions. The underlying cause is hypothesised to be that these networks’ high capacity leaves them vulnerable to overfitting on the negative log-likelihood (NLL) loss they conventionally use during training [7].

Given the importance of this problem, numerous suggestions for how to address it have been proposed. Much work has been inspired by approaches that were not originally formulated in a deep learning context, such as Platt scaling [30], histogram binning [40], isotonic regression [41], and Bayesian binning and averaging [22, 21]. As deep learning has become more dominant, however, various works have begun to directly target the calibration of deep networks. For example, Guo et al. [7] have popularised a modern variant of Platt scaling known as *temperature scaling*, which works by dividing a network’s logits by a scalar $T > 0$ (learnt on a validation subset) prior to performing softmax. Temperature scaling has the desirable property that it can improve the calibration of a network without in any way affecting its accuracy. However, whilst its simplicity and effectiveness

*Joint first authors, order decided by coin flip. Contact: {jishnu, viveka, puneet, phst}@robots.ox.ac.uk, amartya.sanyal@cs.ox.ac.uk, sgolodetz@gxstudios.net

have made it a popular network calibration method, it does have downsides. For example, whilst it scales the logits to reduce the network’s confidence in incorrect predictions, this also slightly reduces the network’s confidence in predictions that were correct [16]. Moreover, it is known that temperature scaling does not calibrate a model under data distribution shift [27].

By contrast, [16] initially eschew temperature scaling in favour of minimising a differentiable proxy for calibration error at training time, called Maximum Mean Calibration Error (MMCE), although they do later also use temperature scaling as a post-processing step to obtain better results than cross-entropy followed by temperature scaling [7]. Separately, [20] propose training models on cross-entropy loss with label smoothing instead of one-hot labels, and show that label smoothing has a very favourable effect on model calibration.

In this paper, we propose a technique for improving network calibration that works by replacing the cross-entropy loss conventionally used when training classification networks with the focal loss proposed by [19]. We observe that unlike cross-entropy, which minimises the KL divergence between the predicted (softmax) distribution and the target distribution (one-hot encoding in classification tasks) over classes, focal loss minimises a regularised KL divergence between these two distributions, which ensures minimisation of the KL divergence whilst *increasing the entropy* of the predicted distribution, thereby preventing the model from becoming overconfident. Since focal loss, as shown in §4, is dependent on a hyperparameter, γ , that needs to be cross-validated, we also provide a method for choosing γ automatically for each sample, and show that it outperforms all the baseline models.

The intuition behind using focal loss is to direct the network’s attention during training towards samples for which it is currently predicting a low probability for the correct class, since trying to reduce the NLL on samples for which it is already predicting a high probability for the correct class is liable to lead to NLL overfitting, and thereby miscalibration [7]. More formally, we show in §4 that focal loss can be seen as *implicitly* regularising the weights of the network during training by causing the gradient norms for confident samples to be lower than they would have been with cross-entropy, which we would expect to reduce overfitting and improve the network’s calibration.

Overall, we make the following contributions:

1. In §3, we study the link that [7] observed between miscalibration and NLL overfitting in detail, and show that the overfitting is associated with the predicted distributions for misclassified test samples becoming peakier as the optimiser tries to increase the magnitude of the network’s weights to reduce the training NLL.
2. In §4, we propose the use of focal loss for training better-calibrated networks, and provide both theoretical and empirical justifications for this approach. In addition, we provide a principled method for automatically choosing γ for each sample during training.
3. In §5, we show, via experiments on a variety of classification datasets and network architectures, that DNNs trained with focal loss are more calibrated than those trained with cross-entropy loss (both with and without label smoothing), MMCE or Brier loss [1]. Finally, we also make the interesting observation that whilst temperature scaling may not work for detecting out-of-distribution (OoD) samples, our approach can. We show that our approach is better at detecting out-of-distribution samples, taking CIFAR-10 as the in-distribution dataset, and SVHN and CIFAR-10-C as out-of-distribution datasets.

2 Problem Formulation

Let $D = \langle (\mathbf{x}_i, y_i) \rangle_{i=1}^N$ denote a dataset consisting of N samples from a joint distribution $\mathcal{D}(\mathcal{X}, \mathcal{Y})$, where for each sample i , $\mathbf{x}_i \in \mathcal{X}$ is the input and $y_i \in \mathcal{Y} = \{1, 2, \dots, K\}$ is the ground-truth class label. Let $\hat{p}_{i,y} = f_{\theta}(y|\mathbf{x}_i)$ be the probability that a neural network f with model parameters θ predicts for a class y on a given input $\mathbf{x}_i$. The class that f predicts for $\mathbf{x}_i$ is computed as $\hat{y}_i = \operatorname{argmax}_{y \in \mathcal{Y}} \hat{p}_{i,y}$, and the predicted confidence as $\hat{p}_i = \max_{y \in \mathcal{Y}} \hat{p}_{i,y}$. The network is said to be *perfectly calibrated* when, for each sample $(\mathbf{x}, y) \in D$, the confidence $\hat{p}$ is equal to the model accuracy $\mathbb{P}(\hat{y} = y|\hat{p})$, i.e. the probability that the predicted class is correct. For instance, of all the samples to which a perfectly calibrated neural network assigns a confidence of 0.8, 80% should be correctly predicted.

A popular metric used to measure model calibration is the *expected calibration error* (ECE) [22], defined as the expected absolute difference between the model’s confidence and its accuracy, i.e. $\mathbb{E}_{\hat{p}}[|\mathbb{P}(\hat{y} = y|\hat{p}) - \hat{p}|]$. Since we only have finite samples, the ECE cannot in practice be computed using this definition. Instead, we divide the interval $[0, 1]$ into M equispaced bins, where the i^{th} bin

is the interval $(\frac{i-1}{M}, \frac{i}{M}]$. Let B_i denote the set of samples with confidences belonging to the i^{th} bin. The accuracy A_i of this bin is computed as $A_i = \frac{1}{|B_i|} \sum_{j \in B_i} \mathbb{1}(\hat{y}_j = y_j)$, where $\mathbb{1}$ is the indicator function, and $\hat{y}_j$ and y_j are the predicted and ground-truth labels for the j^{th} sample. Similarly, the confidence C_i of the i^{th} bin is computed as $C_i = \frac{1}{|B_i|} \sum_{j \in B_i} \hat{p}_j$, i.e. C_i is the average confidence of all samples in the bin. The ECE can be approximated as a weighted average of the absolute difference between the accuracy and confidence of each bin: $\text{ECE} = \sum_{i=1}^M \frac{|B_i|}{N} |A_i - C_i|$.

A similar metric, the *maximum calibration error* (MCE) [22], is defined as the maximum absolute difference between the accuracy and confidence of each bin: $\text{MCE} = \max_{i \in \{1, \dots, M\}} |A_i - C_i|$.

AdaECE: One disadvantage of ECE is the uniform bin width. For a trained model, most of the samples lie within the highest confidence bins, and hence these bins dominate the value of the ECE. We thus also consider another metric, AdaECE (Adaptive ECE), for which bin sizes are calculated so as to evenly distribute samples between bins (similar to the adaptive binning procedure in [25]): $\text{AdaECE} = \sum_{i=1}^M \frac{|B_i|}{N} |A_i - C_i|$ s.t. $\forall i, j \cdot |B_i| = |B_j|$.

Classwise-ECE: The ECE metric only considers the probability of the predicted class, without considering the other scores in the softmax distribution. A stronger definition of calibration would require the probabilities of all the classes in the softmax distribution to be calibrated [14, 36, 39, 15]. This can be achieved with a simple classwise extension of the ECE metric: $\text{ClasswiseECE} = \frac{1}{K} \sum_{i=1}^M \sum_{j=1}^K \frac{|B_{i,j}|}{N} |A_{i,j} - C_{i,j}|$, where K is the number of classes, $B_{i,j}$ denotes the set of samples from the j^{th} class in the i^{th} bin, $A_{i,j} = \frac{1}{|B_{i,j}|} \sum_{k \in B_{i,j}} \mathbb{1}(j = y_k)$ and $C_{i,j} = \frac{1}{|B_{i,j}|} \sum_{k \in B_{i,j}} \hat{p}_{kj}$.

A common way of visualising calibration is to use a *reliability plot* [26], which plots the accuracies of the confidence bins as a bar chart (see Appendix Figure A.1). For a perfectly calibrated model, the accuracy for each bin matches the confidence, and hence all of the bars lie on the diagonal. By contrast, if most of the bars lie above the diagonal, the model is more accurate than it expects, and is under-confident, and if most of the bars lie below the diagonal, then it is over-confident.

3 What Causes Miscalibration?

We now discuss why high-capacity neural networks, despite achieving low classification errors on well-known datasets, tend to be miscalibrated. A key empirical observation made by [7] was that poor calibration of such networks appears to be linked to overfitting on the negative log-likelihood (NLL) during training. In this section, we further inspect this observation to provide new insights.

For the analysis, we train a ResNet-50 network on CIFAR-10 with state-of-the-art performance settings [31]. We use Stochastic Gradient Descent (SGD) with a mini-batch of size 128, momentum of 0.9, and learning rate schedule of $\{0.1, 0.01, 0.001\}$ for the first 150, next 100, and last 100 epochs, respectively. We minimise cross-entropy loss (a.k.a. NLL) $\mathcal{L}_c$, which, in a standard classification context, is $-\log \hat{p}_{i,y_i}$, where $\hat{p}_{i,y_i}$ is the probability assigned by the network to the correct class y_i for the i^{th} sample. Note that the NLL is minimised when for each training sample i , $\hat{p}_{i,y_i} = 1$, whereas the classification error is minimised when $\hat{p}_{i,y_i} > \hat{p}_{i,y}$ for all $y \neq y_i$. This indicates that even when the classification error is 0, the NLL can be positive, and the optimisation algorithm can still try to reduce it to 0 by further increasing the value of $\hat{p}_{i,y_i}$ for each sample (see Appendix A).

To study how miscalibration occurs during training, we plot the average NLL for the train and test sets at each training epoch in Figures 1(a) and 1(b). We also plot the average NLL and the entropy of the softmax distribution produced by the network for the correctly and incorrectly classified samples. In Figure 1(c), we plot the classification errors on the train and test sets, along with the test set ECE.

Curse of misclassified samples: Figures 1(a) and 1(b) show that although the average train NLL (for both correctly and incorrectly classified training samples) broadly decreases throughout training, after the 150th epoch (where the learning rate drops by a factor of 10), there is a marked rise in the average test NLL, indicating that the network starts to overfit on average NLL. This increase in average test NLL is caused only by the incorrectly classified samples, as the average NLL for the correctly classified samples continues to decrease even after the 150th epoch. We also observe that after epoch 150, the test set ECE rises, indicating that the network is becoming miscalibrated. This corroborates the observation in [7] that miscalibration and NLL overfitting are linked.

Peak at the wrong place: We further observe that the entropies of the softmax distributions for both the correctly and incorrectly classified *test* samples decrease throughout training (in other words, the

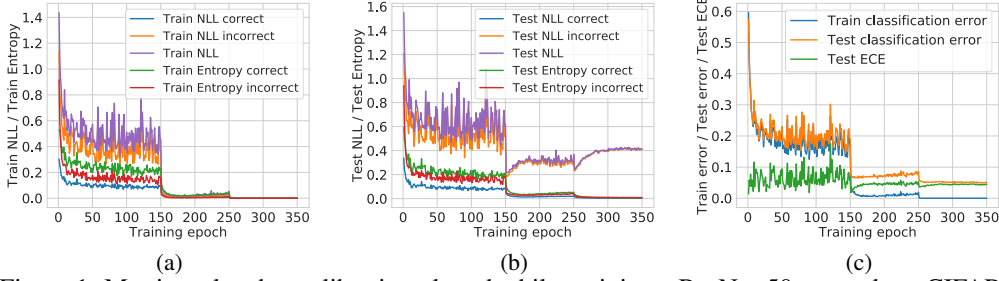


Figure 1: Metrics related to calibration plotted whilst training a ResNet-50 network on CIFAR-10.

distributions get peakier). This observation, coupled with the one we made above, indicates that *for the wrongly classified test samples, the network gradually becomes more and more confident about its incorrect predictions.*

Weight magnification: The increase in confidence of the network’s predictions can happen if the network increases the norm of its weights W to increase the magnitudes of the logits. In fact, cross-entropy loss is minimised when for each training sample i , $\hat{p}_{i,y_i} = 1$, which is possible only when $\|W\| \rightarrow \infty$. Cross-entropy loss thus inherently induces this tendency of weight magnification in neural network optimisation. The promising performance of weight decay [7] (regulating the norm of weights) on the calibration of neural networks can perhaps be explained using this. This increase in the network’s confidence during training is one of the key causes of miscalibration.

4 Improving Calibration using Focal Loss

As discussed in §3, overfitting on NLL, which is observed as the network grows more confident on all of its predictions irrespective of their correctness, is strongly related to poor calibration. One cause of this is that the cross-entropy objective minimises the difference between the softmax distribution and the ground-truth one-hot encoding over an entire mini-batch, irrespective of how well a network classifies individual samples in the mini-batch. In this work, we study an alternative loss function, popularly known as *focal loss* [19], that tackles this by weighting loss components generated from individual samples in a mini-batch by how well the model classifies them. For classification tasks where the target distribution is a one-hot encoding, it is defined as $\mathcal{L}_f = -(1 - \hat{p}_{i,y_i})^\gamma \log \hat{p}_{i,y_i}$, where γ is a user-defined hyperparameter².

Why might focal loss improve calibration? We know that cross-entropy forms an upper bound on the KL-divergence between the target distribution q and the predicted distribution $\hat{p}$, i.e. $\mathcal{L}_c \geq \text{KL}(q||\hat{p})$, so minimising cross-entropy results in minimising $\text{KL}(q||\hat{p})$. Interestingly, a general form of focal loss can be shown to be an upper bound on the regularised KL-divergence, where the regulariser is the negative entropy of the predicted distribution $\hat{p}$, and the regularisation parameter is γ , the hyperparameter of focal loss (a proof of this can be found in Appendix B):

$$\mathcal{L}_f \geq \text{KL}(q||\hat{p}) - \gamma \mathbb{H}[\hat{p}]. \quad (1)$$

The most interesting property of this upper bound is that it shows that replacing cross-entropy with focal loss has the effect of adding a maximum-entropy regulariser [29] to the implicit minimisation that was previously being performed. In other words, trying to minimise focal loss minimises the KL divergence between $\hat{p}$ and q , whilst simultaneously increasing the entropy of the predicted distribution $\hat{p}$. Note, in the case of ground truth with one-hot encoding, only the component of the entropy of $\hat{p}$ corresponding to the ground-truth index, $\gamma(-\hat{p}_{i,y_i} \log \hat{p}_{i,y_i})$, will be maximised (refer Appendix B). Encouraging the predicted distribution to have higher entropy can help avoid the overconfident predictions produced by DNNs (see the ‘Peak at the wrong place’ paragraph of §3), and thereby improve calibration.

Empirical observations: To analyse the behaviour of neural networks trained on focal loss, we use the same framework as mentioned above, and train four ResNet-50 networks on CIFAR-10, one using cross-entropy loss, and three using focal loss with $\gamma = 1, 2$ and 3. Figure 2(a) shows that the test NLL

²We note in passing that unlike cross-entropy loss, focal loss in its general form is not a proper loss function, as minimising it does not always lead to the predicted distribution $\hat{p}$ being equal to the target distribution q (see Appendix B for the relevant definition and a longer discussion). However, when q is a one-hot encoding (as in our case, and for most classification tasks), minimising focal loss does lead to $\hat{p}$ being equal to q .

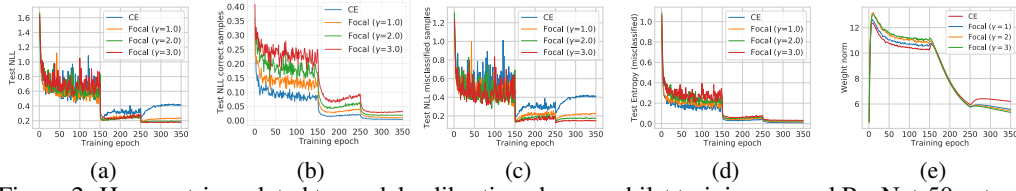


Figure 2: How metrics related to model calibration change whilst training several ResNet-50 networks on CIFAR-10, using either cross-entropy loss, or focal loss with γ set to 1, 2 or 3.

for the cross-entropy model significantly increases towards the end of training (before saturating), whereas the NLLs for the focal loss models remain low. To better understand this, we analyse the behaviour of these models for correctly and incorrectly classified samples. Figure 2(b) shows that even though the NLLs for the correctly classified samples broadly-speaking decrease over the course of training for all the models, the NLLs for the focal loss models remain consistently higher than that for the cross-entropy model throughout training, implying that the focal loss models are relatively less confident than the cross-entropy model for samples that they predict correctly. This is important, as we have already discussed that it is overconfidence that normally makes deep neural networks miscalibrated. Figure 2(c) shows that in contrast to the cross-entropy model, for which the NLL for misclassified test samples increases significantly after epoch 150, the rise in this value for the focal loss models is much less severe. Additionally, in Figure 2(d), we notice that the entropy of the softmax distribution for misclassified test samples is consistently (if marginally) higher for focal loss than for cross-entropy (consistent with Equation 1).

Note that from Figure 2(a), one may think that applying early stopping when training a model on cross-entropy can provide better calibration scores. However, there is no ideal way of doing early stopping that provides the best calibration error and the best test set accuracy. For fair comparison, we chose 3 intermediate models for each loss function with the best val set ECE, NLL and accuracy, and observed that: a) for every stopping criterion, focal loss outperforms cross-entropy in both test set accuracy and ECE, b) when using val set ECE as a stopping criterion, the intermediate model for cross-entropy indeed improves its test set ECE, but at the cost of a significantly higher test error. Please refer to Appendix J for more details.

As per §3, an increase in the test NLL and a decrease in the test entropy for misclassified samples, along with no corresponding increase in the test NLL for the correctly classified samples, can be interpreted as the network starting to predict softmax distributions for the misclassified samples that are ever more peaky in the wrong place. Notably, our results in Figures 2(b), 2(c) and 2(d) clearly show that this effect is significantly reduced when training with focal loss rather than cross-entropy, leading to a better-calibrated network whose predictions are less peaky in the wrong place.

Theoretical justification: As mentioned previously, once a model trained using cross-entropy reaches high training accuracy, the optimiser may try to further reduce the training NLL by increasing the confidences for the correctly classified samples. It may achieve this by magnifying the network weights to increase the magnitudes of the logits. To verify this hypothesis, we plot the L_2 norm of the weights of the last linear layer for all four networks as a function of the training epoch (see Figure 2(e)). Notably, although the norms of the weights for the models trained on focal loss are initially higher than that for the cross-entropy model, a *complete reversal* in the ordering of the weight norms occurs between epochs 150 and 250. In other words, as the networks start to become miscalibrated, the weight norm for the cross-entropy model also starts to become greater than those for the focal loss models. In practice, this is because focal loss, by design, starts to act as a regulariser on the network’s weights once the model has gained a certain amount of confidence in its predictions. This behaviour of focal loss can be observed even on a much simpler setup like a linear model (see Appendix C). To better understand this, we start by considering the following proposition (proof in Appendix D):

Proposition 1. For focal loss $\mathcal{L}_f$ and cross-entropy $\mathcal{L}_c$, the gradients $\frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} g(\hat{p}_{i,y_i}, \gamma)$, where $g(p, \gamma) = (1-p)^\gamma - \gamma p(1-p)^{\gamma-1} \log(p)$, $\gamma \in \mathbb{R}^+$ is the focal loss hyperparameter, and $\mathbf{w}$ denotes the parameters of the last linear layer. Thus $\left\| \frac{\partial \mathcal{L}_f}{\partial \mathbf{w}} \right\| \leq \left\| \frac{\partial \mathcal{L}_c}{\partial \mathbf{w}} \right\|$ if $g(\hat{p}_{i,y_i}, \gamma) \in [0, 1]$.

Proposition 1 shows the relationship between the norms of the gradients of the last linear layer for focal loss and cross-entropy loss, for the same network architecture. Note that this relation depends on a function $g(p, \gamma)$, which we plot in Figure 3(a) to understand its behaviour. It is clear that for

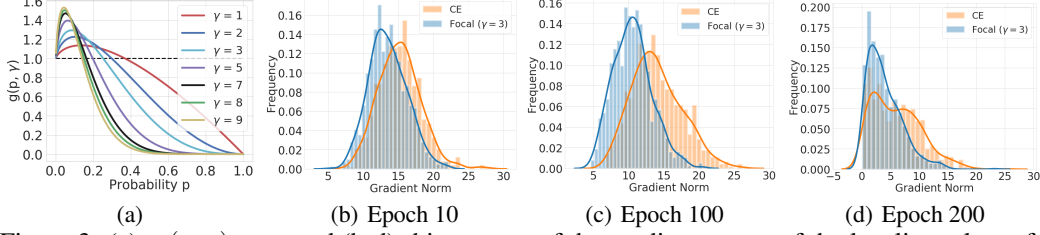


Figure 3: (a): $g(p, \gamma)$ vs. p and (b-d): histograms of the gradient norms of the last linear layer for both cross-entropy and focal loss.

every γ , there exists a (different) threshold p_0 such that for all $p \in [0, p_0]$, $g(p, \gamma) \geq 1$, and for all $p \in (p_0, 1]$, $g(p, \gamma) < 1$. (For example, for $\gamma = 1$, $p_0 \approx 0.4$.) We use this insight to further explain why focal loss provides implicit weight regularisation.

Implicit weight regularisation: For a network trained using focal loss with a fixed γ , during the initial stages of the training, when $\hat{p}_{i, y_i} \in (0, p_0)$, $g(\hat{p}_{i, y_i}, \gamma) > 1$. This implies that the confidences of the focal loss model’s predictions will initially increase faster than they would for cross-entropy. However, as soon as $\hat{p}_{i, y_i}$ crosses the threshold p_0 , $g(\hat{p}_{i, y_i}, \gamma)$ falls below 1 and reduces the size of the gradient updates made to the network weights, thereby having a regularising effect on the weights. This is why, in Figure 2(e), we find that the weight norms of the models trained with focal loss are initially higher than that for the model trained using cross-entropy. However, as training progresses, we find that the ordering of the weight norms reverses, as focal loss starts regularising the network weights. Moreover, we can draw similar insights from Figures 3(b), 3(c) and 3(d), in which we plot histograms of the gradient norms of the last linear layer (over all samples in the training set) at epochs 10, 100 and 200, respectively. At epoch 10, the gradient norms for cross-entropy and focal loss are similar, but as training progresses, those for cross-entropy decrease less rapidly than those for focal loss, indicating that the gradient norms for focal loss are consistently lower than those for cross-entropy throughout training.

Finally, observe in Figure 3(a) that for higher γ values, the fall in $g(p, \gamma)$ is steeper. We would thus expect a greater weight regularisation effect for models that use higher values of γ . This explains why, of the three models that we trained using focal loss, the one with $\gamma = 3$ outperforms (in terms of calibration) the one with $\gamma = 2$, which in turn outperforms the model with $\gamma = 1$. Based on this observation, one might think that, in general, a higher value of gamma would lead to a more calibrated model. However, this is not the case, as we notice from Figure 3(a) that for $\gamma \geq 7$, $g(p, \gamma)$ reduces to nearly 0 for a relatively low value of p (around 0.5). As a result, using values of γ that are too high will cause the gradients to die (i.e. reduce to nearly 0) early, at a point at which the network’s predictions remain ambiguous, thereby causing the training process to fail.

How to choose γ : As discussed, focal loss provides implicit entropy and weight regularisation, which heavily depend on the value of γ . Finding an appropriate γ is normally done using cross-validation. Also, traditionally, γ is fixed for all samples in the dataset. However, as shown, the regularisation effect for a sample i depends on $\hat{p}_{i, y_i}$, i.e. the predicted probability for the ground truth label for the sample. It thus makes sense to choose γ for each sample based on the value of $\hat{p}_{i, y_i}$. To this end, we provide Proposition 2 (proof in Appendix D), which we use to find a solution to this problem:

Proposition 2. *Given a p_0 , for $1 \geq p \geq p_0 > 0$, $g(p, \gamma) \leq 1$ for all $\gamma \geq \gamma^* = \frac{a}{b} + \frac{1}{\log a} W_{-1} \left(-\frac{a^{(1-a/b)}}{b} \log a \right)$, where $a = 1 - p_0$, $b = p_0 \log p_0$, and W_{-1} is the Lambert-W function [4]. Moreover, for $p \geq p_0 > 0$ and $\gamma \geq \gamma^*$, the equality $g(p, \gamma) = 1$ holds only for $p = p_0$ and $\gamma = \gamma^*$.*

It is worth noting that there exist multiple values of γ where $g(p, \gamma) \leq 1$ for all $p \geq p_0$. For a given p_0 , Proposition 2 allows us to compute γ s.t. (i) $g(p_0, \gamma) = 1$; (ii) $g(p, \gamma) \geq 1$ for $p \in [0, p_0]$; and (iii) $g(p, \gamma) < 1$ for $p \in (p_0, 1]$. This allows us to control the magnitude of the gradients for a particular sample i based on the current value of $\hat{p}_{i, y_i}$, and gives us a way of obtaining an informed value of γ for each sample. For instance, a reasonable policy might be to choose γ s.t. $g(\hat{p}_{i, y_i}, \gamma) > 1$ if $\hat{p}_{i, y_i}$ is small (say less than 0.25), and $g(\hat{p}_{i, y_i}, \gamma) < 1$ otherwise. Such a policy will have the effect of making the weight updates larger for samples having a low predicted probability for the correct class and smaller for samples with a relatively higher predicted probability for the correct class.

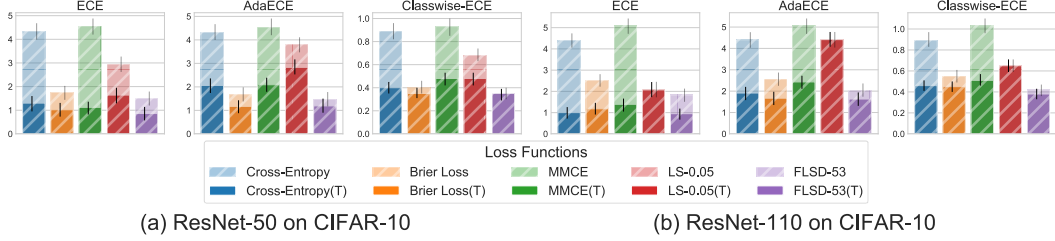


Figure 4: Bar plots with confidence intervals for ECE, AdaECE and Classwise-ECE, computed for ResNet-50 (first 3 figures) and ResNet-110 (last 3 figures) on CIFAR-10.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FL-3 (Ours)		FLSD-53 (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	17.52	3.42(2.1)	6.52	3.64(1.1)	15.32	2.38(1.8)	7.81	4.01(1.1)	5.13	1.97(1.1)	4.5	2.0(1.1)
	ResNet-110	19.05	4.43(2.3)	7.88	4.65(1.2)	19.14	3.86(2.3)	11.02	5.89(1.1)	8.64	3.95(1.2)	8.56	4.12(1.2)
	Wide-ResNet-26-10	15.33	2.88(2.2)	4.31	2.7(1.1)	13.17	4.37(1.9)	4.84	4.84(1)	2.13	2.13(1)	3.03	1.64(1.1)
	DenseNet-121	20.98	4.27(2.3)	5.17	2.29(1.1)	19.13	3.06(2.1)	12.89	7.52(1.2)	4.15	1.25(1.1)	3.73	1.31(1.1)
CIFAR-10	ResNet-50	4.35	1.35(2.5)	1.82	1.08(1.1)	4.56	1.19(2.6)	2.96	1.67(0.9)	1.48	1.42(1.1)	1.55	0.95(1.1)
	ResNet-110	4.41	1.09(2.8)	2.56	1.25(1.2)	5.08	1.42(2.8)	2.09	2.09(1)	1.55	1.02(1.1)	1.87	1.07(1.1)
	Wide-ResNet-26-10	3.23	0.92(2.2)	1.25	1.25(1)	3.29	0.86(2.2)	4.26	1.84(0.8)	1.69	0.97(0.9)	1.56	0.84(0.9)
	DenseNet-121	4.52	1.31(2.4)	1.53	1.53(1)	5.1	1.61(2.5)	1.88	1.82(0.9)	1.32	1.26(0.9)	1.22	1.22(1)
Tiny-ImageNet	ResNet-50	15.32	5.48(1.4)	4.44	4.13(0.9)	13.01	5.55(1.3)	15.23	6.51(0.7)	1.87	1.87(1)	1.76	1.76(1)
20 Newsgroups	Global Pooling CNN	17.92	2.39(3.4)	13.58	3.22(2.3)	15.48	6.78(2.2)	4.79	2.54(1.1)	8.67	3.51(1.5)	6.92	2.19(1.5)
SST Binary	Tree-LSTM	7.37	2.62(1.8)	9.01	2.79(2.5)	5.03	4.02(1.5)	4.84	4.11(1.2)	16.05	1.78(0.5)	9.19	1.83(0.7)

Table 1: ECE (%) computed for different approaches both pre and post temperature scaling (cross-validating T on ECE). Optimal temperature for each method is indicated in brackets. $T \approx 1$ indicates innately calibrated model.

Following the aforementioned arguments, we choose a threshold p_0 of 0.25, and use Proposition 2 to obtain a γ policy such that $g(p, \gamma)$ is observably greater than 1 for $p \in [0, 0.25)$ and $g(p, \gamma) < 1$ for $p \in (0.25, 1]$. In particular, we use the following schedule: if $\hat{p}_{i, y_i} \in [0, 0.2)$, then $\gamma = 5$, otherwise $\gamma = 3$ (note that $g(0.2, 5) \approx 1$ and $g(0.25, 3) \approx 1$: see Figure 3(a)). We find this γ policy to perform consistently well across multiple classification datasets and network architectures. Having said that, one can calculate multiple such schedules for γ following Proposition 2, using the intuition of having a relatively high γ for low values of $\hat{p}_{i, y_i}$ and a relatively low γ for high values of $\hat{p}_{i, y_i}$.

5 Experiments

We conduct image and document classification experiments to test the performance of focal loss. For the former, we use CIFAR-10/100 [13] and Tiny-ImageNet [6], and train ResNet-50, ResNet-110 [8], Wide-ResNet-26-10 [42] and DenseNet-121 [10] models, and for the latter, we use 20 Newsgroups [17] and Stanford Sentiment Treebank (SST) [32] datasets and train Global Pooling CNN [18] and Tree-LSTM [33] models. Further details on the datasets and training can be found in Appendix E.

Baselines Along with cross-entropy loss, we compare our method against the following baselines: a) *MMCE* (Maximum Mean Calibration Error) [16], a continuous and differentiable proxy for calibration error that is normally used as a regulariser alongside cross-entropy, b) *Brier loss* [1], the squared error between the predicted softmax vector and the one-hot ground truth encoding (Brier loss is an important baseline as it can be decomposed into calibration and refinement [5]), and c) *Label smoothing* [20] (LS): given a one-hot ground-truth distribution $\mathbf{q}$ and a smoothing factor α (hyperparameter), the smoothed vector $\mathbf{s}$ is obtained as $\mathbf{s}_i = (1 - \alpha)\mathbf{q}_i + \alpha(1 - \mathbf{q}_i)/(K - 1)$, where $\mathbf{s}_i$ and $\mathbf{q}_i$ denote the i^{th} elements of $\mathbf{s}$ and $\mathbf{q}$ respectively, and K is the number of classes. Instead of $\mathbf{q}$, $\mathbf{s}$ is treated as the ground truth. We train models using $\alpha = 0.05$ and $\alpha = 0.1$, but find $\alpha = 0.05$ to perform better. We thus report the results obtained from LS-0.05 with $\alpha = 0.05$.

Focal Loss: As mentioned in §4, our proposed approach is the sample-dependent schedule FLSD-53 ($\gamma = 5$ for $\hat{p}_y \in [0, 0.2)$, and $\gamma = 3$ for $\hat{p}_y \in [0.2, 1]$), which we find to perform well across most classification datasets and network architectures. In addition, we also train other focal loss baselines, including ones with γ fixed to 1, 2 and 3, and also ones that have a training epoch-dependent schedule for γ . Among the focal loss models trained with a fixed γ , using validation set we find $\gamma = 3$ (FL-3) to perform the best. Details of all these approaches can be found in Appendix F.

Temperature Scaling: In order to compute the optimal temperature, we use two different methods: (a) learning the temperature by minimising val set NLL, and (b) performing grid search over tempera-

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	FL-3 (Ours)	FLSD-53 (Ours)
CIFAR-100	ResNet-50	23.3	23.39	23.2	23.43	22.75	23.22
	ResNet-110	22.73	25.1	23.07	23.43	22.92	22.51
	Wide-ResNet-26-10	20.7	20.59	20.73	21.19	19.69	20.11
	DenseNet-121	24.52	23.75	24.0	24.05	23.25	22.67
CIFAR-10	ResNet-50	4.95	5.0	4.99	5.29	5.25	4.98
	ResNet-110	4.89	5.48	5.4	5.52	5.08	5.42
	Wide-ResNet-26-10	3.86	4.08	3.91	4.2	4.13	4.01
	DenseNet-121	5.0	5.11	5.41	5.09	5.33	5.46
Tiny-ImageNet	ResNet-50	49.81	53.2	51.31	47.12	49.69	49.06
20 Newsgroups	Global Pooling CNN	26.68	27.06	27.23	26.03	29.26	27.98
SST Binary	Tree-LSTM	12.85	12.85	11.86	13.23	12.19	12.8

Table 2: Test set error (%) computed for different approaches.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		FL-3 (Ours)		FLSD-53 (Ours)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-10/SVHN	ResNet-110	61.71	59.66	94.80	95.13	85.31	85.39	68.68	68.68	96.74	96.92	90.83	90.97
	Wide-ResNet-26-10	96.82	97.62	94.51	94.51	97.35	97.95	84.63	84.66	98.19	98.05	98.29	98.20
CIFAR-10/CIFAR-10-C	ResNet-110	77.53	75.16	84.09	83.86	71.96	70.02	72.17	72.18	82.27	82.18	85.05	84.70
	Wide-ResNet-26-10	81.06	80.68	85.03	85.03	82.17	81.72	71.10	71.16	82.17	81.86	87.05	87.30

Table 3: AUROC (%) computed for models trained on CIFAR-10 (in-distribution), and using SVHN and CIFAR-10-C (Gaussian Noise corruption with severity level 5) respectively as the OoD datasets.

tures between 0 and 10, with a step of 0.1, and finding the one that minimises val set ECE. We find the second approach to produce *stronger baselines* and report results obtained using this approach.

Performance Gains: We report ECE% (computed using 15 bins) along with optimal temperatures in Table 1, and test set error in Table 2. We report the other calibration scores (AdaECE, Classwise-ECE, MCE and NLL) in Appendix F. Firstly, for all dataset-network pairs, we obtain very competitive classification accuracies (shown in Table 2). Secondly, *it is clear from Table 1, and Tables F.1 and F.2 in the appendix, that focal loss with sample-dependent γ and with $\gamma = 3$ outperform all the baselines: cross-entropy, label smoothing, Brier loss and MMCE.* They broadly produce the lowest calibration errors *both before and after temperature scaling*. This observation is particularly encouraging, as it also indicates that a principled method of obtaining values of γ for focal loss can produce a very calibrated model, with no need to use validation set for tuning γ . As shown in Figure 4, we also compute 90% confidence intervals for ECE, AdaECE and Classwise-ECE using 1000 bootstrap samples following [15], and using ResNet-50/110 trained on CIFAR-10 (see Appendix G for more results). Note that FLSD-53 produces the lowest calibration errors in general, and the difference in the metric values between FLSD-53 and other approaches (except Brier loss) is mostly statistically significant (i.e., confidence intervals don’t overlap), especially before temperature scaling. In addition to the lower calibration errors, there are other advantages of focal loss as well, which we explore next.

More advantages of focal loss: *Behaviour on Out-of-Distribution (OoD) data:* A perfectly calibrated model should have low confidence whenever it misclassifies, including when it encounters data which is OoD [34]. Although temperature scaling calibrates a model under the i.i.d. assumption, it is known to fail under distributional shift [27]. Since focal loss has implicit regularisation effects on the network (see §4), we investigate if it helps to learn representations that are more robust to OoD data. To do this, we use ResNet-110 and Wide-ResNet-26-10 trained on CIFAR-10 and consider the SVHN [23] test set and CIFAR-10-C [9] with Gaussian noise corruption at severity 5 as OoD data. We use the entropy of the softmax distribution as the measure of confidence or uncertainty, and report the corresponding AUROC scores both before and after temperature scaling in Table 3. For both SVHN and CIFAR-10-C (using Gaussian noise), models trained on focal loss clearly obtain the highest AUROC scores. *Note that Focal loss even without temperature scaling performs better than other methods with temperature scaling.* We also present the ROC plots pre and post temperature scaling for models trained on CIFAR-10 and tested on SVHN in Figure 5. Thus, it is quite encouraging to note that models trained on focal loss are not only better calibrated under the i.i.d. assumption, but also seem to perform better than other competitive loss functions when we try shifting the distribution from CIFAR-10 to SVHN or CIFAR-10-C (pre and post temperature scaling).

Confident and Calibrated Models: It is worth noting that focal loss with sample-dependent γ has optimal temperatures that are very close to 1, mostly lying between 0.9 and 1.1 (see Table 1). This property is shown by the Brier loss and label smoothing models as well, albeit with worse calibration

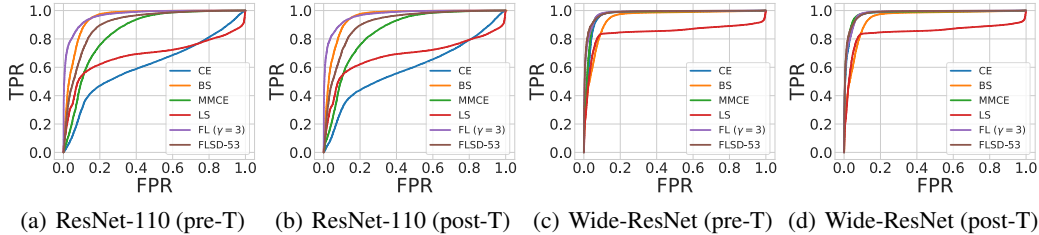


Figure 5: ROC plots obtained from ResNet-110 and Wide-ResNet-26-10 architectures trained on CIFAR-10 (in-distribution) and tested on SVHN (OoD), both pre and post temperature scaling.

errors. By contrast, the temperatures for cross-entropy and MMCE models are significantly higher, with values lying between 2.0 and 2.8. An optimal temperature close to 1 indicates that the model is innately calibrated, and cannot be made significantly more calibrated by temperature scaling. In fact, a temperature much greater than 1 can make a model underconfident in general, as it is applied irrespective of the correctness of model outputs. We observe this empirically for ResNet-50 and ResNet-110 trained on CIFAR-10. Although models trained with cross-entropy have much higher confidence before temperature scaling than those trained with focal loss, after temperature scaling, focal loss models are significantly more confident in their predictions. We provide quantitative and qualitative empirical results to support this claim in Appendix H.

6 Conclusion

In this paper, we have studied the properties of focal loss, an alternative loss function that can yield classification networks that are more naturally calibrated than those trained using the conventional cross-entropy loss, while maintaining accuracy. In particular, we show in §4 that focal loss implicitly maximises entropy while minimising the KL divergence between the predicted and the target distributions. We also show that, because of its design, it naturally regularises the weights of a network during training, reducing NLL overfitting and thereby improving calibration. Furthermore, we empirically observe that models trained using focal loss are not only better calibrated under i.i.d. assumptions, but can also be better at detecting OoD samples which we show by taking CIFAR-10 as the in-distribution dataset and SVHN and CIFAR-10-C as out-of-distribution datasets, something which temperature scaling fails to achieve.

7 Broader Impact

Our work shows that using the right kind of loss function can lead to a calibrated model. This helps in improving the reliability of these models when used in real-world applications. It can help in deployment of the models such that users can be alerted when its prediction may not be trustworthy. We do not directly see a situation where calibrated neural networks can have a negative impact on society, but we do believe that research on making models more calibrated will help improve fairness and trust in AI.

Acknowledgments and Disclosure of Funding

This work was started whilst J Mukhoti was at FiveAI, and completed after he moved to the University of Oxford. V Kulharia is wholly funded by a Toyota Research Institute grant. A Sanyal acknowledges support from The Alan Turing Institute under the Turing Doctoral Studentship grant TU/C/000023. This work was also supported by the Royal Academy of Engineering under the Research Chair and Senior Research Fellowships scheme, EPSRC/MURI grant EP/N019474/1 and FiveAI.

References

- [1] Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 1950.
- [2] Chensi Cao, Feng Liu, Hai Tan, Deshou Song, Wenjie Shu, Weizhong Li, Yiming Zhou, Xiaochen Bo, and Zhi Xie. Deep Learning and Its Applications in Biomedicine. *Genomics, Proteomics & Bioinformatics*, 16(1):17–32, 2018.
- [3] Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke. The rise of deep learning in drug discovery. *Drug Discovery Today*, 23(36):1241–1250, 2018.
- [4] Robert M Corless, Gaston H Gonnet, David EG Hare, David J Jeffrey, and Donald E Knuth. On the Lambert W Function. *Advances in Computational Mathematics*, 5(1):329–359, 1996.
- [5] Morris H DeGroot and Stephen E Fienberg. The Comparison and Evaluation of Forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 1983.
- [6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.
- [7] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On Calibration of Modern Neural Networks. In *ICML*, 2017.
- [8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.
- [9] Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *arXiv preprint arXiv:1903.12261*, 2019.
- [10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *CVPR*, 2017.
- [11] Andreas Kamilaris and Francesc X Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018.
- [12] Justin Ker, Lipo Wang, Jai Rao, and Tchoyoson Lim. Deep Learning Applications in Medical Image Analysis. *IEEE Access*, 6:9375–9389, 2018.
- [13] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, Department of Computer Science, University of Toronto, 2009.
- [14] Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with Dirichlet calibration. In *NeurIPS*, 2019.

- [15] Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified Uncertainty Calibration. In *NeurIPS*, 2019.
- [16] Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. Trainable Calibration Measures For Neural Networks From Kernel Mean Embeddings. In *ICML*, 2018.
- [17] Ken Lang. NewsWeeder: Learning to Filter Netnews. In *Machine Learning Proceedings 1995*, pages 331–339. Elsevier, 1995.
- [18] Min Lin, Qiang Chen, and Shuicheng Yan. Network In Network. In *ICLR*, 2014.
- [19] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *ICCV*, 2017.
- [20] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When Does Label Smoothing Help? In *NeurIPS*, 2019.
- [21] Mahdi Pakdaman Naeini and Gregory F Cooper. Binary Classifier Calibration using an Ensemble of Near Isotonic Regression Models. In *ICDM*, 2016.
- [22] Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. Obtaining Well Calibrated Probabilities Using Bayesian Binning. In *AAAI*, 2015.
- [23] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NeurIPS-W*, 2011.
- [24] 20 Newsgroups. <https://github.com/aviralkumar2907/MMCE>, 2018. Accessed: 2019-05-22.
- [25] Khanh Nguyen and Brendan O’Connor. Posterior calibration and exploratory analysis for natural language processing models. *arXiv preprint arXiv:1508.05154*, 2015.
- [26] Alexandru Niculescu-Mizil and Rich Caruana. Predicting Good Probabilities With Supervised Learning. In *ICML*, 2005.
- [27] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can You Trust Your Model’s Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift. In *NeurIPS*, 2019.
- [28] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014.
- [29] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing Neural Networks by Penalizing Confident Output Distributions. *arXiv preprint arXiv:1701.06548*, 2017.
- [30] John Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Advances in Large Margin Classifiers*, 10(3):61–74, 1999.
- [31] Train CIFAR10 with PyTorch, 2019. Available online at <https://github.com/kuangliu/pytorch-cifar> (as of 22nd May 2019).
- [32] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2013.
- [33] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Association for Computational Linguistics (ACL)*, 2015.
- [34] Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks. In *NeurIPS*, 2019.

- [35] TreeLSTM. <https://github.com/ttpro1995/TreeLSTMSentiment>, 2015. Accessed: 2019-05-22.
- [36] Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas B Schön. Evaluating model calibration in classification. In *AISTATS*, 2019.
- [37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, 2017.
- [38] Jinjiang Wang, Yulin Ma, Laibin Zhang, Robert X Gao, and Dazhong Wu. Deep learning for smart manufacturing: Methods and applications. *JMSY*, 48:144–156, 2018.
- [39] David Widmann, Fredrik Lindsten, and Dave Zachariah. Calibration tests in multi-class classification: A unifying framework. In *NeurIPS*, 2019.
- [40] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *ICML*, 2001.
- [41] Bianca Zadrozny and Charles Elkan. Transforming Classifier Scores into Accurate Multiclass Probability Estimates. In *SIGKDD*, 2002.
- [42] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. In *BMVC*, 2016.

D.4 Similarity Learning for Dense Label Transfer

Contributions:

- Mohammad Najafi set up the initial pipeline.
- Viveka Kulharia contributed in making the pipeline more fast and efficient.
- Mohammad Najafi, Ajanthan Thalaiyasingam, and Philip H. S. Torr helped in the writing.

Similarity Learning for Dense Label Transfer

Mohammad Najafi* Viveka Kulharia* Thalaiyasingam Ajanthan Philip H. S. Torr

University of Oxford, UK

{monaj, viveka, ajanthan, phst}@robots.ox.ac.uk

Abstract

In this work, we introduce a simple and flexible method for video object segmentation based on similarity learning. The proposed method can learn to perform dense label transfer from one image to the other. More specifically, the objective is to learn a similarity metric for dense pixel-wise correspondence between two images. This learned model can then be used in a label transfer framework to propagate object annotations from a reference frame to all the subsequent frames in a video. Unlike previous methods, our similarity learning approach works fairly well across various domains, even when no domain adaptation is involved. Using the proposed approach, we achieved the second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

1. Introduction

Object segmentation in a video is an important problem in computer vision with many interesting real world applications in surveillance, robotics, and autonomous driving. Given one or more target objects, the task is to segment those objects in every frame of the video.

In this paper, we introduce a simple and flexible method for video object segmentation based on *similarity learning*. More specifically, we turn the problem of video object segmentation into a dense binary classification problem using a Siamese network, where the objective is to predict whether or not a pair of selected pixels from two frames belong to the same object (or instance of an object). Our goal here is to learn an embedding space in which, pixels of the same object are encouraged to be close to each other, whereas pixels of different objects are pushed away from each other. In short, we train a similarity metric for pixel correspondence between two images. We then use the trained model in a *label transfer* framework to propagate object annotations from a frame with ground-truth information (*e.g.*, first frame of the video in case of *one-shot* video object segmentation task) to other frames.

Note that, with this simple similarity learning approach,



Figure 1: One-shot segmentation outputs for sample videos from DAVIS-2016, obtained using our model without any test domain adaptation, online learning or interaction. Each row illustrates outputs for different time frames of the video.

our method performs fairly well across various domains and clearly outperforms state-of-the-art methods when no domain adaptation is involved. Furthermore, unlike previous approaches [9], multi-target segmentation comes at virtually no cost in our approach. Figure 1 demonstrates qualitative results computed using our method for one-shot object segmentation on sample videos, with no test domain adaptation, online learning or human interaction involved.

In this work, we consider the scenario of *interactive* video object segmentation with scribbles, as suggested by [1]. In this process, the segmentation system learns interactively by getting feedbacks from the evaluation system. Moreover, at each interaction, the method is fine-tuned using only a set of squiggles from one video frame, which are much easier to obtain compared to dense ground-truth object masks, but make the problem more challenging.

Following the above-mentioned protocol, we developed an interactive segmentation system based on our dense similarity learning and label transfer method, and achieved the second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

2. Method

Our approach builds upon the standard Siamese network to learn the dense correspondence between the target objects in two given images. To this end, we first review the Siamese architecture and then introduce our idea for using it in a dense label transfer framework.

* Joint first author

2.1. Siamese Networks

The Siamese network is a special type of neural network, which is trained to make comparisons between pairs of input data. In other words, it learns whether two input items belong to the same class or not. Figure 2 illustrates an example of a Siamese network that is used in our framework.

Let $\{x, y\}$ be a pair of input data (usually a pair of images) which is fed into two identical deep neural networks with a shared set of parameters $\mathcal{W}$. These networks generate two embeddings $\{\Phi(x; \mathcal{W}), \Phi(y; \mathcal{W})\}$ at their outputs. The inner-product of these embeddings can be considered as the similarity between x and y . Also let ℓ be the target of the Siamese network, which is defined as $\ell = 1[l_x == l_y]$, where l_x and l_y are the class labels of x and y , respectively. In other words, ℓ is set to one if the input pair have the same class labels and zero otherwise. Now the objective of the standard Siamese network can be defined as:

$$\min_{\mathcal{W}} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\langle \Phi(x_i; \mathcal{W}), \Phi(y_i; \mathcal{W}) \rangle, \ell_i), \quad (1)$$

where i ranges over the training set of data pairs of size N , $\mathcal{L}$ is a standard loss function, e.g., binary cross entropy, and $\langle \cdot, \cdot \rangle$ denotes the inner-product operation. The hypothesis is that, the embedding space obtained by this optimization process corresponds to a similarity metric, in which, data of similar class are clustered together

2.2. Object Segmentation using Siamese Networks

In this section, we extend the idea of similarity learning to use it for dense video object segmentation. Figure 2 illustrates the proposed structure, where each branch of the Siamese network is a Deeplab-Resnet-101 [2] model with dilated convolutions to preserve spatial information, connected to a decoder network which upsamples the embeddings to a higher resolution. Note that, as shown in [2, 7], preserving spatial information is crucial for segmentation tasks. The outputs of the decoders are two pixel embedding maps $\{\Phi(x; \mathcal{W}), \Phi(y; \mathcal{W})\}$, where x and y are input images of size $[H, W]$ and Φ is a three-dimensional tensor of size $[h, w, d]$, where $h = H/2$, $w = W/2$ and d is the embedding size.

Next, we compute the similarities of the pixels between images x and y by taking the inner-product of their embeddings. The output of this step is a dense similarity map of size $[h \times w, h \times w]$, which indicates the similarity of every pixel embedding in Φ_x with all the pixel embeddings in Φ_y . At this point, we apply batch-norm [4] to the output of inner-product to prevent the gradients from exploding. Finally, the sigmoid nonlinearity is applied to convert the similarity scores into probabilities. Furthermore, the target of the Siamese network is a matrix of size $[h \times w, h \times w]$ (denoted as $\mathbf{L}$), which is a binary map that shows if two pixels in the images belong to the same object class.

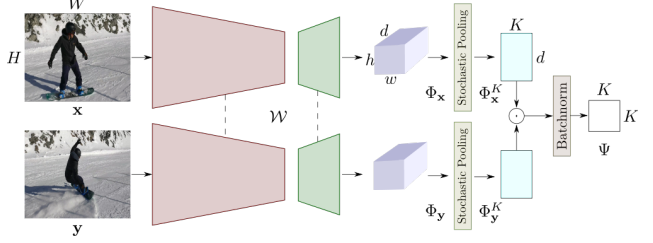


Figure 2: **The Siamese architecture used for learning dense correspondence.** The model takes in a pair of images and outputs the similarity of their pixels by computing the inner-product of their subsampled pixel embeddings ($\langle \Phi_x^K, \Phi_y^K \rangle$). During evaluation, stochastic pooling is not applied on the target image y and the inner-product output is $\langle \Phi_x^K, \Phi_y \rangle$.

The proposed network is trained according to the objective function in Eq. 1, but for an aggregate loss over $[h \times w, h \times w]$ data points for each pair of images. After training, the output similarity map can be used to predict the label correspondence between pixels of images x and y . Then, during test time, we leverage this similarity map to predict the segmentation map of the second image y , given a ground-truth segmentation map of the first image x , via a *label transfer* approach, as described in Sec. 2.2.1.

The main challenge when using the Siamese architecture to learn dense correspondence is the high memory requirement. Note that, the output of the inner-product $\Psi = \langle \Phi_x, \Phi_y \rangle$ is of dimension $[h \times w, h \times w]$. In our case, $h = 240$ and $w = 427$, which leads to 1×10^{10} floating point values to be stored. This is infeasible to store on a standard GPU. To tackle this, we approximate the inner-product $\Psi = \langle \Phi_x, \Phi_y \rangle$ using only a subset of pixels.

We observe that, in an image, a large portion of the pixels are highly correlated to each other. Hence, it is possible to estimate the embedding of the image pixels using only a small fraction of them. Based on this argument, we introduce a *stochastic pooling* layer to the network, that subsamples k pixels uniformly at random for each object class present in the image. Thus, a total of K pixels are chosen from each of the embeddings. Now, the inner-product can be approximated as:

$$\Psi = \langle \Phi_x, \Phi_y \rangle \approx \langle \Phi_x^K, \Phi_y^K \rangle, \quad (2)$$

where Φ_x^K and Φ_y^K represent the embeddings corresponding to the chosen subsets of x and y , respectively. Note that, Ψ is of size $[K, K]$, where K is the chosen subset size. Also note that, the same chosen subsets are used to create the target map $\mathbf{L}^K$ for the Siamese network. In our experiments, we observed that the performance plateaus for $k > 100$, which supports the above argument. This effectively reduces the memory requirement from 1×10^{10} to

In this section, for brevity, we call the pixel embedding as a pixel.

1×10^4 (assuming one object in each image), and also makes it faster in both training and inference phases.

2.2.1 Label Transfer

Once the network is trained, one can label a query image $\mathbf{y}$ by matching its pixels with the pixels of a reference image $\mathbf{x}$ (whose ground-truth annotation is available) using the learned similarity model. At test time, the reference image is fed into the top branch of the Siamese network, while the query image is given to the bottom branch. Then a representative set of pixel embeddings are subsampled from the reference image via the stochastic pooling layer, and their similarities with the embeddings of the query pixels are computed. Subsequently, the class labels of the selected reference pixels are transferred to all the query pixels based on their similarity scores. Note that, in this step, the stochastic pooling is applied only to the reference image and the similarity scores using the inner-product are computed for all the query pixels, *i.e.*, $\Psi \approx \langle \Phi_{\mathbf{x}}^K, \Phi_{\mathbf{y}} \rangle$.

Let $\mathcal{N}_{l'}$ be the selected subset of pixels from the reference image for class l' , then the class label of pixel $i \in \mathcal{Q}$ within the query image $\mathcal{Q}$ is predicted using the following formulation:

$$l_i = \arg \max_{l'} \sum_{j \in \mathcal{N}_{l'}} \tilde{\Psi}_{ij}, \quad (3)$$

where $\tilde{\Psi}$ is the similarity score after the batch-norm layer.

3. Experiments

We evaluated our method interactively on DAVIS-2017 [8] validation (30 videos) and Test-dev (30 videos) sets. For validation, we used a model pre-trained on DAVIS-2017 training set (60 videos), however in the test phase, we pre-trained our model using the entire training+validation set (90 videos). All the experiments in this paper were performed on 480×854 resolution videos.

Interactive Segmentation The proposed approach was evaluated using the interactive toolkit released by [1]. Initially a set of squiggles representing the objects of interests in one of the video frames are provided as training examples. The segmentation system is fine-tuned based on this sparse set of ground-truth information and predicts pixel-level masks for targets over the entire video. The outputs are then analysed, and the video frame with the worst segmentation output is chosen to provide the user with a set of feedbacks on which regions the method has failed. Note that the feedbacks are also in the form of scribbles. This interaction improves the segmentation system, as it learns where it fails the most and updates its parameters such that those mistakes are resolved.

Evaluation Metrics Two evaluation metrics are designed for this interactive task by [1], which take into account both segmentation accuracy and speed. The first one is Area Under the Curve (AUC) of the plot Time vs Jaccard ($\mathcal{J}$) performance. Note that $\mathcal{J}$ is a standard metric which is defined

as the intersection-over-union (IoU) of the predicted object mask and its respective ground-truth mask. Each point in the AUC- $\mathcal{J}$ plot is computed considering the average time and the average Jaccard, obtained for a certain interaction, for the whole dataset. The second metric is called $\mathcal{J}@60$ which evaluates what Jaccard performance a method can reach within 60 seconds.

Network Structure Each branch of our network (Figure 2) is a Deeplab-Resnet-101 model (pre-trained on ImageNet [3] and Microsoft-coco image dataset [6]), which produces feature maps whose resolutions are eight times smaller than the resolution of input images. These features maps are upsampled using the decoder network which yields high-resolution embeddings. The decoder consists of a transposed convolution layer (with stride = 2) and a non-parametric bilinear upsampling layer (with stride = 2).

Training and Label Transfer At each training iteration, a video is randomly selected from the training pool, and then a pair of frames are drawn randomly from the video to feed the network. We followed the pooling strategy explained in Sec. 2.2 and selected $k = 100$ pixel embeddings in our experiments. The network was trained for 100000 iterations, using dense ground-truth annotations in the training split, with the base learning rate of 2.5×10^{-4} for the branches and a learning rate factor of two for the decoder ($\text{lr}_{\text{decoder}} = 5 \times 10^{-4}$, with a momentum of 0.9 and weight decay of 0.0005. In addition, learning rates were decreased polynomially, with a power of 0.9.

At the validation/test time, the second branch of the network is fed by a query image, and we use the learned similarity model to transfer the ground-truth labels of the training input image in the top branch to the query image. Label transfer, as explained in Sec. 2.2.1, is merely a label matching operation, which does not leverage any prior information over the image pixels. Hence we can further improve the output of this step by passing it through a dense CRF [5].

The interactive process included a sequence of domain adaptation (fine-tuning based on the provided set of ground-truth scribbles), label transfer, and post-processing (CRF), which were designed differently at different interactions. We tailored the following strategy based on a set of experiments on the validation set. We initially fine-tuned our pre-trained model using the first set of scribbles for 5s, but extended this fine-tuning time to 20s for the second interaction and 60s for all remaining interactions. At each interaction, a new video frame together with a set of scribbles were provided by the evaluation toolkit. We augmented the frame by random horizontal flipping, scaling (0.5 to 1.5) and rotation (-30° to 30°). The augmented data (with different augmentation parameters) fed the branches of the network and fine-tuned the model using their scribbles, together with the images and scribbles supplied at previous interactions. In addition, we stopped fine-tuning when the maximum value

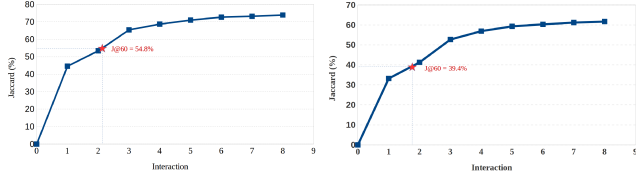


Figure 3: The performance of our method on DAVIS-2017 validation split (left) with $AUC = 70.2\%$ and $\mathcal{J}@60 = 54.8\%$, and Test-Dev split (right) with $AUC = 54.5\%$ and $\mathcal{J}@60 = 39.4\%$. Note that in the Test-Dev set, the 60s time-point occurs before the second interaction because more complex videos with larger number of objects are present and processed in this data split.

of loss dropped below 0.04 at each interaction. Also we skipped CRF for the first couple of interactions to speed up the evaluation. Furthermore, the provided scribbles at each interaction were dilated with a square kernel of size three, and we also denoted the image pixels that were far from all of the object scribbles, as background pixels. Note that whenever the feedbacks did not include any region from background during interactions, pixels which were the least similar to any of the initial ground-truth object scribbles were selected to represent the background.

Following the above strategy, we achieved the second place in both quality ($AUC-\mathcal{J}$) and speed ($\mathcal{J}@60$) tracks. The charts in Figure 3 show the results obtained using our model on the validation and test-dev splits of DAVIS-2017 dataset. As shown in the figure, the model has consistently improved by learning from the previous mistakes. Figure 4 demonstrates some examples, where the interactive process has incrementally improved the segmentation outputs.

4. Conclusion

In this work, we tackled the problem of video object segmentation using sparse ground-truth data, through similarity learning. In particular, we trained a Siamese network, whose objective is to predict whether or not two pixels from two video frames belong to the same object class. Next, a dense label transfer approach was proposed that used the learned similarity to propagate annotations from one video frame to the rest of the videos. The proposed method achieved the second place in the first DAVIS challenge for interactive video object segmentation, in both quality and speed tracks.

5. Acknowledgement

This work was supported by the EPSRC, ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We also thank NVIDIA for donating a GPU for this project. Viveka Kulharia is wholly funded by Toyota Research Institute’s grant.



Figure 4: **The impact of successive interactions on the quality of the segmentation outputs.** The top row illustrates the input images and the segmentation results for consecutive interactions are depicted in the 2nd to 8th rows.

References

- [1] S. Caelles, A. Montes, K.-K. Maninis, Y. Chen, L. Van Gool, F. Perazzi, and J. Pont-Tuset. The 2018 davis challenge on video object segmentation. *arXiv:1803.00557*, 2018.
- [2] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE T-PAMI*, 40(4):834–848, 2018.
- [3] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [4] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.
- [5] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *ICML*, 2013.
- [6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. 2014.
- [7] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.
- [8] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017.
- [9] P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017.

Bibliography

- 20 Newsgroups (2018). <https://github.com/aviralkumar2907/MMCE>. Accessed: 2019-05-22.
- Abadi, M. and D. Andersen (2016). “Learning to Protect Communications with Adversarial Neural Cryptography”. In: *arXiv preprint arXiv:1610.06918*.
- Ahn, Jiwoon, Sunghyun Cho, and Suha Kwak (2019). “Weakly supervised learning of instance segmentation with inter-pixel relations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2209–2218.
- Ahn, Jiwoon and Suha Kwak (2018). “Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Anderson, Sarah J, Heather A Jamniczky, Olave E Krigolson, Sylvain P Coderre, and Kent G Hecker (2019). “Quantifying two-dimensional and three-dimensional stereoscopic learning in anatomy using electroencephalography”. In: *NPJ science of learning* 4.1, pp. 1–9.
- Arandjelović, Relja and Andrew Zisserman (2019). “Object Discovery with a Copy-Pasting GAN”. In: *arXiv preprint arXiv:1905.11369*.
- Arjovsky, M. and L. Bottou (2017). “Towards Principled Methods For Training Generative Adversarial Networks”. In: *International Conference on Learning Representations*.
- Arjovsky, M., S. Chintala, and L. Bottou (2017). “Wasserstein gan”. In: *International Conference on Machine Learning*.
- Arora, S., R. Ge, Y. Liang, T. Ma, and Yi Zhang (2017). “Generalization and Equilibrium in Generative Adversarial Nets (GANs)”. In: *International Conference on Machine Learning*.
- Ashukha, Arsenii, Alexander Lyzhov, Dmitry Molchanov, and Dmitry Vetrov (2020). “Pitfalls of in-domain uncertainty estimation and ensembling in deep learning”. In: *ICLR*.
- Aubry, Mathieu and Bryan C. Russell (2015). “Understanding Deep Features With Computer-Generated Imagery”. In: *ICCV*.
- Bearman, Amy, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei (2016). “What’s the point: Semantic segmentation with point supervision”. In: *European conference on computer vision (ECCV)*. Springer, pp. 549–565.
- Berthelot, David, Tom Schumm, and Luke Metz (2017). “Began: Boundary equilibrium generative adversarial networks”. In: *arXiv preprint arXiv:1703.10717*.
- Bertinetto, Luca, João F. Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi (2016a). “Learning feed-forward one-shot learners”. In: *NIPS*.

- Bertinetto, Luca, Jack Valmadre, João F Henriques, Andrea Vedaldi, and Philip HS Torr (2016b). “Fully-Convolutional Siamese Networks for Object Tracking”. In: *ECCV Workshops*.
- Bishop, C (2007). “Pattern Recognition and Machine Learning (Information Science and Statistics), 1st edn. 2006. corr. 2nd printing edn”. In: *Springer, New York*.
- Blundell, Charles, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra (2015). “Weight uncertainty in neural network”. In: *ICML*. PMLR.
- Brier, Glenn W (1950). “Verification of forecasts expressed in terms of probability”. In: *Monthly weather review*.
- Caelles, S., K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool (2017). “One-Shot Video Object Segmentation”. In: *Computer Vision and Pattern Recognition (CVPR)*.
- Caelles, Sergi, Alberto Montes, Kevis-Kokitsi Maninis, Yuhua Chen, Luc Van Gool, Federico Perazzi, and Jordi Pont-Tuset (2018). “The 2018 DAVIS Challenge on Video Object Segmentation”. In: *arXiv:1803.00557*.
- Cao, Chensi, Feng Liu, Hai Tan, Deshou Song, Wenjie Shu, Weizhong Li, Yiming Zhou, Xiaochen Bo, and Zhi Xie (2018). “Deep Learning and Its Applications in Biomedicine”. In: *Genomics, Proteomics & Bioinformatics* 16.1, pp. 17–32.
- Cao, Yun-Hao and Jianxin Wu (2022). “A Random CNN Sees Objects: One Inductive Bias of CNN and Its Applications”. In: *AAAI*.
- Carreira, Joao and Andrew Zisserman (2017). “Quo vadis, action recognition? a new model and the kinetics dataset”. In: *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308.
- Chandra, Siddhartha, Nicolas Usunier, and Iasonas Kokkinos (2017). “Dense and low-rank gaussian crfs using deep embeddings”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5103–5112.
- Chaudhry, Arslan, Puneet K Dokania, and Philip HS Torr (2017). “Discovering class-specific pixels for weakly-supervised semantic segmentation”. In: *British Machine Vision Conference (BMVC)*.
- Che, T., Y. Li, A. Jacob, Y. Bengio, and W. Li (2017). “Mode Regularized Generative Adversarial Networks”. In: *International Conference on Learning Representations*.
- Chen, Hongming, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke (2018a). “The rise of deep learning in drug discovery”. In: *Drug Discovery Today* 23.36, pp. 1241–1250.
- Chen, L. C., G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille (2017a). “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs”. In: *IEEE T-PAMI* 40.4, pp. 834–848.
- Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille (2014). “Semantic image segmentation with deep convolutional nets and fully connected crfs”. In: *arXiv preprint arXiv:1412.7062*.
- Chen, Liang-Chieh, George Papandreou, Florian Schroff, and Hartwig Adam (2017b). “Re-thinking Atrous Convolution for Semantic Image Segmentation”. In: *CoRR* abs/1706.05587.

- Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam (2018b). “Encoder-decoder with atrous separable convolution for semantic image segmentation”. In: *European Conference on Computer Vision (ECCV)*, pp. 801–818.
- Chen, Xi, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel (2016). “Infogan: Interpretable representation learning by information maximizing generative adversarial nets”. In: *Advances in Neural Information Processing Systems*.
- Cheng, Ho Kei, Yu-Wing Tai, and Chi-Keung Tang (2021). “Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5559–5568.
- Corless, Robert M, Gaston H Gonnet, David EG Hare, David J Jeffrey, and Donald E Knuth (1996). “On the Lambert W Function”. In: *Advances in Computational Mathematics* 5.1, pp. 329–359.
- Cruise (2020). *Machine Learning for Autonomous Vehicle Perception at Cruise*. URL: <https://www.youtube.com/watch?v=-UPfyvDJz9I>.
- Cutting, James E and Peter M Vishton (1995). “Perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth”. In: *Perception of space and motion*. Elsevier, pp. 69–117.
- Dai, Jifeng, Kaiming He, and Jian Sun (2015). “Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation”. In: *International Conference on Computer Vision (ICCV)*, pp. 1635–1643.
- Dalal, Navneet and Bill Triggs (2005). “Histograms of oriented gradients for human detection”. In: *CVPR*.
- Damianou, Andreas and Neil D Lawrence (2013). “Deep gaussian processes”. In: PMLR.
- Dean, Jeff (2021). *Introducing Pathways: A next-generation AI architecture*. URL: <https://blog.google/technology/ai/introducing-pathways-next-generation-ai-architecture/>.
- DeGroot, Morris H and Stephen E Fienberg (1983). “The Comparison and Evaluation of Forecasters”. In: *Journal of the Royal Statistical Society: Series D (The Statistician)*.
- Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Durugkar, I., I. Gemp, and S. Mahadevan (2017). “Generative Multi-Adversarial Networks”. In: *International Conference on Learning Representations*.
- Everingham, Mark, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman (2010). “The Pascal Visual Object Classes (VOC) Challenge”. In: *International journal of computer vision* 88.2, pp. 303–338.
- Feichtenhofer, Christoph, Axel Pinz, and Andrew Zisserman (2016). “Convolutional two-stream network fusion for video action recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1933–1941.

- Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017). “Model-agnostic meta-learning for fast adaptation of deep networks”. In: *International conference on machine learning*. PMLR, pp. 1126–1135.
- Fort, Stanislav, Huiyi Hu, and Balaji Lakshminarayanan (2019). “Deep ensembles: A loss landscape perspective”. In: *Bayesian Deep Learning, NeurIPS 2019 Workshop*.
- Gal, Yarin and Zoubin Ghahramani (2016). “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *international conference on machine learning*. PMLR, pp. 1050–1059.
- Ghosh, A., V. Kulharia, and V. Namboodiri (2016). “Message Passing Multi-Agent GANs”. In: *arXiv preprint arXiv:1612.01294*.
- Ghosh, Aritra, Himanshu Kumar, and P Shanti Sastry (2017). “Robust loss functions under label noise for deep neural networks”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 31. 1.
- Ghosh, Arnab, Viveka Kulharia, Vinay P Namboodiri, Philip HS Torr, and Puneet K Dokania (2018). “Multi-agent diverse generative adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8513–8521.
- Glorot, Xavier and Yoshua Bengio (2010). “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*.
- Goodfellow, Ian (2016). “NIPS 2016 tutorial: Generative adversarial networks”. In: *arXiv preprint arXiv:1701.00160*.
- Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). “Generative adversarial nets”. In: *Advances in neural information processing systems*, pp. 2672–2680.
- Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger (2017). “On Calibration of Modern Neural Networks”. In: *ICML*.
- Hariharan, Bharath, Pablo Arbelaez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik (2011). “Semantic Contours from Inverse Detectors”. In: *International Conference on Computer Vision (ICCV)*.
- He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick (2017). “Mask R-CNN”. In: *International Conference on Computer Vision (ICCV)*.
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2014). “Spatial pyramid pooling in deep convolutional networks for visual recognition”. In: *ECCV*.
- (2016). “Deep Residual Learning for Image Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Hendrycks, Dan and Thomas Dietterich (2019). “Benchmarking Neural Network Robustness to Common Corruptions and Perturbations”. In: *arXiv preprint arXiv:1903.12261*.
- Hendrycks, Dan, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan (2020). “Augmix: A simple data processing method to improve robustness and uncertainty”. In: *ICLR*.

- Heo, Yuk, Yeong Jun Koh, and Chang-Su Kim (2020). “Interactive video object segmentation using global and local transfer modules”. In: *European Conference on Computer Vision*. Springer, pp. 297–313.
- Hou, Ji, Saining Xie, Benjamin Graham, Angela Dai, and Matthias Nießner (2021). “Pri3D: Can 3D Priors Help 2D Representation Learning?” In: *ICCV*.
- Hou, Qibin, Daniela Massiceti, Puneet Kumar Dokania, Yunchao Wei, Ming-Ming Cheng, and Philip HS Torr (2017). “Bottom-up top-down cues for weakly-supervised semantic segmentation”. In: *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer, pp. 263–277.
- Hsu, Kuang-Jui, Yen-Yu Lin, and Yung-Yu Chuang (2019). “DeepCO3: Deep Instance Co-Segmentation by Co-Peak Search and Co-Saliency Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Hu, Ronghang, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick (2018). “Learning to segment every thing”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4233–4241.
- Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger (2017). “Densely Connected Convolutional Networks”. In: *CVPR*.
- Huang, Zilong, Xinggang Wang, Jiasi Wang, Wenyu Liu, and Jingdong Wang (2018). “Weakly-supervised semantic segmentation network with deep seeded region growing”. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7014–7023.
- Hung, Wei-Chih, Yi-Hsuan Tsai, Yan-Ting Liou, Yen-Yu Lin, and Ming-Hsuan Yang (2018). “Adversarial Learning for Semi-Supervised Semantic Segmentation”. In: *British Machine Vision Conference (BMVC)*.
- Ioffe, Sergey and Christian Szegedy (2015). “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *International Conference on Machine Learning*. PMLR.
- Isola, P., J-Y. Zhu, T. Zhou, and A. Efros (2017). “Image-to-Image Translation with Conditional Adversarial Networks”. In: *Computer Vision and Pattern Recognition*.
- Izmailov, Pavel, Dmitrii Podoprikin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson (2018). “Averaging weights leads to wider optima and better generalization”. In: *UAI*.
- Jampani, Varun, Raghudeep Gadde, and Peter V. Gehler (2017). “Video Propagation Networks”. In: *CVPR*.
- Joachims, T., T. Finley, and C.-N. Yu (2009). “Cutting-plane Training of Structural SVMs”. In: *Machine Learning*.
- Juefei-Xu, Felix, Vishnu Naresh Boddeti, and Marios Savvides (2017). “Gang of GANs: Generative Adversarial Networks with Maximum Margin Ranking”. In: *arXiv preprint arXiv:1704.04865*.
- Kamilaris, Andreas and Francesc X Prenafeta-Boldú (2018). “Deep learning in agriculture: A survey”. In: *Computers and Electronics in Agriculture* 147, pp. 70–90.
- Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen (2018). “Progressive growing of gans for improved quality, stability, and variation”. In: *ICLR*.

- Karras, Tero, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila (2020). “Training Generative Adversarial Networks with Limited Data”. In: *NeurIPS*.
- Karras, Tero, Samuli Laine, and Timo Aila (2019). “A style-based generator architecture for generative adversarial networks”. In: *CVPR*, pp. 4401–4410.
- Ke, Tsung-Wei, Jyh-Jing Hwang, Ziwei Liu, and Stella X Yu (2018). “Adaptive affinity fields for semantic segmentation”. In: *European Conference on Computer Vision (ECCV)*.
- Ker, Justin, Lipo Wang, Jai Rao, and Tchoyoson Lim (2018). “Deep Learning Applications in Medical Image Analysis”. In: *IEEE Access* 6, pp. 9375–9389.
- Khoreva, Anna, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele (2017a). “Simple does it: Weakly supervised instance and semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 876–885.
- Khoreva, Anna, Rodrigo Benenson, Eddy Ilg, Thomas Brox, and Bernt Schiele (2017b). “Lucid Data Dreaming for Object Tracking”. In: *CVPR Workshops*.
- Kingma, D. and M. Welling (2014). “Auto-encoding variational bayes”. In: *International Conference on Learning Representations*.
- Kingma, Durk P and Prafulla Dhariwal (2018). “Glow: Generative flow with invertible 1x1 convolutions”. In: *Advances in Neural Information Processing Systems*, pp. 10215–10224.
- Koch, Gregory, Richard Zemel, and Ruslan Salakhutdinov (2015). “Siamese Neural Networks for One-shot Image Recognition”. In: *ICML Workshops*.
- Kolesnikov, Alexander, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *ICLR*.
- Krähenbühl, Philipp and Vladlen Koltun (2011). “Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials”. In: *Neural Information Processing Systems (NIPS)*.
- (2013). “Parameter Learning and Convergent Inference for Dense Random Fields”. In: *ICML*.
- Krizhevsky, Alex (2009). *Learning Multiple Layers of Features from Tiny Images*. Tech. rep. Department of Computer Science, University of Toronto.
- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 1097–1105.
- Kulharia, Viveka, Siddhartha Chandra, Amit Agrawal, Philip Torr, and Amrith Tyagi (2020). “Box2seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation”. In: *European Conference on Computer Vision*. Springer, pp. 290–308.
- Kulharia, Viveka, Arnab Ghosh, Amitabha Mukerjee, Vinay Namboodiri, and Mohit Bansal (2017). “Contextual RNN-GANs for Abstract Reasoning Diagram Generation”. In: *AAAI Conference on Artificial Intelligence*.
- Kull, Meelis, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach (2019). “Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with Dirichlet calibration”. In: *NeurIPS*.

- Kullback, Solomon and Richard A Leibler (1951). “On information and sufficiency”. In: *The annals of mathematical statistics*.
- Kumar, Ananya, Percy S Liang, and Tengyu Ma (2019). “Verified Uncertainty Calibration”. In: *NeurIPS*.
- Kumar, Aviral, Sunita Sarawagi, and Ujjwal Jain (2018). “Trainable Calibration Measures For Neural Networks From Kernel Mean Embeddings”. In: *ICML*.
- Kuznetsova, Alina, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Mallocci, Tom Duerig, et al. (2018). “The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale”. In: *arXiv preprint arXiv:1811.00982*.
- Kwak, Suha, Seunghoon Hong, and Bohyung Han (2017). “Weakly supervised semantic segmentation using superpixel pooling network”. In: *AAAI Conference on Artificial Intelligence*.
- Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell (2017). “Simple and scalable predictive uncertainty estimation using deep ensembles”. In: *Advances in neural information processing systems*, pp. 6402–6413.
- Lampinen, Jouko and Aki Vehtari (2001). “Bayesian approach for neural networks—review and case studies”. In: *Neural networks* 14.3, pp. 257–274.
- Lang, Ken (1995). “NewsWeeder: Learning to Filter Netnews”. In: *Machine Learning Proceedings 1995*. Elsevier, pp. 331–339.
- LeCun, Yann, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel (1989). “Backpropagation applied to handwritten zip code recognition”. In: *Neural computation*.
- Ledig, Christian, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi (2017). “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”. In: *Computer Vision and Pattern Recognition*.
- Lee, Jungbeom, Eunji Kim, Sungmin Lee, Jangho Lee, and Sungroh Yoon (Oct. 2019). “Frame-to-Frame Aggregation of Active Regions in Web Videos for Weakly Supervised Semantic Segmentation”. In: *International Conference on Computer Vision (ICCV)*.
- Li, Qizhu, Anurag Arnab, and Philip HS Torr (2018). “Weakly-and semi-supervised panoptic segmentation”. In: *Proc. European Conference on Computer Vision (ECCV 18)*, pp. 102–118.
- Li, X., Y. Qi, Z. Wang, K. Chen, Z. Liu, J. Shi, P. Luo, C. Change Loy, and X. Tang (2017). “Video Object Segmentation with Re-identification”. In: *CVPR Workshops*.
- Lin, Di, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun (2016). “Scribblesup: Scribble-supervised convolutional networks for semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3159–3167.
- Lin, Min, Qiang Chen, and Shuicheng Yan (2014a). “Network In Network”. In: *ICLR*.
- Lin, Tsung-Yi, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár (2017). “Focal Loss for Dense Object Detection”. In: *ICCV*.

- Lin, Tsung-Yi, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (2014b). “Microsoft COCO: Common Objects in Context”. In: *CoRR* abs/1405.0312.
- Lin, Zinan, Ashish Kheta, Giulia Fanti, and Sewoong Oh (2018). “Pacgan: The power of two samples in generative adversarial networks”. In: *Advances in neural information processing systems* 31.
- Liu, Ming-Yu and Oncel Tuzel (2016). “Coupled generative adversarial networks”. In: *Advances in neural information processing systems*.
- Liu, Sifei, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz (2017). “Learning affinity via spatial propagation networks”. In: *Neural Information Processing Systems (NIPS)*.
- Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). “Fully convolutional networks for semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440.
- Lowe, David G (2004). “Distinctive image features from scale-invariant keypoints”. In: *IJCV*.
- Ma, Xingjun, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey (2020). “Normalized loss functions for deep learning with noisy labels”. In: *International conference on machine learning*. PMLR, pp. 6543–6553.
- Maddox, Wesley J, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson (2019). “A simple baseline for bayesian uncertainty in deep learning”. In: *NeurIPS*.
- Maire, Michael, Takuya Narihira, and Stella X Yu (2016). “Affinity cnn: Learning pixel-centric pairwise relations for figure/ground embedding”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Mandt, Stephan, Matthew D Hoffman, and David M Blei (2017). “Stochastic gradient descent as approximate bayesian inference”. In: *arXiv preprint arXiv:1704.04289*.
- McIntire, John P, Paul R Havig, and Eric E Geiselman (2014). “Stereoscopic 3D displays and human performance: A comprehensive review”. In: *Displays* 35.1, pp. 18–26.
- Metz, L., B. Poole, D. Pfau, and J. Sohl-Dickstein (2017). “Unrolled Generative Adversarial Networks”. In: *International Conference on Learning Representations*.
- Min, Shaobo, Xuejin Chen, Zheng-Jun Zha, Feng Wu, and Yongdong Zhang (2018). *A Two-Stream Mutual Attention Network for Semi-supervised Biomedical Segmentation with Noisy Labels*. arXiv: 1807.11719 [cs.CV].
- Mirza, M. and S. Osindero (2014). “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784*.
- Mukhoti, Jishnu, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania (2020). “Calibrating deep neural networks using focal loss”. In: *Advances in Neural Information Processing Systems* 33, pp. 15288–15299.
- Müller, Rafael, Simon Kornblith, and Geoffrey Hinton (2019). “When Does Label Smoothing Help?” In: *NeurIPS*.
- Naeini, Mahdi Pakdaman and Gregory F Cooper (2016). “Binary Classifier Calibration using an Ensemble of Near Isotonic Regression Models”. In: *ICDM*.

- Naeini, Mahdi Pakdaman, Gregory F Cooper, and Milos Hauskrecht (2015). “Obtaining Well Calibrated Probabilities Using Bayesian Binning”. In: *AAAI*.
- Najafi, Mohammad, Viveka Kulharia, T Ajanthan, and PH Torr (2018). “Similarity learning for dense label transfer”. In: *The 2018 DAVIS Challenge on Video Object Segmentation-CVPR Workshops*.
- Neal, Radford M and Geoffrey E Hinton (1998). “A view of the EM algorithm that justifies incremental, sparse, and other variants”. In: *Learning in graphical models*. Springer, pp. 355–368.
- Netzer, Y., T. Wang, A. Coates, A. Bissacco, B. Wu, and Andrew Y. Ng (2011). “Reading Digits in Natural Images with Unsupervised Feature Learning”. In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.
- Nguyen, Khanh and Brendan O’Connor (2015). “Posterior calibration and exploratory analysis for natural language processing models”. In: *arXiv preprint arXiv:1508.05154*.
- Niculescu-Mizil, Alexandru and Rich Caruana (2005). “Predicting Good Probabilities With Supervised Learning”. In: *ICML*.
- Noh, H., S. Hong, and B. Han (2015). “Learning deconvolution network for semantic segmentation”. In: *arXiv preprint arXiv:1505.04366*.
- Ovadia, Yaniv, Emily Fertig, Jie Ren, Zachary Nado, D Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek (2019). “Can You Trust Your Model’s Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift”. In: *NeurIPS*.
- Papadopoulos, Dim P, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari (2017). “Extreme clicking for efficient object annotation”. In: *IEEE International Conference on Computer Vision (CVPR)*, pp. 4930–4939.
- Papandreou, George, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille (2015). “Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1742–1750.
- Parkhi, O. M., A. Vedaldi, and A. Zisserman (2015). “Deep face recognition”. In: *BMVC*.
- Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). “Automatic differentiation in PyTorch”. In: *NIPS 2017 Workshop Autodiff Submission*.
- Pathak, D., P. Krahenbuhl, J. Donahue, T. Darrell, and Alexei A. Efros (2016). “Context Encoders: Feature Learning by Inpainting”. In: *Computer Vision and Pattern Recognition*.
- Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). “GloVe: Global Vectors for Word Representation”. In: *Empirical Methods in Natural Language Processing (EMNLP)*.
- Perazzi, F., A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung (2017). “Learning Video Object Segmentation from Static Images”. In: *Computer Vision and Pattern Recognition*.
- Perazzi, F., J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung (2016). “A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation”. In: *CVPR*.

- Pereyra, Gabriel, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton (2017). “Regularizing Neural Networks by Penalizing Confident Output Distributions”. In: *arXiv preprint arXiv:1701.06548*.
- Petit, Olivier, Nicolas Thome, Arnaud Charnoz, Alexandre Hostettler, and Luc Soler (2018). “Handling Missing Annotations for Semantic Segmentation with Deep ConvNets”. In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, pp. 20–28.
- Platt, John (1999). “Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods”. In: *Advances in Large Margin Classifiers* 10.3, pp. 61–74.
- Pont-Tuset, Jordi, Pablo Arbelaez, Jonathan T Barron, Ferran Marques, and Jitendra Malik (2016). “Multiscale combinatorial grouping for image segmentation and object proposal generation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 39.1, pp. 128–140.
- Pont-Tuset, Jordi, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool (2017). “The 2017 DAVIS Challenge on Video Object Segmentation”. In: *arXiv:1704.00675*.
- Radford, A., L. Metz, and S. Chintala (2015). “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434*.
- Rahaman, Rahul and Alexandre Thiery (2021). “Uncertainty Quantification and Deep Ensembles”. In: *NeurIPS*. Vol. 34, pp. 20063–20075.
- Rajchl, Martin, Matthew CH Lee, Ozan Oktay, Konstantinos Kamnitsas, Jonathan Passerat-Palmbach, Wenjia Bai, Mellisa Damodaram, Mary A Rutherford, Joseph V Hajnal, Bernhard Kainz, et al. (2016). “Deepcut: Object segmentation from bounding box annotations using convolutional neural networks”. In: *IEEE transactions on medical imaging* 36.2, pp. 674–683.
- Redondo-Cabrera, Carolina, Marcos Baptista-Ríos, and Roberto J López-Sastre (2019). “Learning to Exploit the Prior Network Knowledge for Weakly Supervised Semantic Segmentation”. In: *IEEE Transactions on Image Processing* 28.7, pp. 3649–3661.
- Reed, Scott, Zeynep Akata, Xinchun Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee (2016). “Generative Adversarial Text to Image Synthesis”. In: *International Conference on Machine Learning*.
- Rezende, Danilo and Shakir Mohamed (2015). “Variational inference with normalizing flows”. In: *International conference on machine learning*. PMLR, pp. 1530–1538.
- Richter, Stephan R, Zeeshan Hayder, and Vladlen Koltun (2017). “Playing for benchmarks”. In: *International Conference on Computer Vision (ICCV)*.
- Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention*.
- Rother, Carsten, Vladimir Kolmogorov, and Andrew Blake (2004). “Grabcut: Interactive foreground extraction using iterated graph cuts”. In: *ACM transactions on graphics (TOG)*. Vol. 23. ACM, pp. 309–314.

- Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. (2015). “Imagenet large scale visual recognition challenge”. In: *IJCV*.
- Salimans, Tim, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen (2016). “Improved techniques for training gans”. In: *Advances in Neural Information Processing Systems*.
- Selvaraju, Ramprasaath R, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra (2017). “Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization”. In: *ICCV*.
- Shaban, A., A. Firl, A. Humayun, J. Yuan, X. Wang, P. Lei, N. Dhanda, B. Boots, J. M. Rehg, and F. Li (2017). “Multiple-Instance Video Segmentation with Sequence-Specific Object Proposals”. In: *CVPR Workshops*.
- Shi, Jianbo and Jitendra Malik (2000). “Normalized cuts and image segmentation”. In: *PAMI*.
- Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. (2016). “Mastering the game of Go with deep neural networks and tree search”. In: *nature* 529.7587, pp. 484–489.
- Simonyan, Karen and Andrew Zisserman (2014a). “Two-stream convolutional networks for action recognition in videos”. In: *Advances in neural information processing systems* 27.
- (2014b). “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *arXiv preprint arXiv:1409.1556*.
- Singh, Bharat, Mahyar Najibi, and Larry S Davis (2018). “SNIPER: Efficient Multi-Scale Training”. In: *Neural Information Processing Systems (NIPS)*.
- Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts (2013). “Recursive Deep Models for Semantic Compositional-ity Over a Sentiment Treebank”. In: *Empirical Methods in Natural Language Processing (EMNLP)*.
- Song, Chunfeng, Yan Huang, Wanli Ouyang, and Liang Wang (2019). “Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3136–3145.
- Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014). “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1, pp. 1929–1958.
- Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna (2016). “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.
- Tai, Kai Sheng, Richard Socher, and Christopher D Manning (2015). “Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks”. In: *Association for Computational Linguistics (ACL)*.

- Tan, Mingxing and Quoc Le (2019). “Efficientnet: Rethinking model scaling for convolutional neural networks”. In: *International conference on machine learning*. PMLR, pp. 6105–6114.
- Tang, Meng, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers (2018a). “Normalized cut loss for weakly-supervised cnn segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1818–1827.
- Tang, Meng, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov (2016). “Normalized cut meets MRF”. In: *European Conference on Computer Vision*. Springer, pp. 748–765.
- Tang, Meng, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov (2018b). “On regularized losses for weakly-supervised cnn segmentation”. In: *European Conference on Computer Vision (ECCV)*, pp. 507–522.
- Tesla (2021). *Tesla AI Day*. URL: <https://www.youtube.com/watch?v=j0z4FweCy4M&t=2900s>.
- Thulasidasan, Sunil, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak (2019). “On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks”. In: *NeurIPS*.
- Train CIFAR10 with PyTorch* (2019). Available online at <https://github.com/kuangliu/pytorch-cifar> (as of 22nd May 2019).
- TreeLSTM* (2015). <https://github.com/ttpro1995/TreeLSTMSentiment>. Accessed: 2019-05-22.
- Tsochantaridis, I., T. Hofmann, T. Joachims, and Y. Altun (2004). “Support vector machine learning for interdependent and structured output spaces”. In: *International Conference on Machine Learning*.
- Vaicenavicius, Juozas, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas B Schön (2019). “Evaluating model calibration in classification”. In: *AISTATS*.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). “Attention Is All You Need”. In: *NeurIPS*.
- Vinyals, Oriol, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra (2016). “Matching Networks for One Shot Learning”. In: *NIPS*.
- Voigtlaender, Paul and Bastian Leibe (2017). “Online Adaptation of Convolutional Neural Networks for Video Object Segmentation”. In: *BMVC*.
- Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba (2016). “Generating videos with scene dynamics”. In: *Advances In Neural Information Processing Systems*.
- Wang, Bin, Guojun Qi, Sheng Tang, Tianzhu Zhang, Yunchao Wei, Linghui Li, and Yongdong Zhang (2019). “Boundary perception guidance: a scribble-supervised semantic segmentation approach”. In: *International Joint Conference on Artificial Intelligence (IJCAI)*.
- Wang, Jinjiang, Yulin Ma, Laibin Zhang, Robert X Gao, and Dazhong Wu (2018a). “Deep learning for smart manufacturing: Methods and applications”. In: *JMSY* 48, pp. 144–156.
- Wang, Xiang, Shaodi You, Xi Li, and Huimin Ma (2018b). “Weakly-supervised semantic segmentation by iteratively mining common object features”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1354–1362.

- Wen, Yeming, Ghassen Jerfel, Rafael Muller, Michael W Dusenberry, Jasper Snoek, Balaji Lakshminarayanan, and Dustin Tran (2021). “Combining ensembles and data augmentation can harm your calibration”. In: *ICLR*.
- Widmann, David, Fredrik Lindsten, and Dave Zachariah (2019). “Calibration tests in multi-class classification: A unifying framework”. In: *NeurIPS*.
- Wong, Eric, Shibani Santurkar, and Aleksander Mądry (2021). “Leveraging Sparse Linear Layers for Debuggable Deep Networks”. In: *ICML*.
- Wu, Jiajun, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum (2016). “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling”. In: *Advances in Neural Information Processing Systems*.
- Xiao, Tete, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun (2018). “Unified perceptual parsing for scene understanding”. In: *European Conference on Computer Vision (ECCV)*, pp. 418–434.
- Xu, Ning, Brian Price, Scott Cohen, Jimei Yang, and Thomas Huang (Sept. 2017). “Deep GrabCut for Object Selection”. In: *BMVC*.
- Yu, Fisher and Vladlen Koltun (2016). “Multi-scale context aggregation by dilated convolutions”. In: *ICLR*.
- Zadrozny, Bianca and Charles Elkan (2001). “Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers”. In: *ICML*.
- (2002). “Transforming Classifier Scores into Accurate Multiclass Probability Estimates”. In: *SIGKDD*.
- Zagoruyko, Sergey and Nikos Komodakis (2016). “Wide Residual Networks”. In: *BMVC*.
- Zeng, Yu, Yunzhi Zhuge, Huchuan Lu, and Lihe Zhang (2019). “Joint learning of saliency detection and weakly supervised semantic segmentation”. In: *International Conference on Computer Vision (ICCV)*.
- Zhang, Hang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Amrith Tyagi, and Amit Agrawal (June 2018a). “Context Encoding for Semantic Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Zhang, Hang, Han Zhang, Chenguang Wang, and Junyuan Xie (2019). “Co-occurent Features in Semantic Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Zhang, Hongyi, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz (2018b). “mixup: Beyond empirical risk minimization”. In: *ICLR*.
- Zhang, Zhilu and Mert Sabuncu (2018). “Generalized cross entropy loss for training deep neural networks with noisy labels”. In: *Advances in neural information processing systems* 31.
- Zhao, Hengshuang, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia (2017). “Pyramid Scene Parsing Network”. In: *CVPR*.
- Zhao, Xiangyun, Shuang Liang, and Yichen Wei (2018). “Pseudo mask augmented object detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4061–4070.

- Zheng, Shuai, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr (2015). “Conditional Random Fields as Recurrent Neural Networks”. In: *International Conference on Computer Vision (ICCV)*.
- Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba (2016). “Learning deep features for discriminative localization”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2921–2929.
- Zhou, Bolei, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba (2017a). “Places: A 10 million Image Database for Scene Recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Zhou, Bolei, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba (2017b). “Scene Parsing through ADE20K Dataset”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Zhu, Jun-Yan, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros (2016). “Generative visual manipulation on the natural image manifold”. In: *European Conference on Computer Vision*.
- Zhu, Yi, Yanzhao Zhou, Huijuan Xu, Qixiang Ye, David Doermann, and Jianbin Jiao (2019). “Learning instance activation maps for weakly supervised instance segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3116–3125.