Use and Examination of Convolutional Neural Networks for Scene Understanding

Saumya Jetley

St. Cross College

University of Oxford

A thesis submitted for the degree of

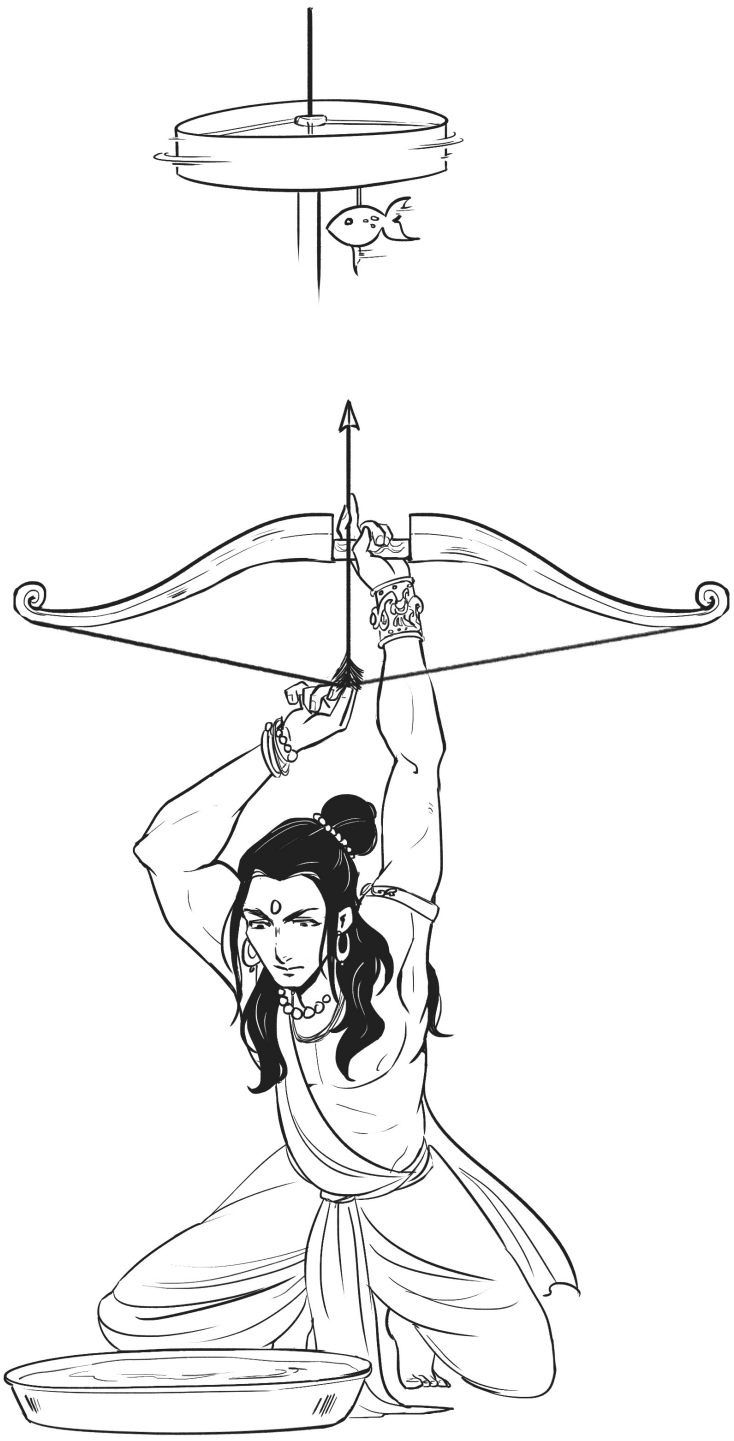
Doctor of Philosophy

Michaelmas 2018

To mummy and papa,
for teaching me to believe and to strive.

कर्मण्येवाधिकारस्ते मा फलेषु कदाचन।

- गीता उपदेश (२-४७)



Arjuna aiming for the fish's eye by looking at its reflection in the water at *Draupadi's swayamvara*. (Author's note: This imagery from the Indian epic *Mahabharata* epitomises greatness in skill and focus. In me, additionally, it evokes the spirit of true scientific investigation - the single-minded pursuit of the complete truth via partial and noisy observations from indirect inquiries into the real-world processes.)

Contents

List of Figures

List of Tables

List of Abbreviations

AP	Average Precision.
BL	Belga Logos.
CIFAR	Canadian Institute For Advanced Research.
CNN	Convolutional Neural Network.
CUB	Caltech-UCSD Birds.
FGSM	Fast Gradient Sign Method.
GTS	German Traffic Sign.
ILSVRC	ImageNet Large Scale Visual Recognition Challenge.
IoU	Intersection over Union.
mAP	Mean Average Precision.
MIT	Massachusetts Institute of Technology.
MNIST	Modified National Institute of Standards and Technology.
N**D**	N-dimensional, where N is any scalar.
NiN	Network-in-Network.
OSIE	Object and Semantic Images and Eye-tracking.
PASCAL VOC	Pattern Analysis, Statistical Modelling and Computational Learning - Visual Object Classes.
SALICON	Saliency in Context.
SBD	Semantic Boundaries Dataset.
STS	Straight To Shapes.
SVD	Singular Value Decomposition.
SVHN	Street View House Numbers.
UAP(s)	Universal Adversarial Perturbation(s).
UCSD	University of California San Diego.
VGG	Visual Geometry Group.
VOCA-2012	Visual Object Classes for Actions - 2012

Glossary

AlexNet . . . . . The first deep convolutional neural network architecture to be successfully trained for classifying ImageNet images.

$\mathbf{AP_T}$ or $\mathbf{AP_T^b}$ . . . . . Area under the precision-recall curve for an IoU threshold of $T \in (0, 1)$, where the IoU is measured between ground-truth and predicted bounding boxes.

$\mathbf{AP_T^r}$ . . . . . Area under the precision-recall curve for an IoU threshold of $T \in (0, 1)$, where the IoU is measured between ground-truth and predicted regions.

BL . . . . . An image dataset of logos and trademarks obtained from the recordings provided by BELGA news agency.

CIFAR-10 . . . . . A large-scale dataset of 32×32 images divided into 10 object categories.

CIFAR-100 . . . . . A large-scale dataset of 32×32 images divided into 100 object categories.

CUB-200 . . . . . An image dataset with photos of 200 bird species (mostly North American).

Darknet19 . . . . . A deep convolutional network architecture for real time object detection.

FGSM . . . . . An approximate gradient-based adversarial attack approach.

GTS . . . . . German traffic sign recognition benchmark.

ImageNet . . . . . A large-scale hierarchical visual database designed for use in visual object recognition software research.

$\mathbf{mAP}$ or $\mathbf{mAP^b}$ . . . . . Mean of the average precision over all classes and/or IoU thresholds for bounding box predictions.

$\mathbf{mAP^r}$ . . . . . Mean of the average precision over all classes and/or IoU thresholds for predicted regions.

MIT-300 . . . . . A human saliency benchmark dataset containing 300 natural images with eye tracking data from 39 observers.

MNIST . . . . . A large-scale dataset of handwritten digits from 0 to 9.

NiN . . . . . A deep convolutional network architecture for image classification comprising a hierarchy of complex micro neural networks.

OSIE . . . . . An eye-gaze dataset recorded using the eye tracking camera EyeLink 1000.

PASCAL VOC . . . . . A dataset of images obtained from *flickr* and annotated for object class recognition and detection.

PASCAL VOC 2007 . . . Train, validation and test sets used for the PASCAL VOC challenge conducted in 2007.

PASCAL VOC 2012 . . . Train, validation and test sets used for the challenge conducted in 2012 containing additional images and annotations over PASCAL VOC 2007.

SALICON . . . . . A large-scale eye-gaze dataset collected using a mouse based paradigm.

SBD . . . . . An instance segmentation benchmark providing object boundary annotations for images from PASCAL VOC 2012.

STS . . . . . A CNN based approach for real-time instance segmentation proposed in this thesis.

SVHN . . . . . A natural image dataset of numbers 0-9 collected from Google Street View images.

UAP . . . . . A single perturbation image that can cause a large percentage of input images to be misclassified by a network.

val set . . . . . Validation set, a set aside portion of a given dataset used for tuning the hyperparamters of a model.

VGG . . . . . A popular deep convolutional network architecture for image classification.

VOCA-2012 . . . . . A human eye movement dataset collected under the primary task constraint of action recognition.

Acknowledgements

Personal

I would like to extend a humble thanks to my supervisor Prof. Philip Torr for giving me the opportunity to pursue my PhD in the beautiful city of Oxford. I have grown in numerous ways during my time here, through my association with the excellence, fervour and positive scepticism that defines the scientific tradition of the University of Oxford. And, by working alongside some great scientific minds here at the Torr Vision Group including Phil himself, Nicholas A. Lord, Michael Sapienza, Bernardino Romera Paredes, Sadeep Jayasumana and Stuart Golodetz. A special cheers to Nicholas for being a mentor, a critic and for challenging me the way he did.

Without Daniela the lab wouldn't have been half the home that it has been over the past 4 years. She has been a friend, a supporter and a confidante. Rodrigo, Luca, Anurag, AJ, Tommaso, Puneet, Christian, Viveka, Arnab and Harkirat have all brightened my days with their conversations on many different occasions. First Maddy and then Cassandra graciously and gracefully helped with the administrative work. Steven Hudspith was my go-to man for all the financial queries and always did he resolve them impeccably.

Last but not the least, a big thank-you to mummy, papa and Divya for being my pillars of support and the ideals I could always look up to, and to Miguel for his unlimited reserves of patience and his confidence whenever I faltered.

Institutional

Thanks again to Phil, the European Research Council, and the Department of Engineering Science for taking care of all my financial and administrative worries so I could focus on my studies while at Oxford. A big shout out to my internal examiners Prof. Andrew Zisserman and Prof. Paul Newman under whose watchful eye I always felt challenged and confident. Thanks also to Xerox Research Centre Europe (now Naver Labs) for giving me the chance to work alongside the two most smart and inspiring women in tech - Dr. Naila Murray and Dr. Eleonora Vig, that I know. Thank you all for motivating me and for believing in me.

Abstract

This thesis concerns itself with the use and examination of convolutional neural networks in the context of visual scene understanding. Towards this, **the first part of the thesis proposes novel extensions to vanilla CNNs**. These extensions attempt to incorporate domain knowledge into the computational framework of CNNs in order to better adapt them to targeted visual tasks. We begin by integrating a class prototypical embedding space in a conventional classification network, whereby real world object samples are recognised by matching them to correct class visual prototypes in this space. This use of side information is not only able to improve the network classification performance on the categories seen during training but also boosts the recognition of similar categories that are unseen during training. Likewise, we propose a deep neural model for real time instance segmentation that makes use of an intermediate shape embedding space. This continuous and learned latent space allows unseen input object images to be matched to new and realistic shape masks at test time. In the follow-up work, we draw inspiration from the recent advances in network design and training for object detection and segmentation. The assimilation of these techniques in our instance segmentation system allows us to further improve its accuracy while still operating in real time. In yet another application of CNNs, to the task of human saliency estimation, we revisit the interpretation of the task as a competitive process: humans look at some regions of an image at the cost of not looking at others. We thus model the output saliency maps as spatial probability distributions, and propose the use of losses that are suitable for measuring distances between probability distributions to train a deep network for the task. This formulation yields significant gains in terms of the network predictive performance as measured on an array of saliency metrics.

After augmenting and applying conventional CNNs to a variety of visual tasks, **the later part of the thesis shifts its focus to an examination of these networks.**We begin by investigating classification CNNs at a qualitative level by modelling network visual attention. In particular, we formulate attention modules that can be trained in tandem with the network weights to optimise the end goal of image classification. The resulting spatial attention scores, associated with the local features at predefined network layers, are able to identify the semantic parts of the input images. In other words, the attention maps are able to suppress the irrelevant and highlight the relevant regions of the input images in a way that lends greater transparency to the inference procedure of nets and

also boosts the output classification accuracy. Further, the binarised maps serve as useful weakly-supervised foreground segmentation masks. We then perform a more principled analysis of the class decision functions learned by classification CNNs by contextualising an existing geometrical framework for network decision boundary analysis. Our research uncovers some very intriguing yet simplistic facets of the class score functions learned by these networks that explain their adversarial vulnerability. We identify the fact that specific input image space directions tend to be associated with fixed class identities. This means that simply increasing the magnitude of correlation between an input image and a single image space direction causes the nets to believe that more (or less) of the class is present. This allows us to provide a new perspective on the existence of universal adversarial perturbations. Further, the input image space directions which the networks use to achieve their classification performance are the same along which they are most vulnerable to attack; the vulnerability arises from the rather simplistic non-linear use of the directions. Thus, as it stands, the performance and vulnerability of these nets are closely entwined. Various notable observations emerge from this, one of which is that any attempt to make a trained network more robust to an adversarial attack by suppressing input data dimensions or intermediate network features would always be accompanied by a corresponding loss in network accuracy. Moreover, if the attack is optimised to take into account the suppression of dimensions, it regains its effectiveness. These results present key implications for future efforts to construct neural nets that are both accurate and robust to adversarial attack.

The codebase needed to reproduce the above described experimental results and observations has been made available through `www.robots.ox.ac.uk/~sjetley`. We conclude by discussing some of the ways in which this work has been interpreted, used and extended, and by highlighting some open questions in an effort to inform future work in the field.

You might say it's obvious or common sense, but generations of scientists have struggled to articulate that common sense formally, and a robot cannot rely on our common sense when asked to act properly.

– Judea Pearl

1 Introduction

Contents

Consider this, you have accidentally spilled tea on your work desk. You notice some important papers that you immediately move out of the way of harm. The tea cannot be allowed to drip onto the floor for the fear of staining the carpet. You remember the paper towels in the desk drawer, open the drawer, quickly pull out a few of the towels and proceed to wipe the spill, cautiously keeping clear of the nearby objects. Now compare this with another experience. You turn onto a street and notice the sign-board for the museum you are headed to. You walk for a few more minutes, then look across the street, and recognise the entrance to the museum. With the intention of crossing the road over to the museum, you approach the pedestrian crossing and press the cross-walk button.

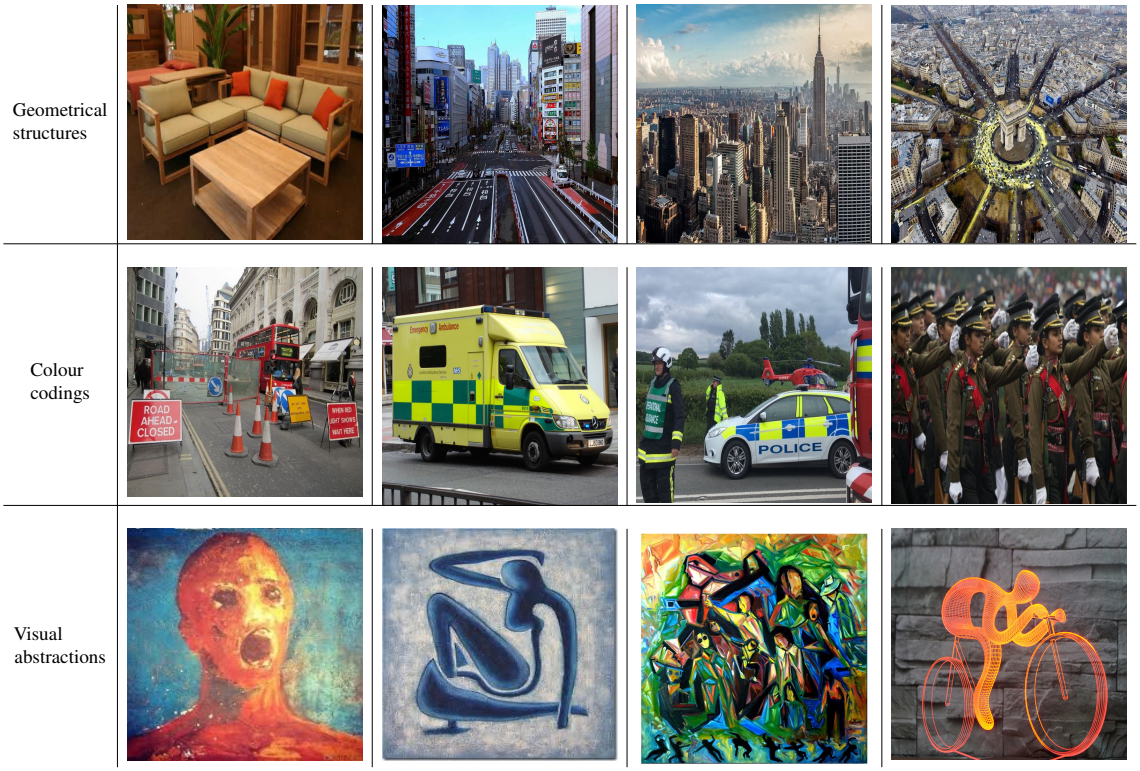


Figure 1.1: The man-made world is to a large extent defined by human visual perception. Colours of objects and their individual and group geometry are salient to human vision. (*Photo credits: the World Wide Web.*)

One might wonder if there is any commonality between the two ordinary situations described above. I believe there is at least one.

1.1 Motivating Interest in Scene Understanding

In the two cases illustrated above, similar to a vast majority of other everyday situations, an important commonality is our extreme reliance on the sense of vision to successfully accomplish the tasks involved. In fact, research studies have argued that not just human activity, but a significant proportion of human learning and cognition also proceeds through vision (Brooks & Meltzoff, 2005; Tacca, 2011). Human vision has been critical to the survival of our species through the predatory environment of the past and has, in turn, significantly shaped the man-made world of today. Looking around one can

easily notice a large number of artificially introduced visual regularities that surround us today. The household furniture, buildings, roads, automobiles and aeroplanes all have fairly regular geometrical structures. Buildings, cars, buses, road signs, traffic lights and duty uniforms are often color coded in order to denote specific associated functions. Even the different artistic representations of the same object exhibit an underlying visual consistency. A small sampling of the above described visual regularities is on display in Figure 1.1. Thus, if we are to develop machines that can replace or simply assist humans in everyday tasks of general intelligence, they need to be equipped with a level of visual scene understanding at par with that of humans' themselves. More specifically, the new fleet of household robots, industrial robots, autonomous driving agents and drones requires scene understanding capabilities in order to navigate and act successfully. The exponentially growing visual traffic on the internet courtesy of Facebook, Youtube, Flickr, Instagram, Amazon is dependent on this technology to better organise, retrieve, recommend and eventually make use of this visual data.

1.2 Historical Background

In the year 1966, Seymour Papert published an internal memo while at the Artificial Intelligence group of MIT. It was titled 'The Summer Vision Project' (Papert, 1966). This memo describes the proposal of hiring a team of graduate students to solve the tasks of image foreground-background separation and object identification over the summer. What was planned for one summer is, till date, an open and core challenge in the field of computer vision. The ability to parse and navigate a given visual environment is mere common sense reasoning for humans. However, it has taken scientists many decades of work to formalise this common sense reasoning into a set of instructions that machines can follow. And, the problem is still largely unsolved today.

1.3 What comprises Scene Understanding?

Marr (1982) summed up scene understanding as the task of establishing the 'what and where' of a particular scene. Thus, given a visual environment, represented using a single

image or a set of images, the first step is to identify meaningful elements within the scene, where these elements may be scene objects or surfaces of interest. This step is closely tied to an inference of the structure and properties of the individual elements, their functional affordances, intents and inter-relations, and ultimately the 3D layout of the entire scene.

Clearly, an extensive understanding of a scene relies on the successful resolution of numerous distinct properties of the scene as illustrated in Fig. 1.2. Thus, since the inception of the field of computer vision, various benchmark datasets and tasks have been advanced that focus on resolving these individual scene properties. A sampling of such tasks includes object recognition, object localisation, semantic segmentation, instance segmentation, pedestrian detection, human action recognition, face identification, depth estimation, texture recognition, and object discovery.

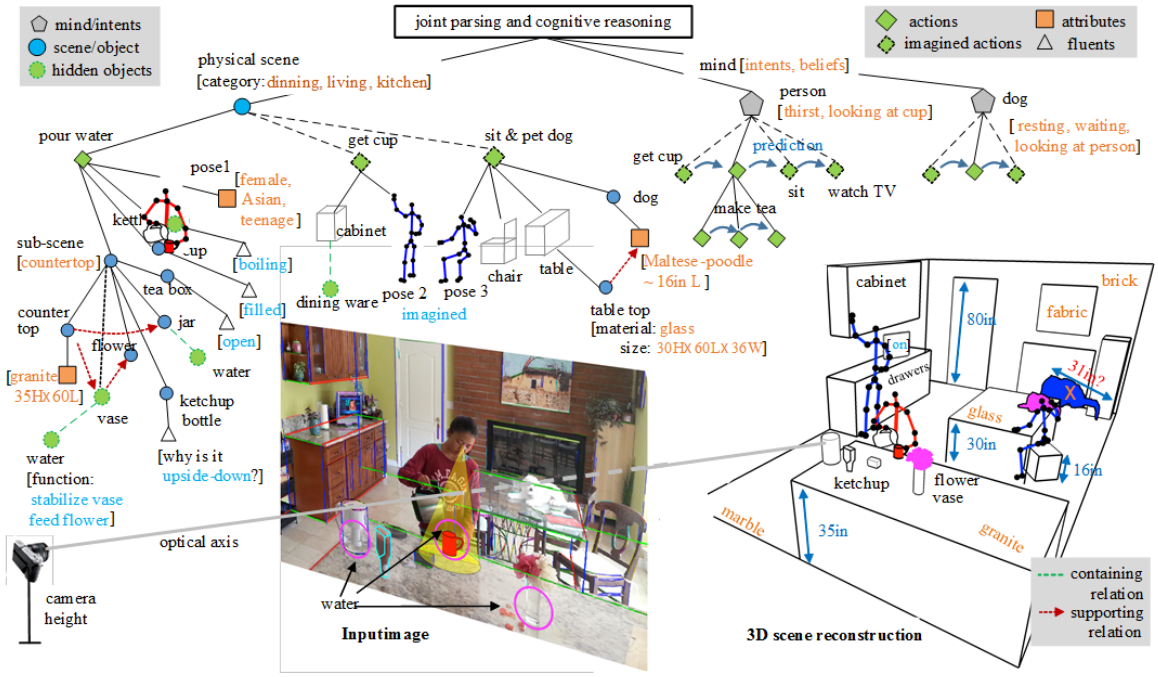


Figure 1.2: The task of scene understanding subsumes a diverse set of sub-tasks in order to establish 'what is where' in a given visual environment. (*Created by Prof. Song-Chun Zhu for MURI 2015 project grant proposal (Zhu, 2017).*)

In order to infer the above discussed properties of a scene, specialised computational functions need to be applied to the input image data. Consider the standard task of image classification. For simplicity, we assume that the object of interest is centrally located in the image and occupies a major portion of the 2D image. This task requires a

computational function $f_c : I \mapsto p_c$, that takes as input an image $I \in \mathbb{R}^d$ where d is the data dimensionality given by the number of image pixels $height \times width \times channels$, and outputs a categorical probability distribution p_c over N discrete target classes $\{1, 2, \cdots N\}$ s.t. $p_c \in [0, 1]^N$ and $\sum_t p_c(t) = 1 \quad \forall \ t \in \{1, 2, \cdots N\}$. As advanced by Marr (1982), it becomes tractable to think of this computational function as a two step process. The first is a standard information processing step that, via a series of transformations, maps the input data to a space where it is disentangled in terms of the underlying explanatory factors. This can be followed by any suitable predictor that uses these disentangled representations to produce the output distribution.

The above analytical framework allows us to appreciate the importance of data representations. The axes of such a representation space correspond to the intrinsic 'explanatory factors' or 'features' of the data that organise the data points so as to facilitate downstream classification or regression tasks. Thus, **a vast majority of computer vision literature is dedicated to identifying new and improved features or representation spaces for disentangling visual data to make it amenable to the vast variety of scene understanding tasks at hand.**

1.4 Visual Representations for Scene Understanding

The representational framework used by the vision community has progressed significantly over the last decade, a brief review is as follows.

1.4.1 Heuristic Models of Visual Representations

The period before the last decade was largely the era of handcrafted feature representations. Feature extractors such as wavelet-based analysis (Mallat, 1996), Scale Invariant Feature Transform (Lowe, 2004), Histogram of Oriented Gradients (Dalal & Triggs, 2005), Local Binary Patterns (Ahonen et al., 2006; Heikkilä et al., 2009), Speeded-Up Robust Features (Bay et al., 2008) were manually designed to supply representations that fit the needs of specific downstream tasks such as object recognition, image retrieval, facial analysis, pedestrian detection, character recognition to name a few.

1.4.2 Learned Models of Visual Representations

LeCun et al. (1990) presented one of the first successful neural network based models, learned entirely from the input data, for the task of digit recognition. This model was inspired by a decade old architecture called *neocognitron* (Fukushima, 1980) and its parameter optimisation was based on the principle of back-propagation (Rumelhart et al., 1988). The availability of big data courtesy of the internet revolution and increased computational power made it possible to combine these two techniques to create a wide array of practical use cases for neural networks. Further, Krizhevsky et al. (2012a) unlocked the larger potential of these networks by successfully applying them to the task of real world image classification. They made use of convolutional units for parsimony in parameters and dropout for regularisation. On real-sized images of 1000-category ImageNet dataset (Russakovsky* et al., 2015), their 'deep' network achieved an 8% reduction in the top-1 error as compared to the then state-of-art model based on Fisher vectors for the task of image classification. Since then **deep convolutional neural nets have become a staple for representation learning across the breadth of visual scene understanding tasks.**

1.5 Research Agenda

My research deals with the use and examination of convolutional neural networks applied to the task of scene understanding. I **began by focusing on techniques to incorporate domain knowledge into convolutional neural networks** with the dual motive of achieving performance gains and improving the interpretability of these models.

As I **progressed in my research work, I became more interested in interpreting and understanding the internal workings** of these models. Around the same time, discussions about the adversarial vulnerability of classification CNNs (Szegedy et al., 2014; Goodfellow et al., 2015; Tanay & Griffin, 2016; Moosavi-Dezfooli et al., 2016; Fawzi* et al., 2017; Moosavi-Dezfooli* et al., 2017, 2018; NIPS, 2017) and concerns about their true generalisation (Zhang et al., 2017) were gaining momentum. The former refers to the phenomenon whereby a systematic perturbation to an image, while leaving

its human recognisability unaffected, causes CNNs to misclassify the resulting image with high-confidence. These behavioural tendencies of CNNs are critically linked to the nature of the functions implemented by the learned networks. Thus, we then undertook a more principled study involving the geometric analysis of the class decision functions implemented by these nets. A more detailed discussion of the research work conducted under each of the areas mentioned above is as follows.

1.5.1 Use of CNNs for Scene Understanding

In this area, we start by **redesigning a conventional deep classification pipeline such that it implements a denoising function which is learned to match real world occurrences of objects to their prototypical templates** (Jetley et al., 2015). In other words, we train a CNN to map the real world occurrence of, say a traffic sign, to a predetermined representation of its standardised template (referred here as the prototypical template) in a way that minimises the empirical risk at the end task of sign classification. We observe that this integration of the prototypical information into the training pipeline affords a stronger supervision, which improves the test time recognition accuracy on the classes seen during training. Further, **by the use of this prototypical embedding as an intermediate disentangling space, we are also able to extend the inference to unseen classes**. In particular, we study the benefits at the task of zero-shot recognition of traffic signs and brand logos - object categories where the class prototypes are easily available. More notably, the practical applicability of the model proposed above goes beyond the specific use-case of traffic-sign or brand-logo recognition. By replacing the prototypical templates by human faces, the network may be trained to perform face identification. By using the object bounding box from the first frame of a video as the prototypical template, the model may be learned to perform video object tracking. These types of networks that are trained to perform similarity matching in a high-dimensional representation space have been traditionally referred to as 'Siamese nets'. They have been widely used in the past for tasks such as signature verification (Bromley et al., 1994) and fingerprint matching (Baldi & Chauvin, 1993), but were reliant upon hand-crafted feature representations at their input. More recently, hierarchies of convolutional layers are used

to define the representation space for similarity-matching and these layers are trained as part of the end-to-end pipeline for tasks such as character recognition (Koch et al., 2015), tracking (Bertinetto et al., 2016) and handwriting verification (Du et al., 2017). In this thesis, we extend this convolutional 'siamese' architecture to the task of real-world object recognition and demonstrate promising results for seen and unseen class recognition.

Next, **we consider the task of human saliency estimation** (Jetley et al., 2016a). Despite the subjective differences between different human observers, there is a significant consensus in terms of the image regions that they look at in the first few seconds of viewing any image. Thus, for every input image, this consensus is gathered in the form of a 2D map of human-eye fixation scores called a human saliency map. Note that human saliency prediction is a widely relevant problem lying at the heart of a great many challenges at the intersection of cognitive science and applied artificial intelligence. While on one hand, human saliency cues have been widely used for practical ends such as webpage design (Shen & Zhao, 2014), advertisement evaluation (Ma et al., 2009), and thumbnailing (Marchesotti et al., 2009); on the other hand, they have been advanced as complementary cues for predicting object bounding boxes (Feng, 2012; Guo et al., 2017) and object segments (Ye et al., 2017). A more detailed survey of the diverse applications of human saliency information may be found in a recent journal by Nguyen et al. (2018). It is in predicting these human saliency maps that we diverge from the traditional CNN models. **The latter have tended to formulate human saliency estimation as a classification or regression problem. Instead, we view the output saliency maps as probability landscapes where, given a fixed total attention span, humans look at some regions of the input image at the cost of not looking at some others.** We thus propose to train the CNN pipeline using losses that are used to measure the distances between probability distributions. We experiment with a variety of probability based losses including Chi-squared distance and Bhattacharyya distance, and notice a consistent improvement in the quality of match between the predicted and ground-truth saliency maps. Thus, here again, bringing in the contextual intuition allows us to achieve a significant performance boost.

Continuing in the supervised setting, we next tackle the task of object instance segmentation (Jetley et al., 2017). This is an important problem in computer vision,

with applications in areas such as autonomous driving, drone navigation and robotic manipulation. However, most existing methods are not real-time, complicating their deployment in time-sensitive contexts. **We advance a novel real-time instance segmentation approach called 'Straight to Shapes' (STS) which can run at 35 FPS on a high-end desktop by predicting and decoding low-dimensional object shape representations. Additionally, the use of a continuous shape embedding space in this deep prediction pipeline allows our model to map unseen class instances to realistic shape masks at the time of testing.** However, the STS model lacks in terms of the output accuracy in comparison to other offline (non real-time) state-of-the-art methods. In an **extension of this work, we make a series of changes to the network architecture, training routine and loss function formulation resulting in significant accuracy improvements whilst still supporting real-time performance** (Miksys et al., 2019). Consequently, our 'Straight to Shapes++' model achieves a remarkable 19.7 point improvement in mAP over the original method at an IOU of 0.5, as evaluated on the PASCAL VOC dataset. This output performance redefines the accuracy frontier for the task of instance segmentation at real-time speeds.

1.5.2 Examination of CNNs

Towards this, **we first appeal to methods that implement visual attention in networks** in order to identify the visual information that a network finds relevant and that which it finds irrelevant in making its classification decision. Thus, for the first time in the context of a classification network, we introduce an attention module whose weights are adapted as part of the end-to-end network training. The attention module can be introduced atop any layer in a convolutional network and predicts a 2D array of scores which are used to scale the local features at individual 2D spatial locations. The score-weighted addition of the local features is used directly as the input to the final linear classification layer. It is through this step that the scores assume the notion of 'attention' whereby the network has the flexibility to suppress some local features and highlight some others in the interest of the final classification performance. Indeed, this turns out to be the case. **The parameters of the attention module are tuned in tandem with those of the network to generate**

attention maps that are able to highlight the semantic parts of interest in the input images (Jetley et al., 2018b). More interestingly, for the task of fine-grained recognition, the same module when applied to different convolutional layers adapts its focus onto complementary parts of a given object category. Indeed, this is useful when performing a more nuanced discrimination between categories that are very similar (as is the case in the task of fine-grained object recognition). By allowing the network the flexibility to selectively attend to the local image features, we are able to gain a qualitative insight into its visual inference process. Moreover, the use of attention serves to learn features that are better focused on the objects of interest and hence more generalisable. Ultimately, this study is very useful. However, it is also limited in its scope.

Thus, we next extend an existing boundary analysis framework (Fawzi* et al., 2017; Moosavi-Dezfooli* et al., 2018) to study the class decision functions learned by classification CNNs at a geometric level. As a result, **we unravel some properties of their class score functions that tie together the performance capabilities of classification CNNs with their adversarial vulnerability** (Jetley et al., 2018a). In particular, we now have substantial empirical evidence to confirm that the class score functions implemented by deep nets are fairly simplistic in their non-linearity. We identify specific input image space directions such that the scores of the output classes are a fairly monotonic function of the magnitude of the input image component in those directions. Thus, these specific image-space directions are associated with fixed class identities. This provides a new perspective on the existence of universal adversarial perturbations. Further, it is in these directions that the celebrated performance of classification CNNs and their vulnerability to adversarial attack are closely entangled, by the use of simplistic non-linearities for class recognition. This allows us to note that naive compression-based defence strategies remain vulnerable to appropriately designed attacks. Ultimately, **we argue that nets' vulnerability is fundamental and can only be remedied through development of nets with a substantially different concept of class identity than exists currently.**

1.6 Publications and Thesis Outline

This thesis is based on five different but inter-related research projects, each of which has culminated in a first-author publication at a reputable venue.

- **Saumya Jetley, Bernardino Romera-Paredes, Sadeep Jayasumana, Philip Torr. 'Prototypical Priors: From Improving Classification to Zero-Shot Learning.' Proceedings of the 26^{th} British Machine Vision Conference (BMVC), 2015.**

  It forms Chapter 2 of the thesis.

- **Saumya Jetley, Naila Murray, Eleonora Vig. 'End-to-End Saliency Mapping via Probability Distribution Prediction.' Proceedings of the 29^{th} IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.**

  This research project was undertaken during my internship at Xerox Research Centre Europe (XRCE), Grenoble (now known as Naver Labs Europe). I won the 'Best Intern Presentation' award at the XRCE Intern Summit in the fall of 2015 for this work. It was accepted for a spotlight presentation at the conference mentioned above. There is also a U.S. patent (Patent no. $9,830,529$) based on this work (Jetley et al., 2016b). Chapter 3 of the thesis presents the publication.

- **Saumya Jetley*, Michael Sapienza*, Stuart Golodetz, Philip Torr. 'Straight to Shapes: Real-time Detection of Encoded Shapes.' Proceedings of the 30^{th} IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.**

  It forms Chapter 4 of the thesis. This project was later extended through joint work between myself and a master's student from the department of computer science at the university of Oxford. The associated research report is available on arxiv, and is discussed in Chapter 5.

- **Saumya Jetley, Nicholas A. Lord, Namhoon Lee, Philip Torr. 'Learn to pay attention.' Proceedings of the 6^{th} International Conference on Learning Representations (ICLR), 2018.**

This was my first experience with the 'open-review' format of peer-review. The work was well received, it garnered interest from the readers and reviewers alike and was accepted for a poster presentation at the conference. It has been discussed in Chapter 6 of the thesis.

- **Saumya Jetley*, Nicholas A. Lord*, Philip Torr. 'With Friends Like These, Who Needs Adversaries?' Proceedings of the 32^{nd} Conference on Neural Information Processing Systems (NeurIPS), 2018.**

  This forms Chapter 7 of the thesis.

The final chapter, Chapter 8, concludes the thesis by discussing the research impact of the work presented in the earlier chapters and by highlighting some open research directions that follow directly from this line of work.

2

Leveraging Prototypical Priors for Classification and Zero-shot Recognition

Contents

Recent approaches to zero-shot learning make use of side information such as visual attributes or natural language semantics to define the relationship between output visual classes and then use these relations to draw inference on new unseen classes at test time. In a novel extension to this idea, we propose the use of visual prototypical concepts as side information. For most real-world visual object categories, it may be difficult to establish a unique prototype. However, in cases such as traffic signs, brand logos, flags, and even natural language characters, these prototypical templates are available and can be leveraged for an improved recognition performance, zero-shot or otherwise.

The present work proposes a way to incorporate this prototypical information in a deep learning framework. Using class prototypes as prior information, the deep neural network learns to recognise input images by mapping them to the associated prototypes in a deep embedding space. Thus, a deep network can be reinterpreted as a denoising function that maps the diverse samples of a given class to a common description as given by the corresponding prototypical template. Based on our experiments with two different datasets comprising traffic signs and brand logos, this use of prototypes in a conventional convolutional neural network improves the recognition performance. The accuracy on the brand logos dataset (Belga logo dataset) is especially noteworthy and establishes a new state-of-the-art. In zero-shot learning scenarios, the same system can be directly deployed to draw inference on unseen classes by simply adding the prototypical information for these new classes at test time. Furthermore, the system can be tuned for a suitable trade-off between seen and unseen class recognition accuracy as per task requirement. Comparison with one of the latest work in the domain of zero-shot recognition (Norouzi et al., 2013) serves to highlight the benefits of our approach.

2.1 Introduction

Automatic object recognition has witnessed a huge improvement in recent years due to the successful application of convolutional neural networks (CNN) to this task. The boost in performance can be explained by the replacement of heuristic parts in the previous feature representation approaches by a methodology based on learning the features straight from the data (Lee et al., 2009; Krizhevsky et al., 2012a). The learned feature representation, which is tailored to the task at hand, generally outperforms heuristic approaches provided the training data is sufficient. When learned over a significant sample variety, this representation captures regularities across samples of a class that help distinguish it from all the other classes.

In an alternative setup, the object recognition problem can be posed as one in which objects in real images are identified by treating them as imperfect and corrupt copies of prototypical concepts. This assumption provides an additional premise that the different samples of a class are not only similar to each other but also resemble a unique prototype.

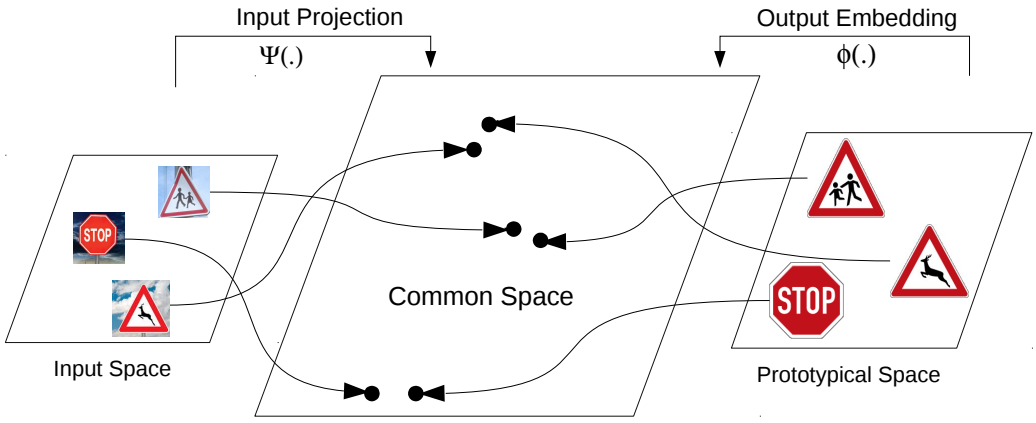


Figure 2.1: A joint embedding space defined by the prototypes.

These prototypical concepts are in many cases not available, for example, there does not exist a chair that contains only the essence of *chair* and nothing else. However, there are many scenarios where such prototypical instances do exist. An example of this is traffic sign recognition, in which each traffic sign class has its canonical template. Real world images contain imperfect instances of it, these imperfections being caused by differing viewpoints, light conditions, damage to surface, occlusion and other factors. These canonical templates, hereafter uniformly referred to as prototypes (*an original or first model of something from which other forms are copied or developed*[1]), can play a very important role in recognition. Conceivably, this prototypical information can benefit by: (a) providing stronger supervision during the learning process and (b) establishing an embedding space wherein the relationship between output visual classes can be used to transfer the learned knowledge to unseen classes directly at test time.

In the present work, we focus on adding this prototypical information into convolutional neural networks. The underlying idea is that the high-level representation learned by a CNN should be comparable to the information extracted from the prototypes. An interpretation of this is that layer-by-layer the CNN is able to learn a representation that is invariant to real world factors such as light variation, view point distortion, occlusion as described in Goodfellow et al. (2009), so that the representation obtained at the end of the network is invariant to the noise appearing in real images, and thus comparable to the prototype.

[1]Definition taken from Merriam-Webster.com dictionary.

Thus, we adjust the traditional CNN pipeline to map both the input and prototypes to a common feature space, wherein class recognition is performed via nearest neighbour matching. This idea of a common space for recognising the instances by matching them to their correct prototypes is shown in Figure 2.1. For the current experiments, this common feature space is defined directly on top of the prototypical images of classes in context using hand-crafted features. The assumption is that these prototypical images, unaffected by noise and distortion, are qualified to define an optimal embedding for maximum discrimination of classes, without the need to perform any complex non-linear mapping.

The proposed use of a joint embedding space also lends the model an interesting possibility of applying it to recognise new classes not present at the training stage, by simply adding the prototypical information of unseen classes at test time. This aligns the approach within the areas of zero and one-shot learning. These areas pursue to emulate the ability of human beings to extrapolate and draw inference on test samples only from a description, or a single instance per class. Indeed, this is a faculty humans own, for example when assimilating and recognising real-world instances of a new character such as €, after being presented with one instance.

In conclusion, this work makes the following contributions: (a) development of a CNN that is able to use prototypical information to guide its learning process, (b) the application of prototypical information to classification tasks yielding a boost in performance, (c) establishment of a new benchmark in logo recognition (on Belga logo dataset), and (d) the seamless application of the proposed model in zero-shot learning scenarios, given the prototypical information of new classes at test time.

2.2 Related Work

Traditional computer vision approaches for classification do not take into account the relationships there may be between the different output classes. Arguably, if these relationships were available as side information, they could be exploited to improve the recognition performance.

Recent work focuses on taking advantage of this side information. A considerable effort has been made towards attribute learning. In this case, side information takes

the form of a high level description of each class in terms of a list of attributes. These attributes are often available in real datasets as tags, and have been popularised within the research community thanks to datasets such as Farhadi et al. (2009); Lampert et al. (2009); Patterson & Hays (2012). Another side information that has recently been exploited by several methods (Socher et al., 2013; Frome et al., 2013; Norouzi et al., 2013) is the semantic vector representation of the name of each class. A semantic space of words can be learned from a large corpus of text in an unsupervised way, so that words are mapped to an Euclidean space in which the distance between vectors depends on the semantic closeness of the words they represent (Mikolov et al., 2013). These vectors, corresponding to the names of the classes, can then be utilised as side information.

The availability of this side information that defines the relationship between output classes has led to the development of the task of zero-shot learning. This is the challenge of identifying a class at test time without ever having seen samples of that class during training, but only using its description in terms of the classes that are indeed seen during training. Over the past few years, this idea has spurred much success, using both attributes (Palatucci et al., 2009; Akata et al., 2013; Romera-Paredes et al., 2015; Lampert et al., 2009), and word embeddings (Norouzi et al., 2013; Socher et al., 2013).

The developed approaches vary in the way knowledge is transferred from the training classes to the new classes. In Lampert et al. (2014); Suzuki et al. (2014) this transfer is done by means of a cascaded probabilistic framework which determines the most likely class. One drawback of probabilistic methods is that they make independence assumptions that do not usually hold in practice. An alternative strategy which bypasses this drawback has been recently exploited in Akata et al. (2013); Romera-Paredes et al. (2015); Weston et al. (2011), where the proposed model learns a linear embedding from both instances and attributes to a common space. This can be seen as a two-layer model that connects the input images to class labels through a layer containing attribute information. The weights connecting the input space to the embedding space are learned to minimise the final classification loss. Our proposed approach builds on this idea, although it presents two significant differences. Firstly, the side information consists of a visual prototype

for each class. Secondly, the mapping function from the input to the embedding space is not linear, but modelled using a deepnet pipeline.

Another related area is that of one-shot learning (Bart & Ullman, 2005; Lake et al., 2011; Fei-Fei et al., 2006). Similar to zero-shot learning, the objective here is to transfer the knowledge learned at training stage to distinguish new classes. The difference is that the information given to the model about the new classes consists in one, or very few, instances. One-shot learning is useful in image retrieval, where given an image as a query, the model returns items that are similar (Sean Bell, 2015). Our work can be considered within this area, with the peculiarity that in our framework the instance provided to the model is a very special one: it is a prototype.

2.3 Proposed Approach

In the usual image classification setup, given training samples of the form (x, y), where $x \in \mathbb{R}^d$ is an image and $y \in \{1, \ldots, C\}$ is the class label of the image, a classifier $h : \mathbb{R}^d \to \{1, \ldots, C\}$ is learned to predict the label of an unseen image x as $\hat{y}$.

If we apply a regular L-layer CNN to this problem, the function that is learned takes the following form:

$$\hat{y} = \underset{c \in \{1,\ldots,C\}}{\operatorname{argmax}} \; s\left(f_L\left(f_{L-1}\left(\ldots f_2\left(f_1\left(x; \theta_1\right); \theta_2\right) \ldots; \theta_{L-1}\right); \theta_L\right)\right)_c . \tag{2.1}$$

Here, f_l, for $l \in \{1, \ldots, L\}$ represents the function (e.g. convolution, pooling) applied at layer l, and θ_l denotes its learnable parameters, if any. The last function f_L maps its inputs to $\mathbb{R}^C$. Finally, $s(.) : \mathbb{R}^C \to [0, 1]^C$ represents the softmax activation function operating on a vector z, as follows:

$$s(z)_c = \frac{\exp(z_c)}{\sum_{j=1}^{C} \exp(z_j)}, \text{ for } c \in \{1, \ldots, C\},$$

where subscripts denote the elements of a vector.

During training, the learnable weights $\theta_1, \theta_2, \ldots, \theta_L$ of the model are adjusted by backpropagating the negative log-likelihood loss with respect to the ground truth class label y of a sample x, defined as follows:

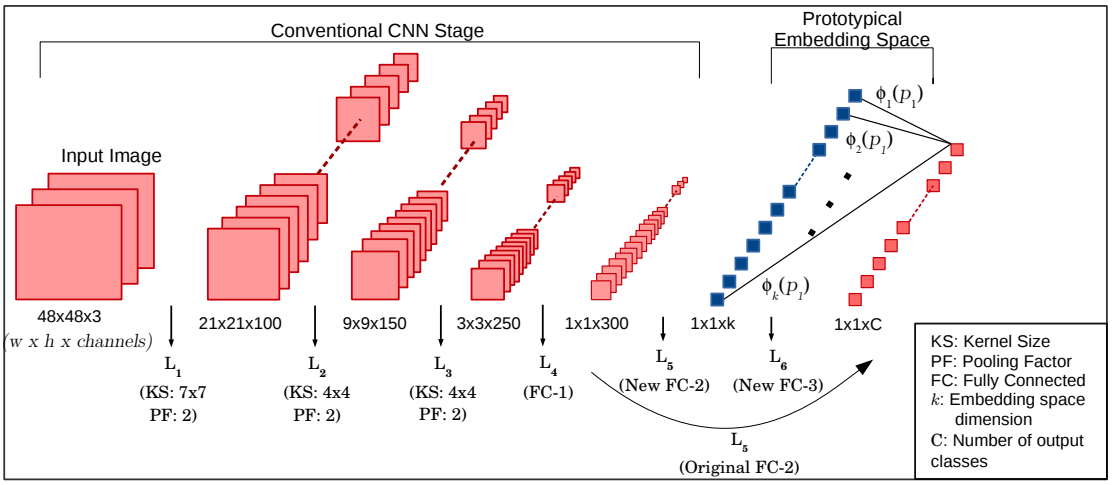


Figure 2.2: Network architecture with the introduction of prototypical priors. In the current experiments, k-dimensional HoG features extracted over the prototypical templates are used to define the common embedding space.

$$\text{loss}(x; \theta_1, \theta_2, \ldots, \theta_L) = -\log(s(z)_y).$$

The CNN represented by Equation (2.1) does not take into account the prior information that class prototypes have to offer. In order to describe our approach, let us assume that a prototype template p_c in the form of a 2D image is provided for each class $c \in \{1, \ldots, C\}$. The proposed approach proceeds by fixing the parameters of the last layer of the CNN as a function of the prototype templates p_c. More precisely, the parameters are given by $\phi(p_c)$, where the function ϕ can be any suitable feature extractor operating on the template p_c. For instance, $\phi(p_c)$ can be a k-dimensional normalised HOG feature vector extracted from a prototypical image p_c, such that the norm $\|\phi(p_c)\|_2$ is held constant for all $c \in \{1, \ldots, C\}$.

We set $f_L : \mathbb{R}^k \rightarrow \mathbb{R}^C s.t. f_L(v)_c = \langle \phi(p_c), v \rangle$, where v denotes the input activations of layer L for a given input image; the subscript denotes vector elements and $\langle ., . \rangle$ denotes the usual dot product in $\mathbb{R}^k$. Since $\|\phi(p_c)\|_2$ is constant, when c is varied for a fixed v, $f_L(v)_c = \langle \phi(p_c), v \rangle$ attains the highest value for the $\phi(p_c)$ closest to v in terms of cosine distance in the k-dimensional feature space. The modified network can now be described using the following formula:

$$\hat{y} = \underset{c \in \{1,\ldots,C\}}{\text{argmax}} \; s\left(f_L\left(f_{L-1}\left(\ldots f_2\left(f_1\left(x\right)\right)\ldots\right)\right)\right)_c = \underset{c \in \{1,\ldots,C\}}{\text{argmax}} \langle \phi(p_c), \psi(x) \rangle, \qquad (2.2)$$

where $\psi(.)$ and $\phi(.)$ represent the projections of the input images and the output labels into the joint feature space defined by the class prototypes. An interpretation of this approach is that the learnable part of the network, $\psi : \mathbb{R}^d \rightarrow \mathbb{R}^k : \psi = f_{L-1} \circ \ldots \circ f_1$, learns a denoising function that maps input images to the latent embedding space of class prototypes such that their cosine distance with the correct prototype is minimised while maximising the distance with all the incorrect prototypes. The mapping from the input image space to the k-dimensional latent space is allowably a complex non-linear mapping. And, the space itself contains both the projections of an input image $\psi(x)$ and the prototype templates $\phi(p_c)$ for all $c \in \{1, \ldots, C\}$, such that the similarity between each instance-prototype pair can be computed by means of an inner product. The softmax based loss function helps to encourage this inner product (i.e. cosine distance) between $\psi(x)$ and $\phi(p_c)$ to be high if x belongs to class c, and to be low otherwise. Thus, we introduce prior information about the classes directly into the network pipeline with the belief that the intra-class noise affecting the real-world samples is shared across classes and that the prototypical representations offer a stronger supervision in learning invariances to these noise factors, while offering maximum discriminability between classes based on true properties that are embodied in the prototypes.

Note that, unlike in other works such as Palatucci et al. (2009); Norouzi et al. (2013), the inference process at test time is exactly the same as in any other CNN. This is by virtue of the fact that the distance calculations for the embedding space are integrated into and optimised as part of the CNN pipeline. There is no need to perform explicit calculations of embedding space distances in a post hoc manner.

This framework easily allows for using new prototypes after the training stage is finished. This is done by replacing, or adding to the last layer new weights according to the new prototypes. The resultant network is potentially capable of distinguishing the new classes because the invariances learned in ψ are conceivably common to all classes.

In the given framework, both functions ψ and ϕ can be learned. However, for the purpose of current research, we focus on the case where ψ is learned as part of the traditional CNN pipeline, while ϕ is a pre-established function, such as HoG transform.

2.4 Details of Implementation

We now describe the architecture of the deep neural network that is used to implement the ideas discussed above. The first stage of our network consists of a CNN to enable the learning of high-level features for input RGB patches of size 48×48 (suitable for both traffic-sign and logo samples in the experimental datasets).

The configuration, as presented in *red* (*light* in a greyscale rendering) in Figure 2.2, is the same as proposed in Cireşan et al. (2012), with the difference that our version does not have a dropout layer after layer L_5, unlike in theirs. As in a traditional CNN designed for classification, the last few layers are fully-connected, and the network is terminated with a layer having the same number of activations as the number of classes C. A softmax function is applied to the last layer to map the output class scores to a probability distribution, which can then be matched against the one-hot encoded ground truth distribution using cross-entropy loss. The network parameters are adjusted through error backpropagation so as to minimise the output loss function.

In the proposed approach, prototypical information is introduced by adding a layer before the softmax layer, fully connected to the C output neurons using $\phi(p_c) \in \mathbb{R}^k \; \forall \; c \in \{1, \ldots, C\}$ as fixed weights. The new layer and its connections are shown in *blue* (*dark* in a greyscale rendering) in Figure 2.2. Thus, the $k \times C$ weight matrix for the last fully connected layer f_L is defined as a set of C k-dimensional vectors, each given by $\phi(p_c)$ where $c \in \{1, \ldots, C\}$. In Figure 2.2, we use $\phi_1(p_c), \phi_2(p_c), \ldots, \phi_k(p_c)$ (note the subscript indices) to denote the elements of a given k-dimensional vector $\phi(p_c)$.

In the current work, the feature vectors given by $\phi(p_c) \; \forall \; c \in \{1, \ldots, C\}$ correspond to the k-dimensional normalised histograms of oriented gradients (HoG features) (Dalal & Triggs, 2005) extracted from the underlying prototypical templates p_c. The prototypical images are all first resized to a fixed size $s \times s$. The HOG features are extracted using the standard *extractHoGFeatures* function provided with Matlab. In our experiments, we make use of an empirically selected cell size $c = 10$, block size $b = 2$, overlap factor $o = 1$ and a bin count $n = 12$, for size $s = 100$, which yields 3888-dimensional HoG features.

Figure 2.3: Sample images of traffic signs (left) with view point distortion, illumination variation and background clutter while logo images (right) additionally contain non-planar distortions and high self-occlusion.

2.5 Experiments

We study the effect of incorporating prototypical information in a deep neural network pipeline for classification, by observing the change in: (a) the overall classification performance when all classes are seen during training, (b) the classification performance for unseen classes, i.e., in a zero-shot recognition setup.

2.5.1 Datasets

We evaluate the proposed approach on two distinct datasets as described below.

Traffic Sign Dataset: We use the German Traffic Sign Recognition benchmark (Stallkamp et al., 2012), hereafter referred to as GTS. This dataset has a substantial sample base of more than 50,000 images distributed over 43 traffic sign classes. It is divided into 39,209 training samples and 12,630 test samples. For the purpose of our experiments, we randomly split the test data into validation and test sets of 6,315 samples each. We crop the images such that they completely overlap with the tight-fitting bounding boxes around the inlying traffic signs. The bounding box information is provided with the dataset. No additional distortion (such as scaling, rotation) is applied to augment the data at train or test time.

Brand Logo Dataset: We use the Belga Logos dataset (Joly & Buisson, 2009), hereafter referred to as BL. This dataset contains bounding-box based sample images for 37 logo categories collected from across 10,000 real images. Out of a total of 9,841

Dropout Factor	w/o Protos Test accuracy (%)	with Protos Test accuracy in (%)
0.5	96.60	**97.98**
0.6	97.18	**97.53**
0.65	97.48	**97.74**

Table 2.1: Despite aggressive dropout, the addition of prototypical information in the inference pipeline (with Protos) effects a consistent improvement in the output classification accuracy on the GTS dataset. Note: The traditional accuracy (without Protos) compares to that in Cireşan et al. (2012) for the case when no data augmentation is used during training.

Dataset	w/o Protos Test accuracy (%)	with Protos Test accuracy in (%)
GTS	97.48	**97.98**
BL	93.48	**93.57**

Table 2.2: The effect on the classification performance on the two datasets (traffic sign and logo datasets respectively) when their corresponding prototypical information is utilised.

logo samples, only 2,697 are marked as 'OK' for their ability to be recognisable without the image context. Thus, we use a subset of 10 logo classes (out of the 37), for which the total number of 'OK' samples per class is at least 100. We set aside 20% of those samples from each class for validation, and 20% for testing. Sample images from both the datasets are as shown in Figure 2.3.

2.5.2 Results and Discussion

Overall Recognition Performance

In this setup, all the classes are treated as seen. Classification results on the GTS dataset for the conventional (without prototypical priors) and the proposed (with prototypical priors) deepnet architectures are shown in Table 2.1. Note that the two architectures are comparable within the constraints of the modifications i.e. modulo the fact that the proposed architecture needs additional parameters in order to regress to points in the prototypical embedding space. To begin with, we experiment with three distinct choices of the dropout factor after layer L_5 in both the architectures. This is to study the extent to which similarity based matching in the prototypical space is affected or replicated by some form of regularisation. We observe that while the conventional architecture benefits by the inclusion of dropout, the proposed architecture needs all the expressivity it can get in

order to predict points in the latent space, as evinced by the marginal loss of performance with dropout based regularisation. Notably, even as we increase the dropout factor, the classification performance gap between the two architectures does not entirely close.

Top results on GTS and BL datasets, without prototypical priors and with the prototypical priors, are shown in Table 2.2. For the GTS dataset, the test performance without prototypical information is comparable to that cited in (Cireşan et al., 2012) in the case when no additional data augmentation technique is used. Thus, the inclusion of prototypical information appears to be able to supplant the need for data augmentation and effects a 0.5% improvement in the performance accuracy ($\sim$ to a 20% reduction in the error rate) over the baseline architecture. On the BL dataset, the proposed approach yields a performance comparable to the baseline. The logo samples exhibit heavy self occlusion, perspective distortion and general lack of visual quality. Hence, the task of matching the real-world samples to the prototypical templates is more challenging in the case of this particular dataset.

Additional Findings: For both the datasets, we experimented with greyscale as well as coloured (RGB) prototypes. Models using prototypical features extracted from the coloured templates consistently perform lower (by an average margin of 0.1%) in comparison to those using features obtained from the greyscale templates.

This suggests that while color coding may be useful for attracting human visual attention, it is not essential for distinguishing the classes. Indeed, for the traffic sign dataset (GTS) 12 out of 43 classes are *Prohibitory* traffic signs and have a common circular red and white color coding scheme, while 8 are *Mandatory* signs and have a uniform circular blue and white color coding. The main distinction in the traffic signs comes from the shape of the inset object/pattern. On the other hand, for the logo dataset (BL), samples show significant color variations within a single class, as shown in Figure 2.3, which may render the color information itself quite irrelevant.

Zero-Shot Recognition Performance

Data Setup: For the purpose of this experiment, we divide the 43 classes of GTS dataset into 33 seen classes ($\boldsymbol{c_s} = \{1, \ldots, C_s\}$), and 10 unseen classes ($\boldsymbol{c_u} = \{C_s + 1, \ldots, C - C_s\}$).

Samples from the classes in the set $\boldsymbol{c_s}$ are used for training the model while the remaining 10 classes in $\boldsymbol{c_u}$ are used for testing the model. During test time, only $c \in \boldsymbol{c_u}$ form the output label set, that is, the network can only predict a label from set $\boldsymbol{c_u}$. Similarly for the BL dataset, 10 classes are divided into 7 seen classes (set $\boldsymbol{c_s}$) and 3 unseen classes (set $\boldsymbol{c_u}$). Samples with the class labels in $\boldsymbol{c_s}$ are used for training the model and the 3 classes in $\boldsymbol{c_u}$ are used for testing.

We perform 10 random trials with different allocations of the classes into $\boldsymbol{c_s}$ and $\boldsymbol{c_u}$ respectively. For the proposed approach, the prototypical representations of $\phi(p_c)$ for $c \in \boldsymbol{c_s}$ are used during training. These are replaced by $\phi(p_c)$ for $c \in \boldsymbol{c_u}$ during testing.

Note: A keen reader may note a practical issue with the zero-shot recognition setup described above. This is to do with the test-time restriction that the network must only cater to the unseen classes during zero-shot inference. Clearly, in the wild, the samples of seen and unseen classes would be mixed and the overall system should first be able to successfully identify whether an incoming sample is of a seen class or an unseen class before making use of the above discussed zero-shot recognition pipeline in the latter case. Since this practical concern is an aside to the current analysis, but nonetheless a crucial aside, we take this up in a greater detail in the Appendix 2.A. We propose a modification to the deepnet classification pipeline to enable it to identify when a given sample does not belong to any of the seen classes. In that case, the sample can then be processed by the zero-shot recognition network along with the prototypical information of the unseen classes to predict which of the unseen classes it belongs to.

Comparison: We compare our approach with the method of convex combination of embedding vectors, as discussed in Norouzi et al. (2013). In their work, the samples of new unseen classes are represented as weighted combinations of the vector embeddings $\phi(p_c)$ of the seen classes $c \in \boldsymbol{c_s}$, where the normalised scores of the softmax layer serve as the weights for the convex combination. Thus, they propose to combine the representations of the top T best scoring classes to yield the feature representation of an unseen-class sample, where T is a hyperparameter that can be tuned by means of a

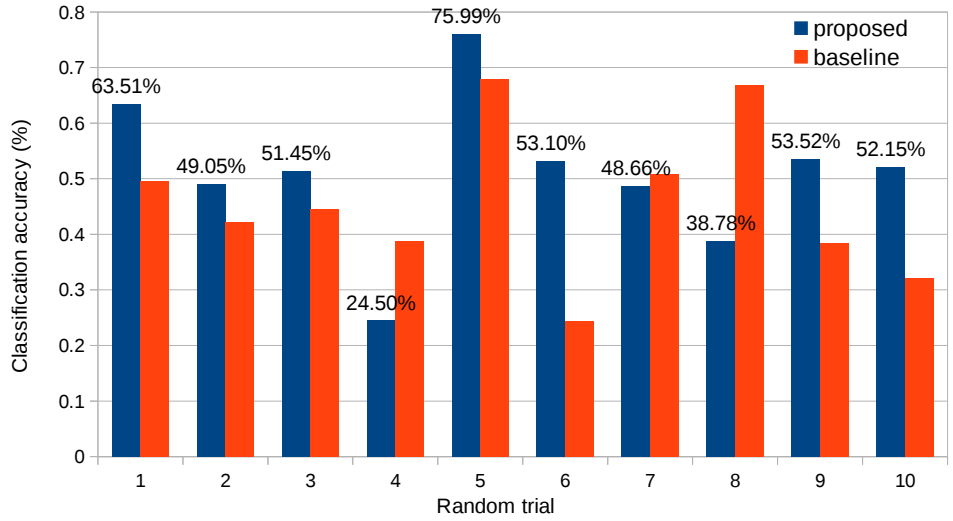


Figure 2.4: Performance comparison of the proposed and baseline (Norouzi et al., 2013) approaches over 10 random trials at the task of recognition of the unseen-class samples of the GTS dataset.

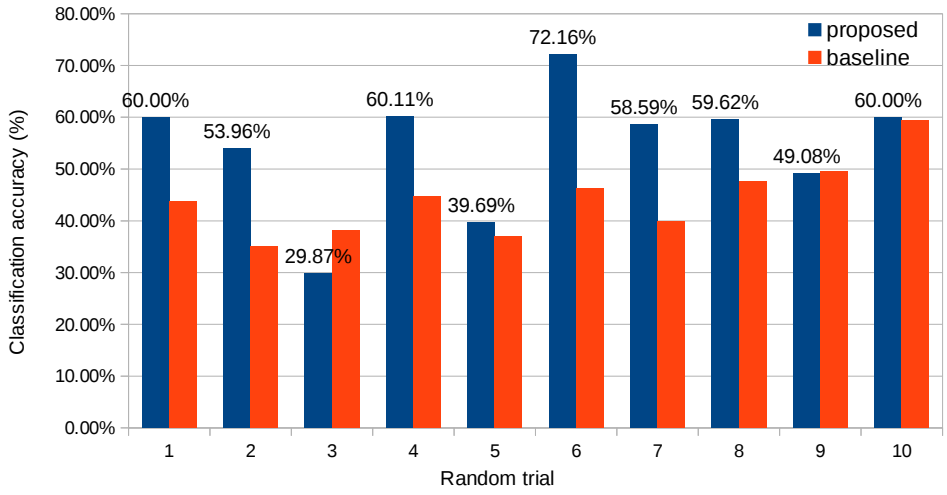


Figure 2.5: Performance comparison of the proposed and baseline (Norouzi et al., 2013) approaches over 10 random trials at the task of recognition of the unseen-class samples of the BL dataset.

validation process. These representations are then compared in the vector space defined by $\phi(p_c)$, where $c \in \boldsymbol{c_u}$. The class of an input sample is inferred to be the class of the nearest prototype in this space. In our experiments, the validation hyperparameter T of Norouzi et al. (2013) is set to the total number of seen classes C_s. *Note that their method assumes that the representation of an unseen class sample must lie in the convex hull of the seen-class representations.*

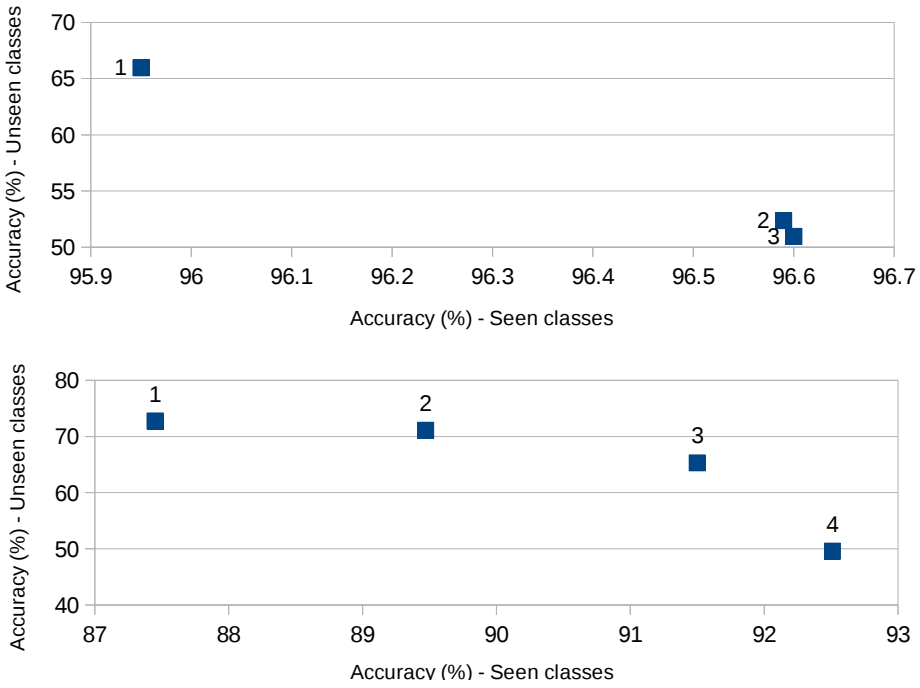


Figure 2.6: Seen and unseen class recognition performance trade-off as evaluated at different stages during the training of the network on GTS and BL datasets respectively. (Lower number corresponds to an earlier model-state during the training process.)

Findings: The classification results for the unseen classes of the GTS and BL datasets are compared in Figures 2.4 and 2.5 respectively. The proposed approach outperforms the baseline approach of Norouzi et al. (2013) by an average of 5.48% and 10.15% accuracy points on GTS and BL datasets respectively. The null hypothesis that the two approaches are statistically equivalent in terms of their performance has a *p-value* of 5% for the GTS dataset and 33% for the BL dataset, thus demonstrating that their performance gap is indeed statistically significant. Due to high visual similarity between different traffic signs, an unseen traffic sign sample can still be fairly well reproduced by the combination of related seen-class prototypical templates as assumed in Norouzi et al. (2013). The major benefit of our approach becomes evident in the case of the BL dataset where the different logo categories are more visually dissimilar, and the resulting zero-shot recognition performance is considerably improved in comparison to the baseline approach of Norouzi et al. (2013).

In the approach of Norouzi et al. (2013), the network can be trained for an optimal performance only with respect to the seen class set $\boldsymbol{c_s}$. At validation time the parameter

T, that defines the number of seen classes used for inference, provides little flexibility for tuning the performance on $\boldsymbol{c_u}$. On the contrary, our model can be more closely adapted to a desired trade-off between seen and unseen class recognition performance. To analyse this, we train the proposed network on the samples of the seen classes and as the training progresses we evaluate the performance of the intermediate models at the task of recognition of samples of the unseen classes. Our experimental observations reveal that during the initial phase of the training the accuracy on seen and unseen class samples improves steadily and jointly. However, past a point the two accuracies are negatively correlated, as shown in Figure 2.6. The above tests are conducted using a randomly selected set of 5 unseen classes for the GTS dataset and 2 unseen classes for the BL dataset. Here, we plot the network accuracies for a certain sampling of intermediate models to depict the overall trade-off trend. Traditionally, we use samples from seen classes $\boldsymbol{c_s}$ during training and validation and hence it comes as no surprise that over the long run the training favours seen-class recognition performance. Early termination would be one way of achieving a desired trade-off between seen and unseen class recognition performance. Another scheme could be to use the samples of some randomly selected unseen classes for fine-tuning the model during the validation stage.

2.6 Conclusion

In this work we have shown that visual prototypes can be successfully used as side information to aid the learning process in traditional deep convolutional networks for classification as well as for zero-shot recognition tasks.

We propose a method for integrating prototypical information in the deep learning framework. Using a conventional CNN pipeline, we demonstrate a learnable mapping from the space of input images to a prototypical space such that the maximisation of similarity between a real sample and its prototypical representation corresponds to the minimisation of the empirical error at the task of classification. In the current research, the prototypical embedding space and its mapping to the space of output classes is fixed by the choice of the prototypical representation, while the input mapping to this space

is learnable as a complex non-linear function. More generally, however, both the input and output mappings can be learned as part of an end-to-end deepnet pipeline.

As observed on two different datasets of traffic signs and brand logos, results of the proposed approach are highly promising. Regarding its application to regular object recognition, we can conclude that constraining the network to incorporate the associated prototypical information does not hamper, but often-times improves the classification performance. With regard to zero-shot recognition, our model shows better results than a state-of-the-art competitor (Norouzi et al., 2013). Furthermore, our model can be flexibly trained for the required trade-off between seen and unseen class recognition performance, and inference on new unseen classes simply involves adding their prototypical information to the deep network pipeline at test time.

2.A Appendix

2.A.1 A Practical System for Zero-shot Recognition

Recent vision systems for object recognition are learned from large amounts of visual data and, by default, can only successfully recognise the objects of the categories they are trained for. In order to extend the inference to new unseen classes that are represented by a handful of samples, recent work in the research community has investigated the use of side information. This side information, by virtue of establishing a relationship between seen and unseen classes, is able to extend the image-to-class learned mapping from the former to the latter. In the current work, we propose and demonstrate the use of class prototypical templates as the side information.

Towards this end, we instantiate a prototypical embedding space in the convolutional network pipeline for object classification. During training, the weights of the network are adapted so as to minimise the distance between the input images and their associated visual prototypes in this embedding space. At test time, we proceed by performing similarity matching in this embedding space i.e. the prototype that a given image is closest to establishes the class label of the input image.

For inference of unseen classes, we only need to place the unseen class visual prototypes in this embedding space. To the extent that the denoising function learned by the network generalises across classes, i.e. it is able to match noisy real-world samples to their untampered prototypes, the input image of an unseen class should exhibit maximum similarity with the correct unseen class prototype. However, there is one caveat to this. The trained network seems to implicitly capture the frequency of occurrence of class samples. In other words, since the network has seen large number of samples of the seen classes $c \in \boldsymbol{c_s}$, it is more likely to identify an incoming sample as belonging to a seen class in comparison to an unseen class $c \in \boldsymbol{c_u}$.

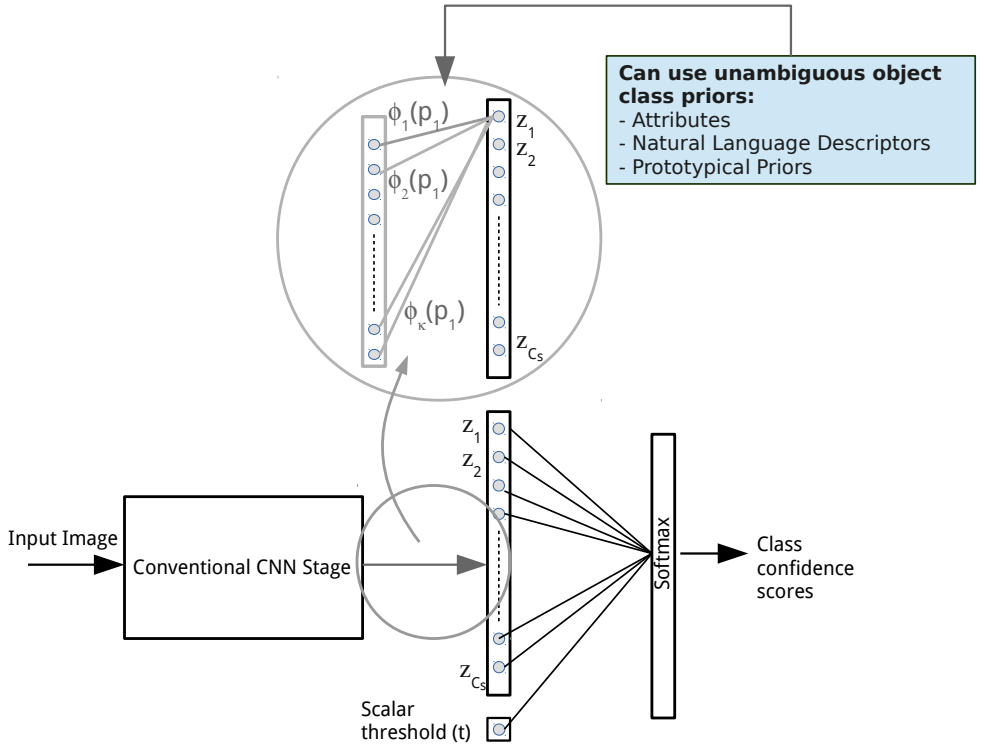


Figure 2.7: Network architecture with threshold for identifying unseen class samples.

To overcome this bias, we separate the inference process for seen and unseen classes. More specifically, in order to disentangle the seen classes, we make use of an embedding space anchored only in the seen class prototypes. Likewise, to disentangle the unseen classes, we make use of an embedding space anchored only in the unseen class prototypes. It soon becomes clear that in order to do this we need an overarching mechanism to

identify whether a sample belongs to a seen class or an unseen class before we can identify its class label by matching for similarity in the appropriate prototypical embedding space.

For this purpose, we propose a new classification architecture that incorporates the concept of an *unseen class* by defining a *threshold*. This network is trained end-to-end together with the scalar threshold such that if the output score of an input sample for all classes is less than the threshold, then the sample belongs to an unseen class, else it belongs to the class with the maximum score, as follows:

$$\text{class}(x) = \begin{cases} \text{unseen}, & \text{if } z_c < t \; \forall \; c \in \{1, 2, \cdots C_s\} \\ \text{argmax}(\text{z}_\text{c}), & \text{otherwise,} \end{cases} \tag{2.3}$$

where x represents the input sample, z_c is the class score before the final softmax layer and C_s is the total number of seen classes. The network implementation is as shown in Figure 2.7. More intuitively, this threshold implements a Gaussian kernel based similarity matching in the seen-class space to identify the class label. If the incoming sample does not match strongly enough with any of the seen class prototypes, then it must belong to an unseen class and can be processed accordingly in the unseen-class space. The scalar threshold itself can be fixed a priori or can be learned during network training. We compare the results for these two different setups on the GTS dataset in Table 2.3. For the purpose of this experiment we augment the GTS dataset by adding samples that do not belong to any seen class. These samples contain image crops that have no overlap with the bounding boxes of traffic sign objects. We collect ≈ 2500 such unseen class samples, ≈ 1500 of them are allocated to the training set, and ≈ 500 are added each to the validation and test set.

Description	Test Acc.(%)
Background category	95.77
Fixed-t	95.19
Learned-t	95.68

Table 2.3: Recognition accuracy for the GTS dataset with the use of fixed and learned thresholds for unseen class identification.

As can be observed, our model that uses a threshold to identify unseen class samples is comparable in performance to approaches like those of Ren et al. (2015); Long et al.

(2015) which treat *unseen classes* as yet another object category, often-times referred to as the *background* category. Both types of models support the identification of samples that do not belong to classes seen during training. However, the use of side information is something that the existing approaches do no easily support. This is because by treating the gamut of unseen classes as yet another object category (*background*), one is forced to define a feature vector for the object category in terms of the side information. However, the *background* category is an ever-changing concept. Its description is dependent on the categories being treated as seen to begin with. Thus, intuitively and practically, there can be no single definition in terms of the side information for this category. Our use of threshold to identify and exclude samples of unseen classes from further processing safely disposes the need for such a definition.

3

End-to-End Saliency Mapping via Probability Distribution Prediction

Contents

Most saliency estimation methods aim to explicitly model low-level conspicuity cues such as edges or blobs and may additionally incorporate top-down cues using face or text detection. On the other hand, data-driven methods for training saliency models using eye-fixation data are becoming increasingly popular, particularly with the introduction of large-scale datasets and deep architectures. However, current methods in this latter paradigm use loss functions designed for classification or regression tasks whereas saliency estimation is evaluated on topographical maps. In this work, we introduce a new saliency estimation model which formulates a map as a generalised Bernoulli distribution.

We then train a deep architecture to predict such maps using novel loss functions which pair the softmax activation function with measures designed to compute distances between probability distributions. Via extensive experiments we show the effectiveness of such loss functions over standard ones on four public benchmark datasets, and demonstrate improved performance over state-of-the-art saliency methods.

3.1 Introduction

This work is concerned with visual attention prediction, specifically with the task of predicting a topographical visual saliency map when given an input image. Visual attention has been traditionally used in computer vision as a pre-processing step in order to focus subsequent processing on regions of interest in images, an ever more important step as vision models and datasets increase in size. Saliency map prediction has found useful applications in tasks such as automatic image cropping (Stentiford, 2007), content aware image resizing (Achanta & Süsstrunk, 2009), image thumb-nailing (Marchesotti et al., 2009), object recognition (Gilani et al., 2015), and fine-grained scene and human action classification (Sharma et al., 2012). Traditional saliency models, such as the seminal work of Itti et al. (1998), have focused on designing hierarchical mechanisms that attempt to explicitly model biological systems. Another popular attention modelling paradigm involves the use of data-driven approaches to learn patch-level classifiers which give every local image patch a 'saliency score' (Kienzle et al., 2007; Judd et al., 2009), using eye-fixation data to derive training labels. A recent trend has emerged which intersects with both of these paradigms: the use of hierarchical constructs to extract saliency maps, through models whose parameters are learned from data in a supervised manner. In particular, end-to-end or 'deep' architectures, which have been successfully used in semantic labelling tasks such as categorisation or object localisation, have been re-purposed as attention models (Kümmerer et al., 2015; Pan & i Nieto, 2015). This trend has been facilitated by the introduction of large visual attention datasets created using novel eye movement collection paradigms (Jiang et al., 2015; Xu et al., 2015b). However, while these deep methods have focused on designing appropriate architectures

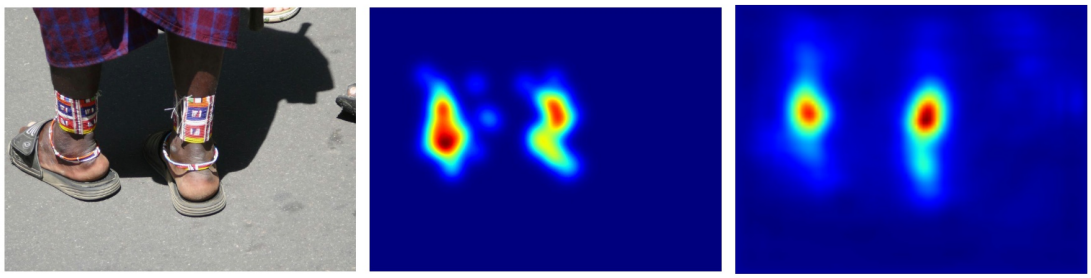


Figure 3.1: Sample image (left) with ground-truth saliency map (middle) and map predicted by our PDP approach (right).

for extracting saliency maps, they continue to use loss functions adapted for labelling tasks, such as classification or regression losses.

In this work, we propose a novel formulation of the task of saliency map prediction as one of predicting a *spatial probability distribution*. The map is formulated as a generalised Bernoulli distribution, and several novel loss functions are proposed based on probability distance measures. We show that training a deep architecture with such loss functions results in a superior performance in comparison to the standard regression based loss functions such as the Euclidean and Huber losses. We also perform a comparison between our proposed loss functions and show that the loss function based on the Bhattacharyya distance for multinomial distributions gives top performance.

Our contributions are therefore the following - (i) a novel formulation which represents a saliency map as a generalised Bernoulli distribution, (ii) a set of novel loss functions which are paired with the softmax function and which penalise the distance between predicted and target distributions, and (iii) a fully-convolutional architecture which can generate a saliency map for a large image in $\sim$ 200ms using modern GPUs. Our extensive experimental validation on four datasets demonstrates the effectiveness of our approach when compared to other loss functions and other state-of-the-art approaches to saliency map generation. Figure 3.1 illustrates the prediction performance of the proposed approach.

The remainder of the chapter is organised as follows: in section 3.2 we discuss the related work, section 3.3 describes our saliency modelling approach, we report and discuss the results in section 3.4 and conclude in section 3.5.

3.2 Related Work

Existing approaches can be organised into one of four broad categories based on whether they involve a *shallow* architecture or a *deep* architecture and an *unsupervised* or *supervised* learning paradigm. We will discuss each of these categories in turn. For an excellent survey of saliency estimation methods, please refer to Borji & Itti (2013).

Shallow Unsupervised Methods: Most early work on saliency builds on psychological and psychophysical models of attention as studied in humans. Koch & Ullman (1987) were amongst the first to use 'feature integration theory' (Treisman & Gelade, 1980) to propose a set of individual topographical maps of elementary cues such as color, contrast, and motion, and combine them to produce a global topographical map of saliency. Their model is implemented using a simple neural circuitry with winner-take-all and inhibition-of-return mechanisms. This method is further investigated in Itti & Koch (2000) by combining features maps over a wider set of modalities (42 such maps) and testing on real-world images. Later approaches largely explore the same idea of complementary feature ensembles (Itti, Koch, and Niebur, 1998; Valenti, Sebe, and Gevers, 2009; Liu, Le Meur, Luo, and Shen, 2013; Lang, Nguyen, Katti, Yadati, Kankanhalli, and Yan, 2012; Zhang and Sclaroff, 2013; Murray, Vanrell, Otazu, and Parraga, 2013) and often add to it additional center-surround cues (Itti, Koch, and Niebur, 1998; Murray, Vanrell, Otazu, and Parraga, 2011; Zhang and Sclaroff, 2013).

Complementing the biologically motivated approaches, a number of methods adopt an information-theoretic justification for attentional selection, *e.g.* by self-information (Zhang et al., 2008), information maximisation (Bruce & Tsotsos, 2006), or Bayesian surprise (Itti & Baldi, 2006). High computational efficiency is achieved by spectrum-based methods (Hou & Zhang, 2007; Schauerte & Stiefelhagen, 2012). All these approaches use bottom-up cues, are shallow (one or few layers) and involve no or minimalistic learning of thresholds/heuristics.

Shallow Supervised Methods: This category includes learning based approaches involving models such as markov chains (Harel et al., 2006), support vector machines (Kienzle et al., 2007; Judd et al., 2009) and adaboost classifiers (Cerf et al., 2008). The approach

of Harel et al. (2006) substitutes the idea of centre-surroundedness and normalisation with learnable graph weights. Cerf et al. (2008), Judd et al. (2009), and Zhao & Koch (2011) enrich learning by incorporating top-down semantic cues in the form of detection maps for faces, persons, cars, and the horizon.

Hierarchical Unsupervised Methods: In the context of saliency prediction, the first attempts to employ deeper architectures are mostly unsupervised. Shen et al. (2012) learn higher-level concepts from fixated image patches using a 3-layer network of sparsely coded units. Vig et al. (2014) perform a large-scale search for optimal network architectures of up to three layers, but the network weights are not learned.

DeepGaze (Kümmerer et al., 2015) employs an existing network architecture, the 5-layer AlexNet (Krizhevsky et al., 2012a) trained for object classification on ImageNet, to demonstrate that off-the-shelf CNN features can significantly outperform non-deep and 'shallower' models at saliency estimation, even if they are not explicitly trained on this task. Learning, in their case, has meant finding the optimal linear combination of features from the different network layers and using them to predict the actual visual fixations.

Hierarchical Supervised Methods: The publication of large-scale attention datasets such as SALICON (Jiang et al., 2015) and TurkerGaze/iSUN (Xu et al., 2015b) has enabled training deep architectures specifically for the task of saliency prediction. Our work lies in this category and involves training an end-to-end deep model with a novel loss function.

SALICON (Jiang et al., 2015) is collected using a new data-collection paradigm, in which observers are shown blurred images and are asked to move the mouse cursor around to explore the image, where the cursor position corresponds to the high-resolution foveation of the visual stimuli. This novel paradigm is then used to annotate 20K images from the MS-COCO dataset (Lin et al., 2014a). Relying on this new large-scale dataset, Pan & i Nieto (2015) trained a network end-to-end for saliency prediction. Their network, called *JuntingNet*, consists of five convolutional and two fully-connected layers, and the parameters of the network are learned to minimise the Euclidean loss between the predicted and the ground-truth saliency maps. Their method reports state-of-the-art results on the LSUN 2015 saliency prediction challenge (Zhang et al., 2015).

Another end-to-end approach that formulates saliency prediction as a regression task is that of Kruthiventi et al. (2015). Their method called 'DeepFix' builds upon the very-deep VGGNet (Simonyan & Zisserman, 2015), uses convolutional layers with large and multi-size receptive fields to capture complementary image contexts, and introduces a location-biased convolutional (LBC) layer to model the center-bias.

Finally, one of the most recent works in this paradigm (Huang et al., 2015) proposes to bridge the semantic gap with conventional saliency literature via a two-pronged strategy. The first is the use of KL-divergence as a loss function, motivated by the fact that it is a standard metric for evaluation of saliency maps. The second is the aggregation of response maps across both coarse and fine resolutions. In this work, we develop this perspective further and argue for a well-motivated probabilistic modelling of the saliency maps. We study the use of KL-divergence, among other probability distance measures, as a loss function. As we discuss in section 3.4, the proposed Bhattacharyya distance based loss function consistently trains the network to outperform the KL-divergence based loss function across 4 standard saliency metrics.

3.3 Saliency Maps as Probability Distributions

Probability distances	$L(\boldsymbol{p}, \boldsymbol{g})$	$-\frac{\partial L(\boldsymbol{p},\boldsymbol{g})}{\partial x_i^p}$
χ^2 divergence	$\sum_j \frac{(g_j)^2}{p_j} - 1$	$p_i \sum_{j \neq i} \frac{g_j^2}{p_j} - \frac{g_i^2}{p_i}(1 - p_i)$
Total Variation distance	$\frac{1}{2} \sum_j \lvert g_j - p_j \rvert$	$\frac{1}{2} \left[p_i \sum_{j \neq i} \frac{g_j - p_j}{\lvert g_j - p_j \rvert} p_j - p_i \frac{g_i - p_i}{\lvert g_i - p_i \rvert}(1 - p_i) \right]$
Cosine distance	$1 - \frac{\sum_j p_j g_j}{\sqrt{\sum_j p_j^2} \sqrt{\sum_j g_j^2}}$	$\frac{1}{C} \left[p_i \sum_{j \neq i} p_j (g_j - p_i \frac{\sqrt{\sum_i g_i^2}}{\sqrt{\sum_i p_i^2}} R) - p_i (g_i - p_i R)(1 - p_i) \right]$ where $R = \frac{\sum_i p_i g_i}{C}$ and $C = \sqrt{\sum_i p_i^2} \sqrt{\sum_i g_i^2}$.
Bhattacharyya distance	$-\ln \sum_j (p_j g_j)^{0.5}$	$\frac{-1}{2 \sum_j (p_j g_j)^{0.5}} \left[p_i \sum_{j \neq i} (p_j g_j)^{0.5} - (p_i g_i)^{0.5} (1 - p_i) \right]$
KL divergence	$\sum_j g_j \log \frac{g_j}{p_j}$	$p_i \sum_{j \neq i} g_j - g_i (1 - p_i)$

Table 3.1: Probability distance measures and their derivatives w.r.t the predicted pixel value x_i^p for purposes of back-propagation in stochastic gradient descent based network training. We propose the use of the first 4 measures as loss functions. We also investigate KL-divergence, which is widely used to train recognition models in the form of the closely related cross-entropy loss.

Saliency estimation methods have typically sought to model local saliency based on conspicuity cues such as local edges or blob-like structures, or on the scores of binary

saliency classifiers trained on fixated and non-fixated image patches. More recently, methods have sought to directly predict maps using pixel-wise regression.

However, visual attention is fundamentally a stochastic process due to it being a perceptual and therefore subjective phenomenon. In an analysis of 300 images viewed by 39 observers, Judd et al. (2012) find that the fixations for a set of n observers match those from a different set of n observers with an AUC score that increases with an increase in the value of n. The lower bound on this AUC score is found to be 85%, signifying a high consistency across observers. At the limit of $n \to \infty$ this AUC score is 92%, which can therefore be considered a realistic upper-bound for machine based saliency estimation performance.

Ground-truth saliency maps are constructed by aggregating the fixations of multiple observers, ignoring any temporal fixation information. Areas with a high fixation density are interpreted as receiving more attention. As attention is thought to be given to a localised region rather than an exact pixel, two-dimensional Gaussian filtering is typically applied to a binary fixation map to construct a smooth 'attentional landscape' (Zangemeister et al., 1996) (see Figure 3.1, middle image for an example). Our goal is to predict this attention landscape, or saliency map. Given the stochastic nature of the fixations upon which the maps are based, and the fact that the maps are based on aggregated fixations without temporal information, we propose to model a saliency map as a probability distribution over pixels, where each value corresponds to the probability of that pixel being fixated upon. That is, we represent a saliency map as a generalised Bernoulli distribution $\boldsymbol{p} = (p_1, \cdots, p_i, \cdots, p_N)$, where $\boldsymbol{p}$ is the probability distribution over a set of pixels forming an image, p_i is the probability of pixel i being fixated upon and N is the number of image pixels. While this formulation is somewhat simplistic, it will allow for the use of novel loss functions that are suitable for training deep models via back-propagation. In the sequel, we first describe these loss functions and then describe our model implementation.

3.3.1 Learning to Predict the Probability of Fixations

We adopt an end-to-end learning framework in which a fully-convolutional network is trained on 'image and ground-truth saliency map ($\boldsymbol{g}$)' pairs. The network predicts the spatial distributions $\boldsymbol{p}$. Both the distributions, $\boldsymbol{g}$ and $\boldsymbol{p}$, are modelled probabilistically via the use of the softmax activation function:

$$p_i = \frac{e^{x_i^p}}{\sum_j e^{x_j^p}}, \qquad g_i = \frac{e^{x_i^g}}{\sum_j e^{x_j^g}}, \tag{3.1}$$

where $\boldsymbol{x} = (x_1, \cdots, x_i, \cdots, x_N)$ is the set of un-normalised saliency response values for either the ground-truth map ($\boldsymbol{x^g}$) or the predicted map ($\boldsymbol{x^p}$). To compute $\boldsymbol{x^g}$, a binary fixation map $\boldsymbol{b}$ is first generated from ground-truth eye-fixations. The binary map $\boldsymbol{b}$ is then convolved with a Gaussian kernel as described earlier in this section to produce $\boldsymbol{y}$. The smoothed map $\boldsymbol{y}$ is then normalised as

$$x_i^g = \frac{y_i - \min[\boldsymbol{y}]}{\max[\boldsymbol{y}] - \min[\boldsymbol{y}]}. \tag{3.2}$$

We generate $\boldsymbol{x^p}$ directly from the last response map of our deep network, whose architecture is described in the next section.

We propose to combine the softmax function with distance measures appropriate for probability distributions in order to construct objective functions to be used for training the network. This combination is inspired by the popular and effective softmax/cross-entropy loss pairing which is often used to train models for multinomial logistic regression.

In our case, we propose to combine the softmax functions with the χ^2, total-variation, cosine and Bhattacharyya distance measures, as listed in Table 3.1. To our knowledge, these pairings have not previously been used to train a network for probability distribution prediction. We also investigate the use of the KL divergence measure, the minimisation of which is equivalent to cross-entropy minimisation, and which is used extensively to learn regression models in deep networks. The partial derivatives of these loss functions with respect to x_i^p are all of the form $ap_i - b(1 - p_i)$ due to the pairing with the softmax function, whose partial derivative with respect to x_i^p is

$$\frac{\partial p_j}{\partial x_i^p} = \begin{cases} p_i(1 - p_i), & \text{if } j = i \\ -p_i p_j, & \text{otherwise.} \end{cases} \tag{3.3}$$

We make comparisons with two standard regression losses, the Euclidean and Huber losses, defined as:

$$L_{euc}(\boldsymbol{p}, \boldsymbol{g}) = \sum_j a_j^2, \tag{3.4}$$

and

$$L_{hub}(\boldsymbol{p}, \boldsymbol{g}) = \sum_j \begin{cases} \frac{1}{2} a_j^2, & \text{for } |a_j| \leq 1 \\ |a_j| - \frac{1}{2}, & \text{otherwise;} \end{cases} \tag{3.5}$$

where $a_j = |p_j - g_j|$.

3.3.2 Training the Prediction Model

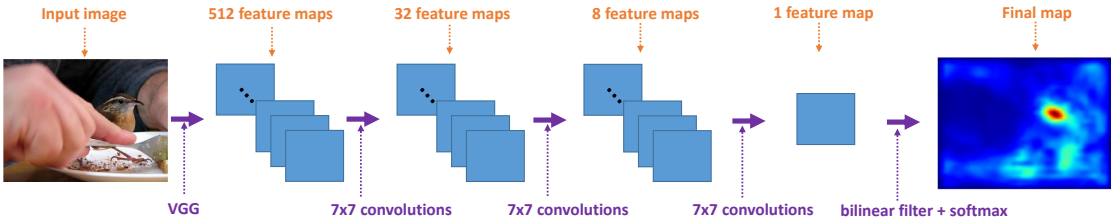


Figure 3.2: Our proposed saliency map extraction pipeline: the input image is introduced into a convnet with an architecture that is identical to the convolutional portion of VGGNet. Additional convolutional layers are then applied, resulting in a single response map which is upsampled and softmax-normalised at test time to produce the final saliency map.

The network architecture and saliency map extraction pipeline is shown in Figure 3.2. We use the convolutional layers of the VGGNet model (Simonyan & Zisserman, 2015), trained on ImageNet images for the task of classification, as the early layers of our model. This convolutional sub-network has been shown to provide good local feature maps for a variety of tasks including object localisation (Ren et al., 2015) and semantic segmentation (Long et al., 2015). As saliency datasets tend to be much too small to train such large networks from random initialisations (the largest dataset has 15K images, compared to 1M for ImageNet), it is essential to initialise with a pre-trained network. We then progressively decrease the number of feature maps using additional convolutional layers, until a final down-sampled saliency map is produced. For this, we add three new layers, rather than just one, to predict the final map in order to improve both discriminability and generalisability (Simonyan & Zisserman, 2015). We experimented with different filter sizes besides 7×7 (*e.g.* 9×9, 5×5, 3×3) and found no significant

performance difference. We explicitly avoided fully-connected layers in order to obtain a memory and time-efficient model. The three new layers are initialised with a uniform Gaussian distribution of sigma = 0.01. Because the response maps undergo several max-pooling operations, the predicted saliency map $\boldsymbol{p}$ is of a lower resolution than the input image. The ground-truth map $\boldsymbol{g}$ is therefore downsampled during training to match the dimensions of $\boldsymbol{p}$. Conversely, during inference the predicted map is upsampled with a bilinear filter to match the dimensions of the input image (see Figure 3.2), and the softmax function is applied to normalise it to a probability distribution.

The final fully-convolutional network thus consists of 16 convolutional layers, each of which is followed by a ReLu layer. Due to the fully-convolutional architecture, the size is quite small for a deep model, with only 15,530,481 weights (60MB of disk space).

Note that while several deep saliency models explicitly include a center bias (see e.g. the work of Kruthiventi et al. (2015)), we hypothesise that the model can learn the center-bias implicitly, given that it is largely an artifact of a composition bias in which photographers tend to place highly salient objects in the center of images (Borji & Tanner, 2015). We test this by adding Gaussian blurring and a center-bias to our maps, with optimised parameters, using the post-processing code of the MIT saliency benchmark (Bylinskii et al., 2012). We find no consistent improvement across different metrics using this post-processing which indicates that a great deal of center-bias and Gaussian blurring is already accounted for in the model.

The objective function is optimised using stochastic gradient descent, with a learning rate of $1 \times$ *global learning rate* for newly-introduced layers and $0.1 \times$ *global learning rate* for those layers which have been pre-trained on ImageNet. To reduce the training time, the first 4 convolutional layers are fixed and thus retain their pre-trained values. We use a momentum of 0.9 and a weight decay of 0.0005. The model is implemented in Caffe (Jia et al., 2014). We train the network using an Nvidia K40 GPU. Training on the SALICON training set takes 30 hours.

Saliency datasets tend to have semantic biases and other idiosyncrasies related to the complexity of collecting eye-tracking information (such as the viewing distance to the screen and the eye-tracker calibration). For this reason, we perform dataset-specific

fine-tuning which tends to improve the network performance. Fine-tuning is particularly essential in our case because the SALICON dataset based on mouse clicks in lieu of actual eye-fixations, though highly correlated to the natural saliency response, is still an approximation to the true human eye movements. As shown on a subset of the SALICON images, image-level conformance between SALICON fixations and human eye fixations can be as low as a shuffled-AUC (sAUC) of 0.655 and as high as an sAUC of 0.965 (Jiang et al., 2015). Therefore it is beneficial to fine-tune the network for each dataset of interest. A detailed description of each of these datasets follows.

3.4 Experimental Evaluation

This section describes the experimental datasets used for training and evaluating the saliency prediction models followed by a discussion of the quantitative and qualitative aspects of the results.

3.4.1 Datasets

SALICON: This is one of the largest saliency datasets available in the public domain (Jiang et al., 2015). It consists of eye-fixation information for 20K images from the MS-COCO dataset (Lin et al., 2014a). These images contain diverse indoor and outdoor scenes and display a range of scene clutter. 10K images are marked for training, 5K for validation and 5K for testing. The fixation data for the test set is held out and performance on it must be evaluated on a remote server. The peculiarity of SALICON lies in its mouse-based paradigm for fixation gathering. The attentional focus (foveation) in the human attention mechanism that defines saliency fixations is simulated using the movements of a mouse over a blurred image. The approximate foveal image region around the mouse position is selectively un-blurred as the user explores the image scene using the mouse cursor. As evaluated on a subset of the dataset, this mouse-movement data is, in general, highly consistent with human eye fixations (at 0.89 sAUC). Therefore, while the mouse-fixation data is an approximation to the human baseline, it is useful in adapting the weights of a deep neural network originally trained for a different task to the new task of saliency prediction. We use this dataset for a comparative study of the

performances of the selected probability distances when used as loss functions for training network parameters. We also submitted our best performing model to the SALICON challenge server (Zhang et al., 2015).

MIT-1003: This dataset was introduced as part of the training and testing paradigm of Judd et al. (2009). The eye-tracking data is collected using a head-mounted eye-tracking device for 15 different viewers. The 1003 images of this dataset cover natural indoor and outdoor scenes. For our experiments, we use the first 900 images for training and the remaining 103 for validation, similar to the paradigm of Kruthiventi et al. (2015).

MIT-300: This benchmark consists of a held out eye-tracking data for 300 images collected across 39 different viewers (Judd et al., 2012). The data collection paradigm for this dataset is very similar to that of MIT-1003. Hence, as suggested for this online benchmark, we use MIT-1003 as the training data to fine-tune the network performance.

OSIE: This benchmark contains a set of 700 images. These include natural indoor and outdoor scenes, as well as high-quality aesthetic pictures taken from Flickr and Google. In order to benefit from a top-down understanding, this dataset provides object and semantic level information (which *we do not use*) along with the eye-tracking data. Following the work of Luo et al. (2015), we randomly divide the set into 500 training and 200 test images and average the results over a 10-fold cross-validation.

VOCA-2012: With the exception of SALICON, the above mentioned datasets are relatively small. Evaluation of the proposed method on large-scale datasets of real fixations would definitely be more informative. However, to our knowledge, there is no such *truly* large-scale dataset of free-viewing fixations. Instead, we evaluate on VOCA-2012, an action recognition dataset which has been augmented with task-dependent eye-fixation data (Mathe & Sminchisescu, 2013). Predicting such fixations is a different task to predicting free-viewing fixations, the task for which our model is designed. However, we still evaluate our method on this dataset to determine its generalisability.

Note: To create ground-truth saliency maps from fixation data, we use the saliency map generation schemes proposed by the authors of the respective datasets. For SALICON, this corresponds to convolving the binary fixation maps with a Gaussian kernel of width

153 and standard deviation 19 (all in units of pixels). For OSIE, this means applying a Gaussian kernel of width 168 and standard deviation 24. The authors of MIT-1003 and MIT-300 provide ground-truth saliency maps which, according to their technical report (Judd et al., 2012), are computed with a Gaussian kernel whose size corresponds to a cut-off frequency of 8 cycles per image.

3.4.2 Results

We first draw a comparison between the results obtained using different loss functions, and then compare them to the performance of state-of-the-art methods. For each dataset, we follow the established evaluation protocol and report results on standard saliency metrics that include sAUC, AUC-Judd, AUC-Borji, Correlation Coefficient (CC), Normalised Scanpath Saliency (NSS), Similarity (SIM), and Earth Mover's Distance (EMD).

Comparison of Loss Functions: We compare the performances of models trained using the proposed loss functions to those trained using standard loss functions such as the Euclidean distance, Huber distance, and KL-divergence. These models are all trained on the SALICON training set of 10K images, and validated on the SALICON validation set of 5K images. Table 3.2 presents the best validation-set performance for each of the above mentioned loss functions, measured in terms of 4 saliency metrics.

These results show that: (a) the loss functions based on distances that are suitable for measuring disparity between probability distributions perform better than standard regression losses; (b) KL-divergence based loss function compares favourably with others; and (c) Bhattacharyya distance based loss function outperforms all other loss functions. The last two loss functions share the property of being robust to outliers, by suppressing large differences between probability values (logarithmically in the case of KL divergence and geometrically in the case of Bhattacharyya distance). This robustness is particularly important given that the ground-truth saliency maps are derived from eye-fixations which exhibit inherent variance owing to the subjectivity of visual attention, and which may also contain stray fixations and other kinds of noise. Figure 3.3 shows the evolution of

the saliency metrics on the SALICON validation set as the training progresses. As can be observed, Bhattacharyya distance is consistently the best performing loss function.

Distance	AUC-Judd	sAUC	CC	NSS
Euclidean	0.865	0.761	0.667	2.108
Huber	0.867	0.766	0.684	2.177
KL divergence	0.876	0.780	0.724	2.371
χ^2 divergence	0.872	0.774	0.711	2.337
Total Variation distance	0.869	0.766	0.716	2.385
Cosine distance	0.871	0.778	0.717	2.363
Bhattacharyya distance	**0.880**	**0.783**	**0.740**	**2.419**

Table 3.2: SALICON validation set: Performance comparison of models trained using different loss functions.

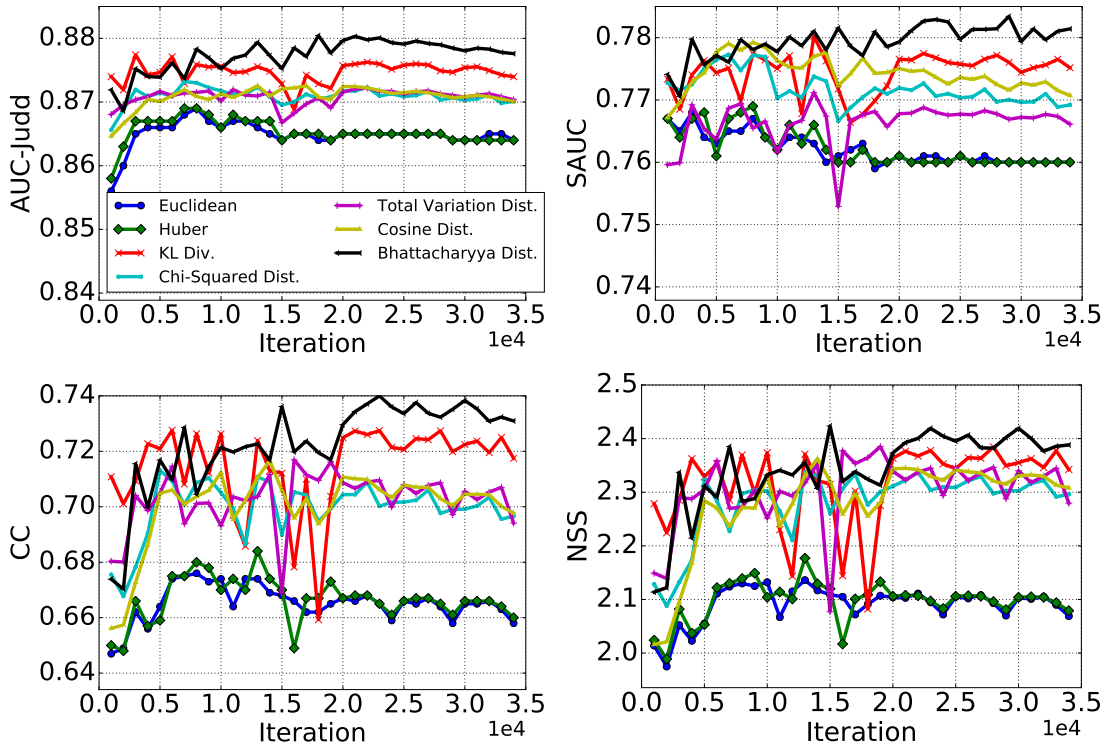


Figure 3.3: Evolution of different saliency metrics on SALICON validation set with respect to the number of training iterations.

Comparison to the State-of-the-art on Various Datasets: We compare the performance of our proposed model, using the Bhattacharyya distance, with the state-of-the-art methods on four public saliency benchmarks as follows.

Method	CC	sAUC	AUC-Borji
Itti(Itti et al., 1998)	0.205	0.610	0.660
GBVS(Harel et al., 2006)	0.421	0.630	0.782
BMS(Zhang & Sclaroff, 2013)	0.427	0.694	0.770
WHU_IIP*	0.457	0.606	0.776
Xidian*	0.481	0.681	0.800
Rare12_Improved*	0.511	0.664	0.805
UPC(Pan & i Nieto, 2015)	0.596	0.670	0.829
PDP	**0.765**	**0.781**	**0.882**

Table 3.3: SALICON Challenge (test): Comparison between different state-of-the-art methods. Methods marked with a * have no associated publication to-date.

Method	AUC-Judd	SIM	EMD	AUC-Borji	sAUC	CC	NSS
eDN(Vig et al., 2014)	0.82	0.41	4.56	0.81	0.62	0.45	1.14
BMS(Zhang & Sclaroff, 2013)	0.83	0.51	3.35	0.82	0.65	0.55	1.41
SALICON(Huang et al., 2015)	**0.87**	0.60	2.62	0.85	**0.74**	0.74	2.12
DeepFix(Kruthiventi et al., 2015)	**0.87**	**0.67**	**2.04**	**0.80**	0.71	**0.78**	**2.26**
PDP	0.85	0.60	2.58	0.80	0.73	0.70	2.05

Table 3.4: MIT-300: Comparison with the top-performing methods on this benchmark.

SALICON challenge (test): The saliency estimation challenge (Zhang et al., 2015) consists of predicting saliency maps for 5K images held out from the SALICON dataset. Table 3.3 shows results for the state-of-the-art methods and our approach, which we call PDP for probability distribution prediction. We outperform all published results, to our knowledge, on this dataset across all three test metrics.

MIT-300: MIT-1003 images serve as the training set for adapting the network performance to this benchmark. The results are compared in Table 3.4. We perform at par with the state-of-the-art methods. Note that DeepFix (Kruthiventi et al., 2015) incorporates external cues such as center and horizon biases in its models. We believe that including such cues may marginally benefit our model as well. In addition, they use a larger architecture, but train with a regression loss. Therefore our approach may complement theirs. Fine-tuning on MIT-1003 could only be performed using a batch size of 1, owing to large variations in size and aspect ratio of the images. We observed that a much reduced momentum of 0.70 improved stability and allowed for an effective learning of the model with the constrained batch size.

Method	sAUC
Itti(Itti et al., 1998)	0.658
SUN(Zhang et al., 2008)	0.735
Signature(Hou et al., 2012)	0.749
GBVS(Harel et al., 2006)	0.706
LCQS-baseline(Luo et al., 2015)	0.765
PDP	**0.797**

Table 3.5: OSIE: The performance metric i.e. shuffled AUC (sAUC) is averaged over 10-fold cross validation. (Baseline results are taken from Luo et al. (2015).)

Method	KL	AUC
HOG detector* (Mathe & Sminchisescu, 2013)	8.54	0.736
Judd et al.* (Judd et al., 2009)	11.00	0.715
Itti & Koch (Itti & Koch, 2000)	16.53	0.533
central bias (Mathe & Sminchisescu, 2013)	9.59	0.780
human (Mathe & Sminchisescu, 2013)	6.14	0.922
PDP(without finetuning)	**7.92**	0.845
PDP*(with finetuning)	8.23	**0.875**

Table 3.6: VOCA: Performance comparison using KL-divergence and AUC measures. Note that the best performance above, using a *human*, corresponds to the use of fixations of one human observer as the predicted fixations of the remaining observers. The results in bold indicate the best performance without the use of any human intervention at test time, and correspond to the models proposed in this work. (* denotes the methods that have been trained on this particular dataset.)

OSIE benchmark: The performance comparison on this dataset is done using 10-fold cross validation by randomly dividing the dataset into 500 training and 200 validation images. Table 3.5 shows that PDP achieves the highest sAUC score. This dataset contains a wide variety of image content and aesthetic properties. Nonetheless, the small set of 500 images is sufficient to successfully adapt our model.

VOCA-2012 (Generalisation to task-dependent fixation prediction): We ran experiments on the VOCA-2012 dataset using the same experimental paradigm as adopted by Mathe & Sminchisescu (2013). We used our final SALICON-trained model to predict maps for test images both before and after fine-tuning the model on training images from VOCA-2012. The results summarised in Table 3.6 show that our method, both with and without finetuning, outperforms the state-of-the-art (Mathe & Sminchisescu, 2013). This suggests that the task-dependent fixations for this action recognition dataset are highly consistent with the free-viewing fixations.

Figure 3.4: Comparison of BMS (Zhang & Sclaroff, 2013), SALICON (Huang et al., 2015), and our proposed PDP method on randomly sampled images from MIT-1003 dataset. GT refers to the ground-truth saliency maps. Note that, to ensure a fair comparison, the PDP results shown here are obtained from a network that is trained only on SALICON images, with no fine-tuning on the MIT-1003 dataset.

3.4.3 Discussion

Our probabilistic perspective to saliency estimation is intuitive in two ways. First, it takes into account the fact that attention is a competitive phenomenon, i.e. we look at certain regions in the image at the expense of others. Hence, the fixation map normalised over the total visual stimulus can be understood as a spatial probability distribution. Secondly, a probabilistic framework allows the model to account for the subjectiveness of the process over different human users and for any noise in the data collection paradigm.

To provide a qualitative insight, some randomly selected predicted maps are shown in Figure 3.4. Our method consistently gives high fixation probabilities to areas of high center-surround contrast, and also to high-level cues such as bodies, faces and, to a lesser extent, text. The higher emphasis on bodies and faces as compared to text is likely due to the large number of images containing people and faces in the SALICON dataset.

Figure 3.5 shows saliency map predictions for SALICON training images obtained on the forward pass, after a certain number of training images have been used to train the model. One can see that center-surround contrast cues are learned very quickly, after having seen fewer than 50 images. Faces (both of animate and inanimate objects) are also learned quickly, having seen fewer than 100 images. The saliency of text also emerges fairly rapidly. However, the cue is not as strongly identified, likely due to the relatively smaller amount of training data involving text.

3.5 Conclusion

We introduce a novel saliency formulation scheme and associated prediction model for estimating saliency maps given input images. We train a deep network using an objective function which penalises the distance between target and predicted maps in the form of probability distributions. Experiments on four different datasets demonstrate the superior performance of our method with respect to other loss functions and other state-of-the-art saliency estimation methods. We thus illustrate the benefit of using suitable learning criteria adapted to this task.

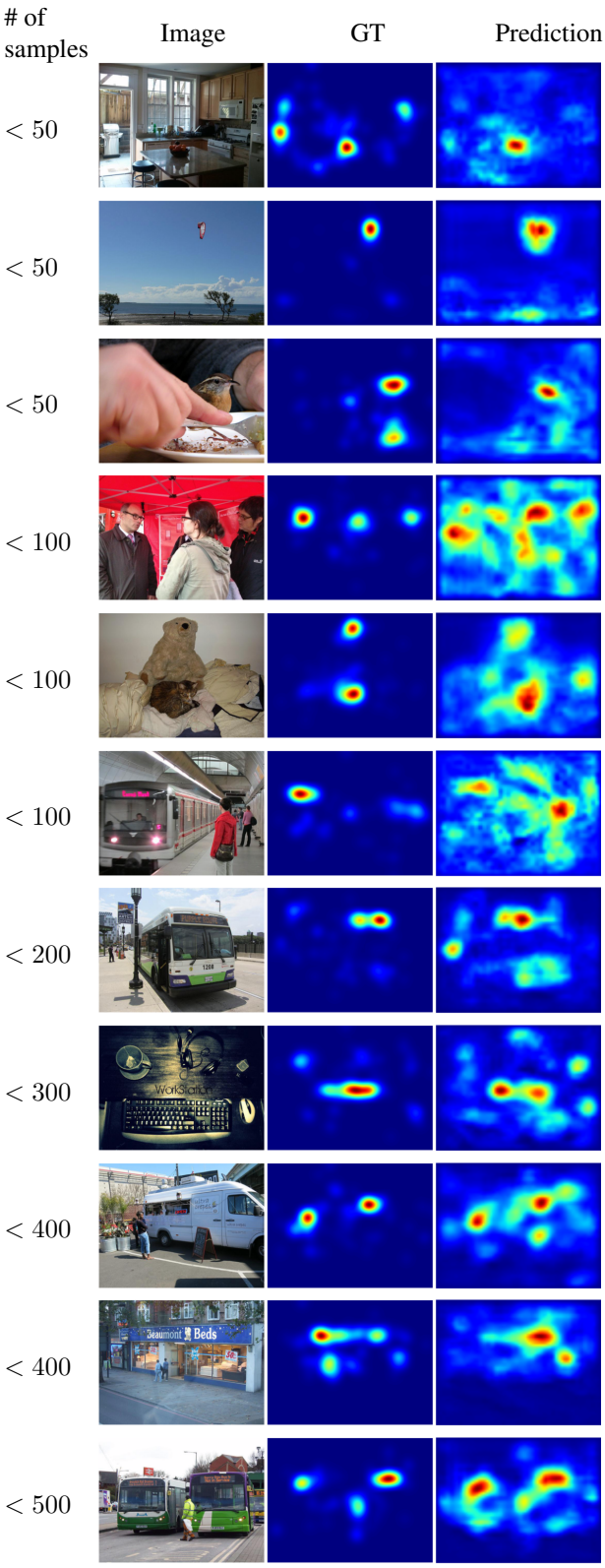


Figure 3.5: Our method quickly learns that regions of high center-surround contrast, and faces and heads, are salient. The *leftmost* column shows the number of samples seen by the model during training before it predicts the saliency maps in the *rightmost* column.

4

Straight to Shapes: Real-time Detection of Encoded Shapes

Contents

Current object detection approaches predict bounding boxes that provide little instance-specific information beyond location, scale and aspect ratio. In this work, we propose to regress directly to objects' shapes in addition to their bounding boxes and categories. It is crucial to find an appropriate shape representation that is compact and decodable, and in which objects can be compared for higher-order concepts such as view similarity, pose

variation and occlusion. To achieve this, we use a denoising convolutional auto-encoder to learn a low-dimensional shape embedding space. We place the decoder network after a fast end-to-end deep convolutional network that is trained to regress directly to the shape vectors provided by the auto-encoder. This yields what to the best of our knowledge is the first real-time shape prediction network, running at 35 FPS on a high-end desktop. With higher-order shape reasoning well-integrated into the network pipeline, the network shows the useful practical quality of generalising to unseen categories that are similar to the ones in the training set, something that most existing approaches fail to handle.

4.1 Introduction

Automatically detecting (Girshick, 2015) and delineating (Dai et al., 2015) object instances in images is a core problem in computer vision, with wide-ranging applications. For example, knowing the individual boundaries of nearby objects can allow robots to grasp them (Rao et al., 2010). To help visually-impaired people become more independent, specially-designed glasses can highlight the boundaries of objects with which to interact (Hicks et al., 2013).

For these applications, real-time frame processing is crucial, and the information provided by bounding box predictions (Ren et al., 2015) or a pixel-wise semantic segmentation of the scene (Zheng et al., 2015) is not enough. A bounding box captures no more than the location, scale and aspect ratio of an object, a fairly coarse representation that provides no information about the object's boundary. On the other hand, bottom-up pixelwise labelling approaches have no explicit notion of local and global object shape. To achieve adherence to local boundaries and impose spatial and appearance consistency in the semantic space, recent works (Chen et al., 2016; Zheng et al., 2015) post-process using conditional random fields (CRFs). Such post-processing quickly becomes intractable for the higher-order constructs required to capture object structure, pose and occlusions.

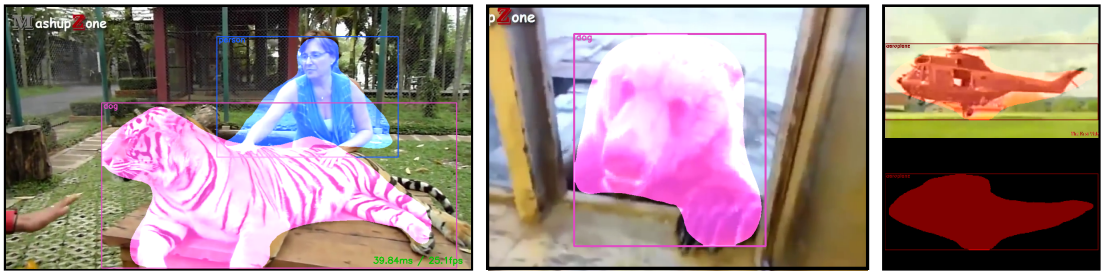
In response to the above-mentioned shortcomings, we implement an embedding space that incorporates notions of object shape, pose and occlusion patterns. A deep regression network is then trained to map input image patches to this embedding space to tease apart an object's category and its shape mask. Figure 4.1 shows an example of our network

Figure 4.1: Testing direct regression to shape on a validation image from PASCAL VOC (Everingham et al., 2005) using **(a)** binary shape masks, **(b)** radial vectors, and **(c)** a learned shape embedding.

regressing to three different shape representations. Observably, object shape and category are correlated for many classes, so training the learner with a single objective function that includes both terms is mutually-reinforcing.

We propose an embedding space that is learnt on the binary instance masks in a class-agnostic manner. By design, the embedding space is *compact*, *decodable* (supports mapping of binary instance masks in and out of the space), *continuous* (instance masks degrade gracefully around the point of interest in the space) and *interpretable* (one in which we can reason about the similarities between shapes and their corresponding categories). Our formulation is thus able to leverage shape reasoning and extend the prediction of shape masks to object categories our network has never seen before, but which have similar characteristics to the ones seen during training, like the tiger, bear and helicopter in Figure 4.2.

It is also important to note that we tackle the detection of shapes of individual instances in a top-down paradigm with a single sweep of the network. This is in contrast to recent works (Dai et al., 2015; Arnab & Torr, 2016) combining top-down and bottom-up paradigms with various sequential arrangements of bounding-box detection, pixel-wise segmentation and category recognition to achieve instance segmentation. Due to a sequential processing, the error in the above networks is additive. For instance, if



(a) (b) (c)

Figure 4.2: Detecting the shapes of previously-unseen categories in images from YouTube videos: **(a)** a person strokes a tiger, **(b)** a bear is fed through a window, and **(c)** a helicopter crashes into the ground (see supplementary video). The tiger and bear are both detected as class 'dog', and the helicopter is detected as 'aeroplane'. Despite wrong categories, the predicted shape masks capture useful information about the types of objects that are present.

bounding-box detection comes first and cuts through an object boundary, pixel-wise labelling inside the box can never retrieve the correct object boundary. Likewise, if detection follows pixel-wise labelling, it is difficult to estimate object boundaries for overlapping instances of the same class.

Contributions: The proposed approach overcomes the aforementioned difficulties by directly regressing to multiple object locations, shapes, and categories, as shown in Figure 4.3. Crucial to the process is the learning of a compact and decodable embedding space in which shapes can be described and compared. To this end, we demonstrate the use of a denoising auto-encoder (Vincent et al., 2008) in encoding real-world shape templates. Moreover, the single-sweep processing affords our implementation real-time capabilities. To the best of our knowledge, this is the first real-time shape prediction network, running at 35 FPS on a high-end desktop with an i7-4960 Processor (3.6GHz, 12-core) and a Titan X GPU. Our shape prediction network thus builds upon what is currently possible at the intersection of object detection (Redmon et al., 2016), instance segmentation (Hariharan et al., 2014) and joint embeddings (Li et al., 2015).

4.2 Related Work

Importance of Shape Cues: In the past, shape cues have been successfully used to guide object recognition (Belongie et al., 2002) and localisation (Jain et al., 1996), by virtue of their ability to capture discriminative, category-specific details. With the

(a) (b) (c)

Figure 4.3: Regressing to single or multiple instances with large intra-category shape variation: consider e.g. the shape of **(a)** a sheep's head, in contrast to **(b)** its body. In **(c)**, we detect multiple overlapping shapes, in this case a person on a bike.

advent of deep networks, suitable features best adapted to the task are assumed to be learnt by the network during training (LeCun & Bengio, 1995), by optimising an appropriate loss function. In existing object detection pipelines, this loss is computed on the 4D bounding box specifications (Redmon et al., 2016), ignoring the use of shape to guide learning. By contrast, regressing to detailed shape vectors affords the network a more informed supervision.

Predicting Object Shape Masks: Predicting object shape masks has previously been identified as an important practical challenge (Ferrari et al., 2009; Marszałek & Schmid, 2007). However, these approaches contain independently optimised processing blocks and have shown limited success, working only for images containing a single prominent object and/or a high foreground-background contrast. More recently, Pinheiro et al. (2015, 2016) have explored the practical benefits of learning to predict object proposals as pixel-wise segmentation masks. However, their networks are composed of disjoint stages for predicting segmentation masks, object locations and object classes. By contrast, and in a spirit similar to YOLO (Redmon et al., 2016), the architecture that we propose for predicting shape embeddings (from which the shape masks can be reconstructed) is simpler and contains only one network, with a single objective.

Extending Bounding Box Detectors to Shape: Recent progress in bounding box object detection has been fuelled by the representational power of deep networks. R-CNN (Girshick et al., 2014) made massive leaps in detection accuracy by using a deep network to predict the category of pre-generated bounding box proposals. Soon after,

Faster R-CNN (Ren et al., 2015) emerged. It does away with separate stages for region proposal generation and classification, instead using a region proposal network that shares convolutional features with the classification network, achieving benefits in computational efficiency and accuracy. More recently, the YOLO (Redmon et al., 2016) detection network streamlined object detection by replacing the separate proposal generation and classification stages with regression to spatially-separated bounding boxes and class probabilities. This resulted in real-time object detection rates. Inspired by YOLO, we extend the network to predict shape encodings, achieving similar speeds, but with the advantage of incorporating additional shape information.

Shape Embeddings: The simplest shape embedding space we explore in this work consists of appropriately down-sampled binary masks. Other shape representations are also possible starting points for establishing this space; a comprehensive review of shape representations can be found in Zhang & Lu (2004). Moreover, inspired by the way in which Mikolov et al. (2013) learn word embeddings to find a feature space in which similar words are close together, we aim to learn a *shape* embedding in which to reason about the shape of objects. In contrast to Mikolov et al. (2013), however, we seek a lower-dimensional encoding (compared to the dimensions of a binary shape mask) to make detection by regression feasible. We observe the compression, noise-handling and reconstruction capabilities of the learned shape encodings to be significantly superior to the hand-crafted shape representations.

Learned Shape Embeddings: Hinton & Salakhutdinov (2006) demonstrated the use of auto-encoders for learning latent disentangling representations for small binary images of digits and faces. More recently, the Shape Boltzmann Machine (SBM) of Eslami et al. (2014) makes use of a deep restricted Boltzmann architecture to model a shape space that is realistic and generalisable. Notably, both the above networks are trained to model shapes and are not optimised for reconstruction. Also, the fully-connected construction of SBM restricts its application to small synthetic images with a maximum resolution of around 50×50. To circumvent this practical limitation, we incorporate convolutional

layers, and train the model as a denoising auto-encoder (Vincent et al., 2008). Thus, in a first, we extend the full benefit of auto-encoders to real-world shape templates.

Our work is also related to that of Li et al. (2015), which handles single image-to-embedding mappings (Tekin et al., 2016) and assumes that objects in the input image are prominent. By contrast, we regress to multiple shape embeddings, thereby extending the approach to object detection (see Figure 4.3). Another important distinction is that the shape embeddings in Li et al. (2015) are hand-crafted and non-decodable, preventing the synthesis of new shapes from arbitrary coordinates in the embedding space and limiting the applications to retrieval with nearest-neighbour search. On the contrary, by learning a continuous and decodable shape embedding, we are able to predict masks for categories that have not been observed previously but share close shape similarity with those in the train set (see Figure 4.2). We are able to achieve this performance at a fraction of the dimensionality of hand-crafted shape representations.

Potential Applications - Instance Segmentation and Occlusion Handling: Recent works (Gupta et al., 2014; Dai et al., 2015) on instance segmentation attempt to combine the benefits of the top-down and bottom-up paradigms by performing pixel-wise class labelling within each predicted bounding box. Such bounding-box based approaches err primarily for occluded objects and their boundaries. The bottom-up approach of Chen et al. (2015) resorts to post-hoc object-level or category-specific reasoning to combine the disjoint parts of object instances. Also noteworthy is the work by Dai et al. (2016a), which extends a fully-convolutional network (FCN) (Long et al., 2015) to predict position-sensitive instance score maps that then need to be assembled into object instances at the output. Another approach (Romera-Paredes & Torr, 2015) makes use of a recurrent neural network (RNN) architecture, which is trained with a single loss function in order to predict object instances sequentially. Although this has shown promise on a limited set of object categories, it has difficulty remembering all past predictions, and therefore finding those that it needs to predict next.

Ultimately, describing an object as a set of pixels with an associated category has its limitations. In particular, shapes represented as pixel sets are hard to compare: there is no

obvious way of computing *meaningful* distances between high-dimensional object masks. This is unfortunate, because the ability to compare shapes is extremely useful: the shape of an object is not merely an abstract property, but can indicate something fundamental about the object's capabilities and role. For example, whilst there is significant variety amongst the many legged animals in the world, they are clearly more similar to each other than to an inanimate object such as a lorry, and the shapes involved reflect this. As such, if we can find a representation in which we are able to compare shapes quantitatively, we can use the distance values to provide important clues about objects, even for categories on which we have not trained. For these reasons, we tackle the shape prediction problem as part of an end-to-end pipeline, using an intermediate embedding space that knows about realistic category-level poses, occlusions and shape priors, as learnt from the training data.

4.3 Proposed Approach

An object's shape is one of its fundamental properties, which, when coupled with its location, scale and category, gives rich information about its possible pose and coarse depth, and how one can interact with it. To take advantage of this, we therefore extend the state-of-the-art YOLO (Redmon et al., 2016) object detection network that we mentioned in §4.2 to regress to not only object locations, confidence scores and conditional probabilities for each category, but also detailed shape encodings, as illustrated in Figure 4.4. The details of how we do this are described in §4.3.1.

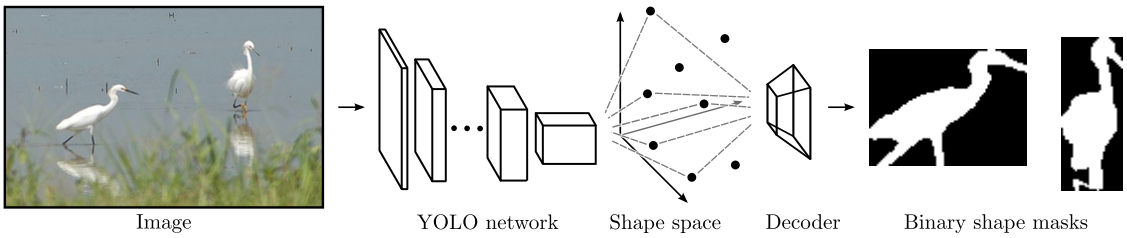


Figure 4.4: An input image is rescaled and passed through a single network that regresses directly to multiple object hypotheses (category, location, aspect and shape). The learned shape encoding is compact, decodable, continuous and interpretable, allowing the prediction of shapes not seen in the training data, and enabling more meaningful distance comparisons between shapes. Note that predicting encoded shapes enables higher-order reasoning about the pose of the bird (upright, facing right) and how it compares to other birds: this would be impossible from a bounding box alone. Moreover, the current shape space is learned in an unsupervised manner and any annotation superimposed on the shape space would add to the predicted information at test time.

For the purpose of object shape prediction as described above, we require a shape embedding that is both compact and decodable. Moreover, we would benefit immensely from a representation that embodies higher-order understanding of object shape, realistic poses and occlusion patterns. For instance, we would like the distances between shapes in the representation to reflect our own understanding of the similarity between shapes. To this end, in §4.3.2 we investigate three decodable shape representations: i) downsampled binary masks, ii) radial vectors and iii) learned shape encodings.

In §4.4, we join the ideas from §4.3.1 and §4.3.2, and evaluate the merits of our real-time shape detection pipeline.

4.3.1 Deep Regression Network

This section draws inspiration from the YOLO detection algorithm (Redmon et al., 2016) and extends it for the prediction of shapes. Like YOLO, our network reasons globally about objects in the image and is trained to minimise a single objective function. However, we augment YOLO's object representation with an encoded vector that denotes the object's shape.

The YOLO pipeline starts by dividing the input image into an $S \times S$ grid. If the centre of an object lies within a cell, then that cell becomes responsible for detecting that object. Here, each grid cell predicts $B = 2$ sets of boxes, confidence scores and shapes with a dimensionality of N per set, as well as a probability mass function.

The box prediction corresponds to the 4-coordinates that fully specify the tightest fitting bounding box to the object. The confidence score is calculated as the intersection over union (IoU) between the predicted and target box, if a target box exists, and zero otherwise. Each shape encoding represents the shape of an object, independent of its location and aspect ratio. For instance, in order to calculate the binary shape mask that we provide as a target for learning, we take the ground truth object segmentation mask, binarise it and rescale it as part of the encoding process. Overall, if the shape is to be encoded by a 16×16 binary mask, then $N = 1 + 4 + 256$.

Each grid cell also predicts a conditional probability mass function, which, when multiplied by the object confidence score, results in a score reflecting the confidence

for the category and its overlap:

$$p(c|o) * p(o) * IoU = p(c \cap o) * IoU. \tag{4.1}$$

As input to our network, we feed in 448×448 images, which are transformed using random rotations, translations, spatial scaling and pixel scaling. The network's target is augmented by transforming the ground truth shapes with the same geometric transformations as above. The final size of the target vector for a single image becomes

$$D = S \times S \times (N \times B + C), \tag{4.2}$$

where $S \times S$ is the total number of cells in the image grid, B is the number of parameter sets predicted for every cell, N includes parameters for the confidence score, minimum bounding box and shape encoding, and C is the total number of dataset categories.

For training and inference, we use Darknet (Redmon, 2013–2016). We optimise the sum of squared errors between the predicted network output and a target tensor. The four-component loss function used to train the network can be expressed as $L = L_{\text{box}} + L_{\text{conf}} + L_{\text{shape}} + L_{\text{pmf}}$, where

$$L_{\text{shape}} = \lambda_{\text{shape}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \sum_{k=0}^{N-5} \mathbb{1}_{ij}^{obj} \left(\tau_{ijk} - \hat{\tau}_{ijk}\right)^2. \tag{4.3}$$

Here, τ is the target shape representation and $\hat{\tau}$ is the predicted shape representation. Notably, L_{box} and L_{pmf} are sum-squared losses, while L_{conf} is a binary cross entropy loss. Detailed definitions of these three loss components can be found in Redmon et al. (2016).

Since a prediction only requires one forward pass through the network, it is very fast at test time (§4.4). We use non-maximal suppression to reduce the number of predicted overlapping shapes.

4.3.2 Decodable Shape Representation

The shape embedding space is a crucial part of the proposed formulation. We believe that the tasks of constructing the embedding space and learning a mapping from the image domain to this embedding space are intrinsically independent and can therefore be treated separately. Moreover, doing so makes the two tasks more tractable (Li et al.,

2015). We thus experiment with two hand-crafted representations and one learned shape representation to establish the embedding space, described as follows.

Downsampled Binary Shape Masks: A very simple shape descriptor can be obtained just by downsampling the full-size binary shape mask. Given a fixed descriptor size $d = k \times k$, we downsize the image using OpenCV's `INTER_AREA` resampling approach. For reconstruction, the descriptor can be resized back up again using bicubic interpolation.

Radial Representation: A radial descriptor represents a shape as a series of distances between some centre point within the shape and points deterministically distributed over the shape's boundary. We choose the boundary points by finding where rays cast outwards from the centre point, at angles uniformly distributed over $[0, 2\pi)$, intersect the boundary. To improve shape reconstruction it makes sense to choose a centre point that has a direct line of sight to as much of the boundary as possible. In practice, we construct several radial descriptors for each shape and pick the one with maximal IoU. More details can be found in the supplementary material.

Learned Shape Encoding: The aim is to implement a network that can compress an input binary mask to a comparatively lower-dimensional space. Auto-encoders have already been shown to improve over PCA and logistic PCA at a similar task for digits, curves and faces (Hinton & Salakhutdinov, 2006). Thus, in an initial experiment, we use the fully-connected auto-encoder architecture (784-1000-500-250-30) of Hinton & Salakhutdinov (2006) to learn a shape embedding space for the Caltech-101 silhouettes contained in the train and validation set. In addition to the restricted Boltzmann machine-style pre-training of Hinton & Salakhutdinov (2006), we fine-tune the network weights using cross-entropy error between the predicted and target shape masks. This setup yields a reconstruction IoU of 0.85 on the test set. It becomes evident that this architecture, also employed by Eslami & Williams (2012); Eslami et al. (2014), is not able to exploit the spatial redundancy in visual data due to its fully-connected nature; for the same reason, it is not scalable to large images.

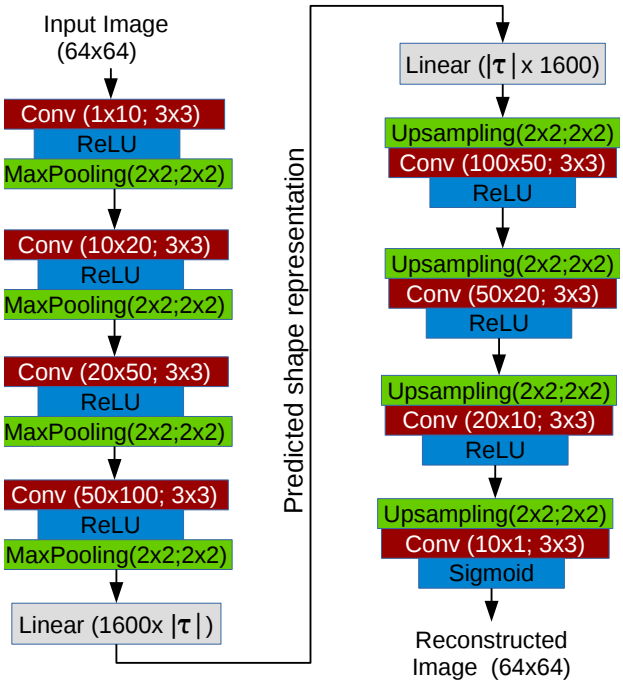


Figure 4.5: Block diagram of the denoising auto-encoder using convolutional layers for learning shape representations of size $|\tau|$. The predicted shape representations corresponding to the ground-truth shape masks are used to train the regression network.

It is important to note that similar shape patterns or boundary formations can be found at different locations in the input image: for example, the front and rear tyre of a bicycle or motorbike look similar. We can leverage this repetition of visual patterns by the use of a convolutional model. Accordingly, a block diagram of our auto-encoder network is shown in Figure 4.5. To prevent the network from learning an identity mapping to a higher-dimensional space in the initial layers of the network meant for data-disentangling, we introduce white noise in the input image so the network generalises to a denoiser (Vincent et al., 2008). We minimise the binary cross-entropy loss to adjust the network weights. It can be simplified as

$$L_{bce} = \sum_{i=1}^{P} \begin{cases} -log(\hat{p}_i) & \text{if } p_i = 1 \\ -log(1-\hat{p}_i) & \text{if } p_i = 0, \end{cases} \tag{4.4}$$

where P is the total number of pixels in the image, $\hat{p}_i$ is the predicted probability and p_i is the pixel value.

4.4 Experiments and Results

Our method allows us to unify detection and segmentation by integrating the higher-order concept of shapes into the end-to-end prediction pipeline. We thus study the application of our proposed prediction network to the task of instance segmentation (§4.4.2), where we find that having the concept of shapes allows it to predict shape masks even for unseen categories. Section 4.4.3 presents a comparative analysis of this zero-shot segmentation capability. Essential to our formulation is the need to reconstruct accurate shape masks from an interpretable and low-dimensional shape space. Thus, we also analyse the trade-offs between three different shape spaces in §4.4.1.

4.4.1 Comparing the Shape Representations

The choice of shape representation is crucial, as it dictates what aspects of a shape are made explicit and the ease with which it can be manipulated (Marr, 1982). We thus compare the shape representations outlined in §4.3.2 on three different axes: i) their ability to accurately represent shapes at low dimensionalities, ii) the continuity of the associated shape space and iii) the structure of the shape space.

Representation Ability: We compute the average shape reconstruction error (one minus the IoU) yielded by each of our three descriptors at different sizes on the SBD dataset. Table 4.1 shows how the error for each descriptor varies with the representation size. At low descriptor dimensionalities the learned encodings are better able to preserve shape than either of the two hand-crafted alternatives. However, the benefits fade away at larger dimensionalities. A qualitative example of the artifacts introduced by lowering the dimensionality of each shape representation can be seen in Figure 4.6: both of the two hand-crafted representations experience a severe decrease in quality as the size decreases, whereas the proposed representation obtained using the denoising auto-encoder still manages to capture the major topological features of the shape even at size 20.

SBD	20 (25)	50 (49)	100	200 (196)	256
Downsampled mask	0.15	0.10	**0.07**	**0.05**	**0.04**
Radial	0.13	**0.08**	**0.07**	0.06	0.06
Shape encoding	**0.125**	**0.08**	**0.07**	0.06	0.06

Table 4.1: Average shape reconstruction error (one minus the IoU, the lower the better) achieved by each of our three descriptors at different sizes on the SBD validation set (Everingham et al., 2005; Hariharan et al., 2011). The unbracketed numbers indicate the sizes used for the radial and shape encoding descriptors; the bracketed numbers indicate the sizes used for the downsampled mask descriptors, which are square by design.

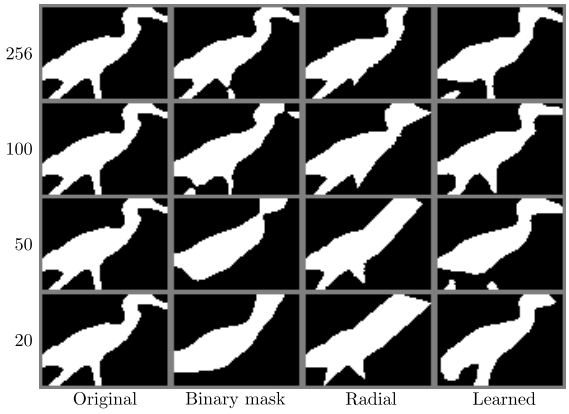


Figure 4.6: As the dimensionality reduces, for both binary mask and radial vector descriptors, the reconstructed shape masks lose precious details, such as the legs and head of the bird, whilst the reconstructed mask for the learned embedding is able to preserve the overall topology of the shape and degrades more gracefully.

Shape Space Continuity: To study the continuity of the shape spaces, we add Gaussian noise to the encodings and visualise its effect on the reconstructed shape masks in Figure 4.7. As expected, an increase in the noise magnitude degrades all representations; however, a little bit of noise in the radial vector distorts the shape beyond recognition. The binary mask loses precision at the boundaries, as the wheels of the bicycle blend into the border of the image. The learned shape representation is able to preserve details such as the seat and the circular wheels better than the hand-crafted alternatives.

Shape Space Structure: We visualise the structure of the shape spaces associated with two representations – the learned encodings and the radial descriptors – in Figure 4.8. Even at a first glance, it is obvious that the shape space for the learned encoding provides a better category-based clustering of the shape masks. For instance, as we move from the top left region of the space to the bottom right along a diagonal, we observe a transition

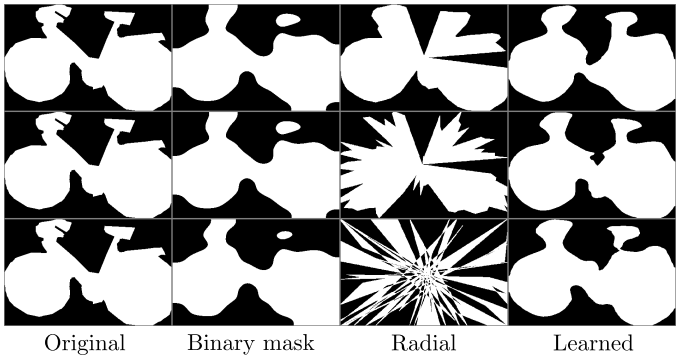


Figure 4.7: The effect of adding multivariate Gaussian noise of zero mean and increasing variance (from top to bottom) to each of the shape representations at dimensionality of 256.

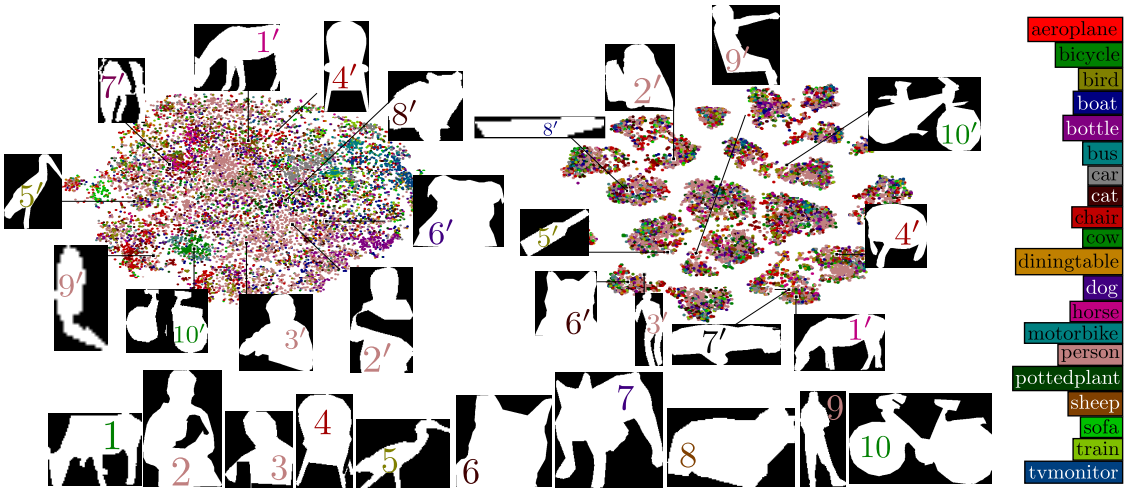


Figure 4.8: Visualising the 20D learned embedding space (left) and the 256D radial vector space (right) using t-SNE (Maaten & Hinton, 2008). Notice the differences between the nearest neighbours (NNs) in the two representation spaces for the anchor templates displayed along the bottom row and marked from 1 through 10. For instance, consider the bicycle anchor (no. 10), which is facing left: its NN in the learned embedding is also facing left (even though it is occluded), whilst its NN in the radial space is facing right.

from samples with large boundary projections (horses and birds) to upright shapes of humans and finally to more streamlined shapes such as bottles. There is no such category-aware structure in the shape space for radial descriptions: indeed, the categories seem to be scattered across various disparate clusters. We also visualise the nearest neighbour of each of a randomly selected set of 10 'anchor' images. The learned embedding space observably shows a closer semantic match between neighbouring shapes, as can be observed in Figure 4.8. A more detailed comparison of the 3 shape descriptors can be found in Figures 4.9, 4.10, and 4.11, where we visualise the embedding spaces produced by the binary masks, radial vectors and learned embeddings respectively. We fix 20

anchor points in each of the three embedding spaces around the main diagonal of the 2D space. For each of the anchor points, we sample the 20 nearest neighbours to observe how the shapes are organised in the space. At a macroscopic level, as we move along the anchor points (across the rows of the tiled images), we notice that the shapes change between being bulky and rounded to shapes with multiple protrusions. For the learned embedding space, this pattern is more pronounced (see Figure 4.11). In the radial space, both bulky and thin shapes are present along a single row (see Figure 4.10 rows 3 and 7), thus in places showing an inferior organisation. Both the binary mask space and the learned embedding space are well-organised for shape similarity. It is interesting to note that the learned embedding space achieves a similar shape organisation to the downsampled binary mask space, but with an order of magnitude reduction in size.

Following this qualitative analysis, we use the PASCAL3D+ (Xiang et al., 2014) dataset to quantify the differences between the structural aspects of the two shape spaces. We do this by summarising object pose and category statistics around neighbourhoods in the respective spaces. To start with, we divide the PASCAL3D+ dataset into a train set of 11578 images and a validation set of 120 images. The validation set is formed by randomly selecting 10 images from each of the 12 rigid object categories of PASCAL3D+ that are a subset of the 20 PASCAL VOC categories. For each validation image as the starting image, we then find the 50 nearest neighbours in the train set and calculate the variances in their poses and their category label statistics with respect to those of the starting image. The results are averaged over all the randomly selected validation images of all the categories, and are compiled in Table 4.2.

We find that the poses (in terms of elevation, azimuth and distance) of the object shapes in the vicinity of a given shape in the learned embedding space are more similar to each other than those in the radial space. This is evident from the lower mean average variance (mAV) values of the pose quantities for the former. Moreover, for the learned shape space, the category that the highest number of neighbouring objects are mapped to is more likely (80% of the time) to match the ground truth category label of the starting shape mask. Thus, overall, the learned shape space can be said to be semantically better organised and more interpretable in comparison to the radial shape space.

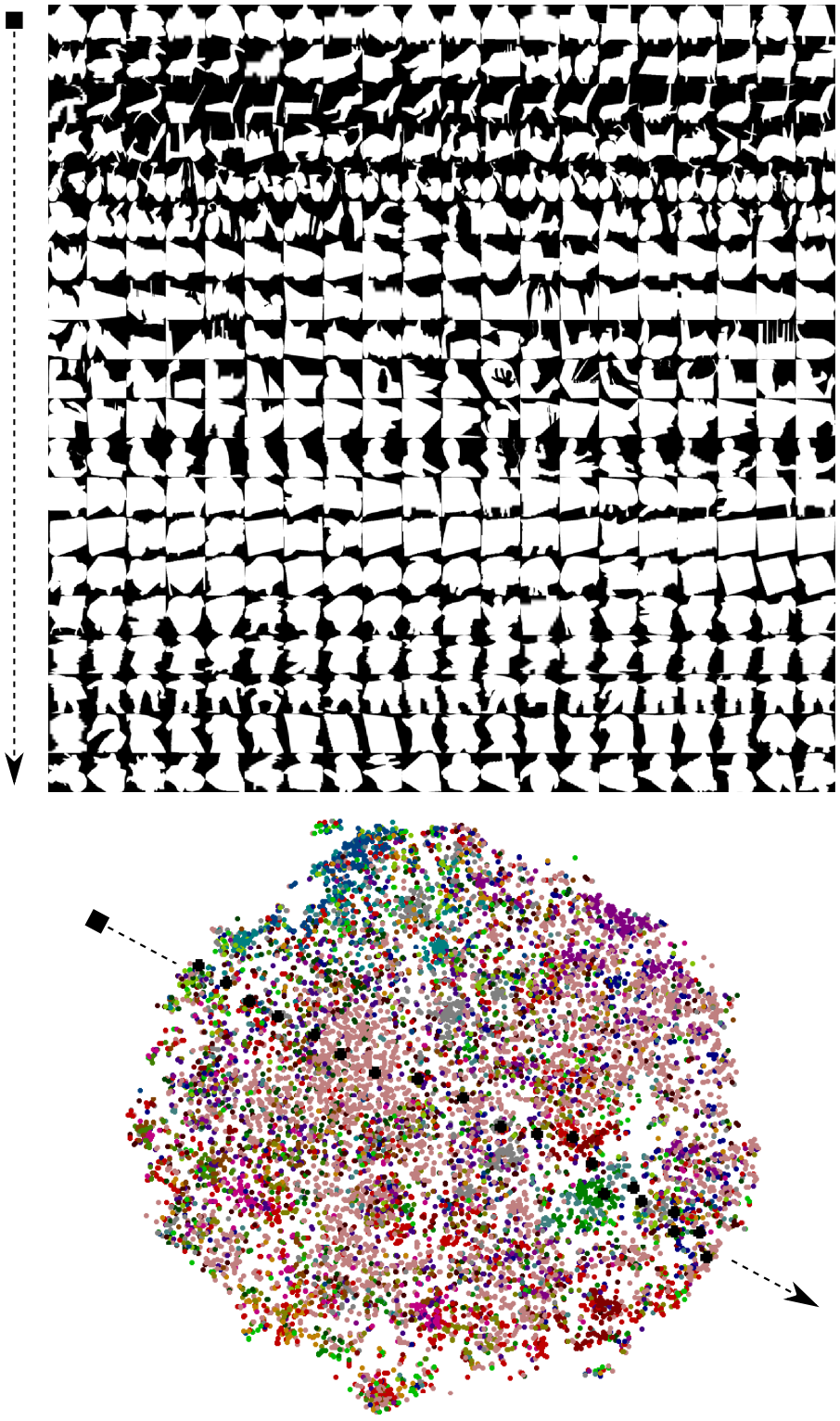


Figure 4.9: Visualisation of the 16×16 binary mask embedding space. (top) Ground truth binary masks. Rows (in top) represent the 20 neighbouring binary masks for each of the 20 anchor points in the embedding space (bottom).

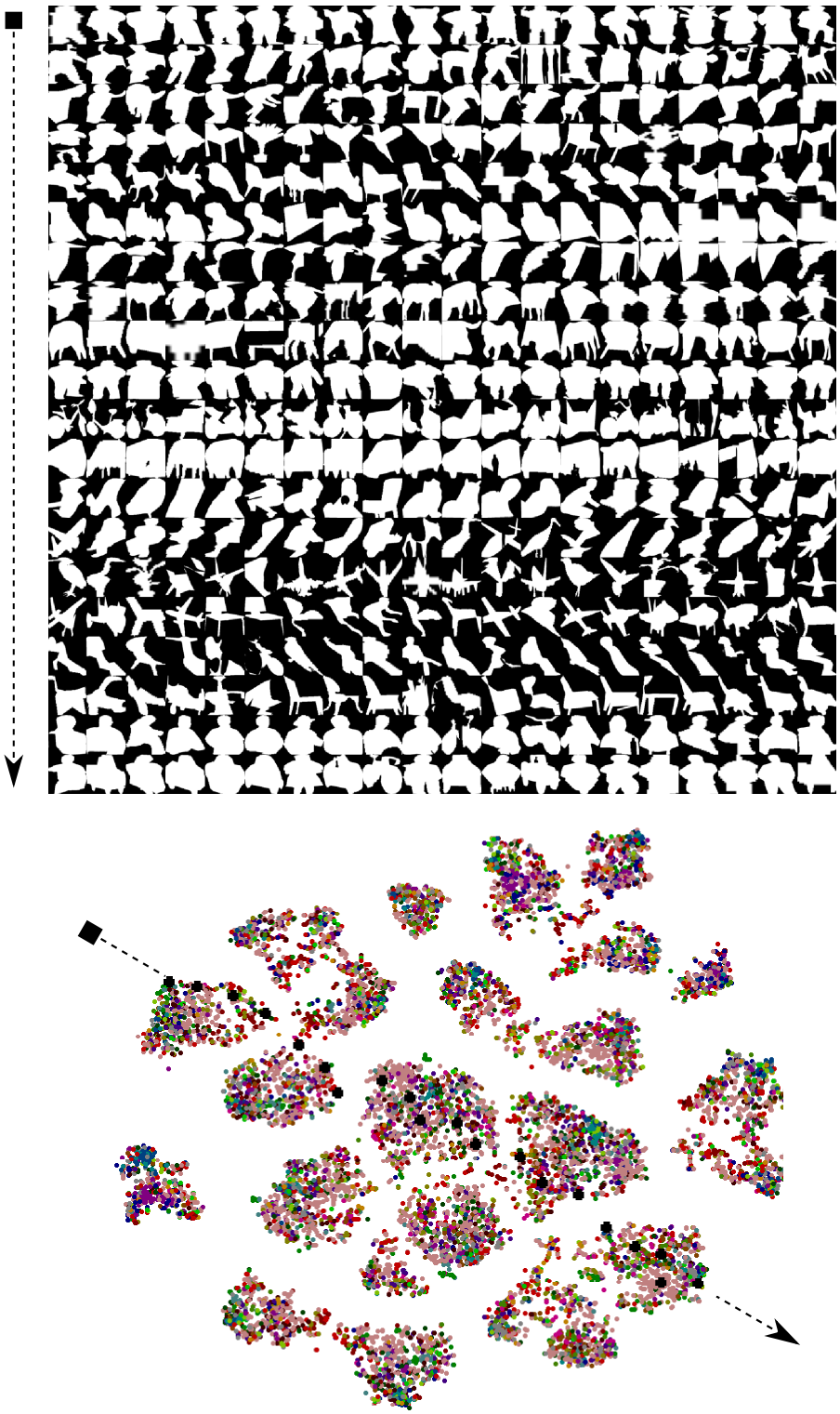


Figure 4.10: Visualisation of the 256D radial vector embedding space. (top) Ground truth binary masks. Rows (in top) represent the 20 neighbouring binary masks for each of the 20 anchor points in the embedding space (bottom).

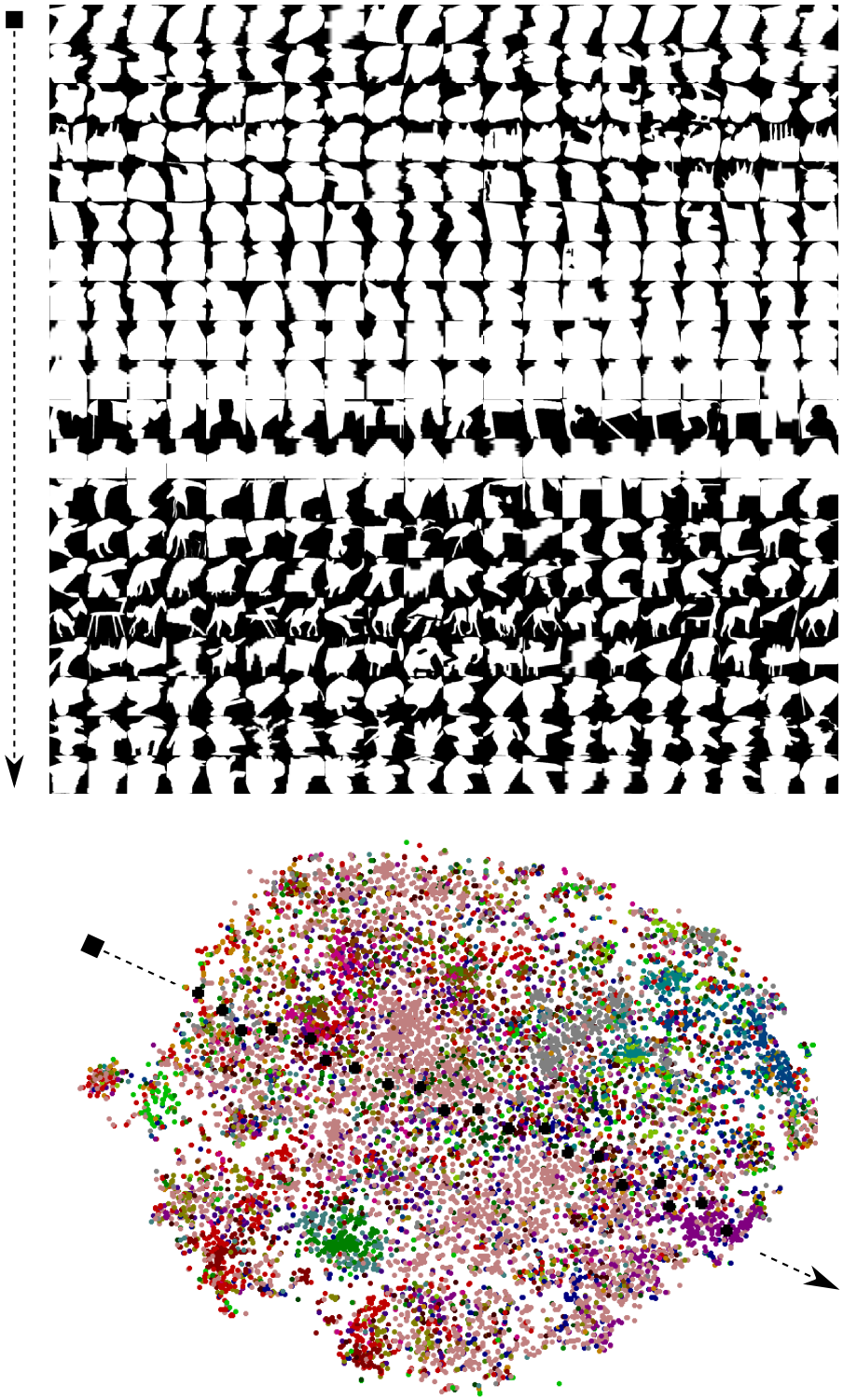


Figure 4.11: Visualisation of the 20D learned embedding space. (top) Ground truth binary masks. Rows (in top) represent the 20 neighbouring binary masks for each of the 20 anchor points in the embedding space (bottom).

	Radial (20D)	**Shape Encoding (20D)**
mAV (azimuth)	1.163	1.008
mAV (elevation)	0.294	0.246
mAV (distance)	21.537	17.968
# (*most-predicted* class == GT class)	7 out of 12	12 out of 12
% votes to *most-predicted* class	0.547	0.800

Table 4.2: Comparing the interpretability of radial representations with that of the learned shape encodings on the PASCAL3D+ dataset. The above scores are estimated over 50 nearest neighbours of a randomly selected validation shape in each respective space, and then averaged over 10 such validation images for each of the 12 rigid object categories.

	SBD (5732 val images)			**Time**
	mAP^r@.5	mAP^r@.7	mAP^r_{vol}	ms
BinaryMask	32.3	12.0	28.6	**26.3**
Radial	30.0	6.5	29.0	27.1
Embedding (50)	32.6	14.8	28.9	30.5
Embedding (20)	**34.6**	**15.0**	**31.5**	28.0
SDS (Hariharan et al., 2014)	49.7	–	41.4	48k
MNC (Dai et al., 2016a)	65.0	46.7	–	330

Table 4.3: Quantitative instance segmentation results on PASCAL VOC (SBD) validation set. The timing results were obtained on a high-end desktop containing a Titan X.

4.4.2 Instance Segmentation as an Application

The hypothesis that we would like to test is that we are able to regress directly to an object-based representation of the scene. For this experiment, we train a YOLO-style network that maps directly from a single image to a collection of object masks and their locations, which can then be compared against those generated by state-of-the-art instance segmentation methods (Hariharan et al., 2014; Arnab & Torr, 2016; Dai et al., 2016a).

Dataset and Performance Metrics: We train and evaluate our method on the established PASCAL VOC dataset with SBD annotations (Everingham et al., 2005; Hariharan et al., 2011), with the same split sets as used by Hariharan et al. (2014); Dai et al. (2016a). We report the mean average precision at various overlap thresholds (mAP^r, mAP^r_{vol}) (Hariharan et al., 2014).

Results and Discussion: In Table 4.3, we report the instance segmentation results we obtain when regressing to the following shape representations: i) downsampled

Figure 4.12: **(a)** Correct predictions using downsampled binary masks. **(b)** Correct predictions using 20D learned shape encodings. In the 3rd image, the horns of the cow are missed and the human shape mask gets elongated due to an incorrect bounding box prediction. **(c)** Missed detections using the 20D shape encodings. The network misses out or false fires on small objects (2nd column). The dogs in the images are falsely categorised as cats, and the sofa incorrectly includes the nearby dining table.

binary shape masks ($d = 256$), ii) radial vectors ($d = 256$), and iii) learned embeddings ($d \in \{20, 50\}$). The dimensionalities for the shape spaces are not based solely on the reconstruction errors of Table 4.1, but also on the reconstructed masks of Figure 4.6. Qualitatively, while a 20D learned encoding is able to retain key semantic details of a given category, it takes 256D binary and radial representations to capture the same level of detail effectively. This reduction in the number of parameters per shape mask for the learned encodings means that we can train neural network models with fewer parameters, and with less chance of overfitting. This intuition is consistent with our empirical findings

in Table 4.3, where we achieve better performance as the dimensionality of the embedding is reduced from 50 to 20. In comparison to the learned embeddings, both downsampled 16×16 binary masks and $256D$ radial descriptors perform worse, despite having more space available in which to represent the shape. One possible reason for this is the fact that a higher output dimensionality increases the number of network parameters and hence the chance of the model overfitting to the training data and failing to generalise to the test distribution. On the contrary, the hand-crafted descriptions are not suitable at low dimensionalities, at which they begin to lose the boundary precision of the object shape masks. In addition, radial descriptors in particular are highly sensitive to noise, as can be seen in Figure 4.7. Qualitative results showing the performance of our method on the SBD dataset can be seen in Figure 4.12, and in the supplementary material.

It can be seen in Table 4.3 that our instance segmentation results somewhat lag behind current state-of-the-art methods; this parallels the lag in object detection results experienced by YOLO with respect to state-of-the-art methods in that field. We hypothesise that this is because both approaches struggle to precisely localise certain objects, especially small ones (see Figure 4.12(c)). However, like YOLO, our approach runs in real time, something that cannot be achieved by any of the *offline* methods against which we compare (as can be seen in Table 4.3, existing methods are in no way real time, taking at least an order of magnitude longer to run). We are also much better than the state-of-the-art instance segmentation method of Arnab & Torr (2016) at segmenting unseen categories (see Figure 4.13), as discussed in the next section.

4.4.3 Zero-shot Segmentation

Traditionally, the task of zero-shot learning has been achieved through the use of attributes (Romera-Paredes et al., 2015; Zhang & Saligrama, 2016). A recent work on zero-shot boundary segmentation (Li et al., 2014) follows a similar approach. By contrast, our method doesn't need an explicit definition of attributes to extend shape segmentation to unseen categories (see Figure 4.13); shape attributes are implicitly captured in the definition of the shape space. We are thus able to scale to object categories that have similar shape appearances at test time. This is very similar to how humans

Figure 4.13: A qualitative comparison between the state-of-the-art **(a)** semantic segmentation (Arnab et al., 2016), **(b)** instance segmentation (Arnab & Torr, 2016), and **(c)** our shape detection results, on images from YouTube videos of animals that are not present in the PASCAL VOC training set. In the first two rows, instance segmentation predicts that the legs of the tiger are human. Our method is more consistent over the tiger images taken from the same video. In the lower rows, the instance segmentation approach **(b)** fails to predict any segments, whilst our method predicts class 'dog' for the tiger, hedgehog, baby elephant and bear instances, and class 'horse' for the large elephant, while getting the instance masks approximately correct.

perform. For instance, someone who has never seen a tiger before would still be able to segment it properly, even though he/she might compare it to a *cat- or dog-like animal*, and this is precisely how our network approaches the task.

In order to evaluate our segmentation of unseen class objects, we take our shape prediction model (50D learned encoding) trained on PASCAL VOC train and val sets, and test it on the 60 object categories in MS-COCO (Lin et al., 2014b) not present in

	COCO (8037 val images)		
	mAP^r@.5 (all)	mAP^r@.5 (large)	AR^r@.5 (large)
Embedding (50)	3.6	7.1	23.2

Table 4.4: Quantitative zero-shot segmentation results obtained by running our shape embedding (50) based prediction model on the 60 categories of MS-COCO not present in PASCAL VOC.

PASCAL VOC. In particular, we filter images from the entire MS-COCO dataset such that the resulting images do not contain any object of a seen category; yielding 8037 images in our test-bed with a total of 24396 instances of unseen object classes. At test time, we gather all object predictions with a detection score greater than $t = 0.05$, since the unseen category scores are likely to be low, and then run them through the evaluation code provided by Lin et al. (2014b), which we modify accordingly to calculate IoU over shapes instead of boxes. The results, shown in Table 4.4, are specially reported for large (in size) objects since the training data i.e. the PASCAL VOC dataset predominantly ($> 80\%$ of the time) contains objects in the large size range as defined for the MS-COCO dataset. Given that the existing methods perform very poorly for segmentation of unseen classes (Figure 4.13(a),(b)), we report our results as a strong baseline. We hope that these results will help catalyse research in this direction.

4.5 Conclusion

In this work, we show for the first time that it is possible to regress directly and simultaneously to multiple object representations that incorporate the notion of shape. We combine ideas from object detection, instance segmentation and low-dimensional embedding spaces to create a real-time system to predict encoded shapes. One key factor that allows us to do this is the introduction of a shape embedding space that is compact, decodable, continuous and interpretable. We find that imbuing object detectors with the knowledge of shape allows us to predict plausible masks for previously-unseen categories, allowing us to detect the presence of objects in images for which current state-of-the-art instance segmentation methods perform poorly or fail.

Our next step is to investigate how our shape prediction approach can be extended to cope with a larger variety of shapes and object categories like those in MS-COCO (Lin et al., 2014b).

4.A Appendix

4.A.1 Training the Regressor

The regression network architecture is based on the YOLO design (Redmon et al., 2016), which has 24 convolutional layers, followed by 2 fully-connected layers; the convolutional layers are pre-trained on the ImageNet dataset (Russakovsky* et al., 2015). The relative weights between the loss components are set to $\lambda_{shape} = 0.1$, $\lambda_{box} = 5$, $\lambda_{obj} = 1$ (for object confidence score), $\lambda_{noobj} = 0.5$ and $\lambda_{pmf} = 1$ (for class probability mass function).

We train the network for 500 epochs on the training set of the PASCAL VOC, SBD split (Hariharan et al., 2014)*, which takes about 3 days on one Titan X. We use a batch size of 64, a momentum of 0.9 and a weight decay of 0.0005. The learning rates for the various batches are set as follows:

Batch Numbers	Learning Rate
0 – 200	0.001
201 – 400	0.0025
401 – 20,000	0.005
20,001 – 30,000	0.0025
30,001 – 40,000	0.00125
40,001+	6.25×10^{-4}

To mitigate the effects of a relatively small dataset ($\sim 5k$ training images), we use data augmentation with parameters generated uniformly within the following ranges: rotation ($\pm 4°$), translation ($\pm$ 20% of image size), scaling ($\pm$ 3% of image size), random flipping, intensity scaling ($\alpha I + \beta$, with α in the range ± 2 and β in the range ± 10).

In the main content, we compare our results qualitatively to the work of Arnab & Torr (2016), who kindly tested their algorithm on the images that we selected from YouTube videos. Note that we do not compare to Arnab & Torr (2016) quantitatively, as the

*https://github.com/bharath272/sds_eccv2014/blob/master/train.txt

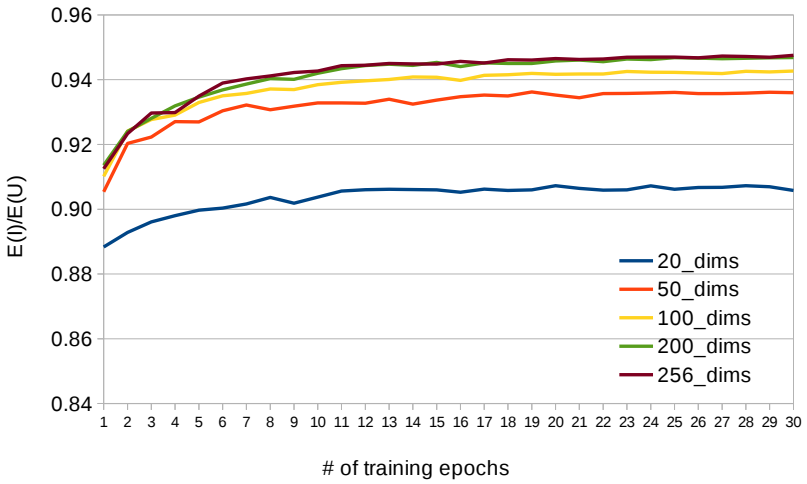


Figure 4.14: Average IoU between input and reconstructed shape masks as a function of the network training. Different curves correspond to different denoising auto-encoder architectures adapted to distinct embedding space sizes. The average IoU is approximated as the ratio of average intersection and average union as measured over a random selection of shape masks.

training and test data used by Arnab & Torr (2016) are different (combining data from the standard PASCAL VOC dataset (Everingham et al., 2005), the SBD dataset (Hariharan et al., 2014), and MS-COCO (Lin et al., 2014b)).

4.A.2 Training the Auto-encoder

We train the auto-encoder architecture, described in §4.3.2 of this chapter, for various dimensionalities of the embedding space, namely 20, 50, 100, 200 and 256. For both training and testing purposes, we crop each image to the boundary of its inset binary mask and resize it to 64×64. We use a batch size of 128 and an initial learning rate of 0.001, and train the networks for a total of 300 epochs, with the learning rate being reduced by a factor of 2 every 60 epochs. The network weights are tuned using back-propagation and stochastic gradient descent (in particular, the Adam optimisation method (Kingma & Ba, 2014)).

We adjust the network weights to minimise the binary cross-entropy loss L_{bce} between the input binary mask p and the reconstructed binary mask $\hat{p}$ as follows:

$$L_{bce} = -\sum_{i=1}^{P} p_i \cdot \log(\hat{p}_i) + (1 - p_i) \cdot \log(1 - \hat{p}_i) \tag{4.5}$$

Another equivalent definition of the loss is provided in Eq. 4.4 of the main part of the chapter. Note that, P is the total number of pixels in the binary mask image.

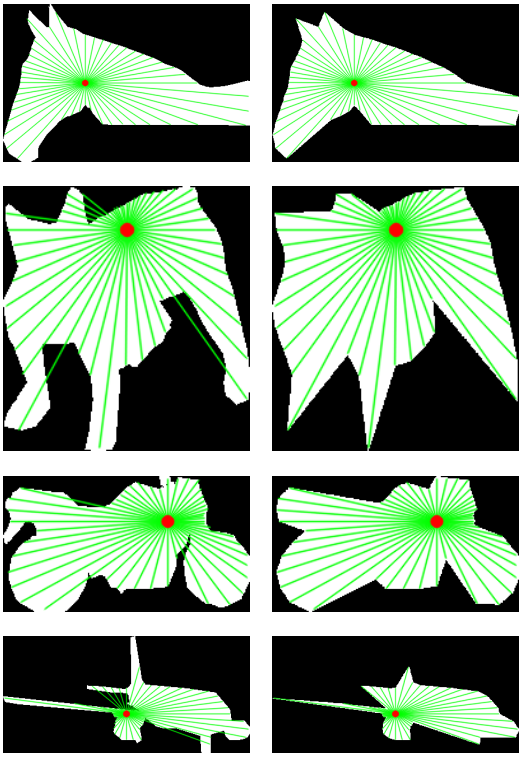


Figure 4.15: Some example radial descriptors of size 50, superimposed on the original masks (left column) and the reconstructed masks (right column). Each chosen reference point is shown with a red dot, and we draw green rays between the reference point and each point on the represented shape contour. By choosing the reference and the way in which we cast rays appropriately in each case, we achieve much better reconstruction results than we could using the standard centre of the mask and casting rays outwards; however, for more complicated shapes such as the dog (second row) and the aeroplane (fourth row), our radial descriptors fail to capture important details such as the shapes of the ears, feet or the tailplane.

The convergence plots for various embedding space sizes are shown in Figure 4.14. It is interesting to note that with the reduction in the size of the embedding space, the performance degrades gracefully and minimally. Noticeably, with a 10 times reduction in the dimensionality, the expected IoU decreases only by around 0.04 points. Note: we measure the expected IoU or average IoU as the ratio of the average intersection and average union as measured over a random selection of shape masks.

4.A.3 Computation of Radial Descriptors

To compute a radial descriptor for a binary shape mask, we adopt the following scheme. Given a fixed descriptor size d, we allocate the first 2 elements of the descriptor to contain the (x, y) coordinates of the reference point we choose within the mask, normalised to

$[0,1] \times [0,1]$ (where $(0,0)$ denotes the top-left of the image containing the mask and $(1,1)$ denotes the bottom-right). We allocate the remaining $d-2$ elements to contain distances between the chosen reference point and the boundary at evenly-spaced angles in $[0, 2\pi)$, each appropriately scaled to fall in the $[0,1]$ range. In particular, element $i+2$ of the descriptor stores the scaled distance between the reference point and the boundary at an angle of $2\pi i/(d-2)$.

In practice, to achieve better IoU scores for reconstruction, we construct several radial descriptors for each shape and pick one with maximal IoU. In particular, we try 25 possible reference points, evenly spaced on a 5×5 grid superimposed over the $w \times h$ mask at locations

$$\left(\frac{jw}{6}, \frac{ih}{6}\right) \ : \ i,j \in [1,5].$$

We also try conceptually casting rays both away from and towards the reference point (in practice, casting rays towards the reference point is implemented by casting rays away from the reference point and only stopping when the outer boundary of the shape is reached). This scheme significantly improves the IoU scores we can achieve for reconstruction with respect to always using the (fixed) centre of the mask and casting rays outwards.

Figure 4.15 shows some of the size 50 radial descriptors we calculate, superimposed on both the original and reconstructed masks so as to illustrate what aspects of shape the descriptors can and cannot capture. The chosen reference in each case is shown with a red dot, and we draw green rays between the centre and each point on the represented shape contour.

4.A.4 Software Implementation

We use the Darknet framework (Redmon, 2013–2016) (written in C) from June 2016 for training our shape prediction model. We developed our own C++ software to deal with dataset loading, manipulation, transformations and preparation for the neural network. We also developed C++ code to evaluate our system on the task of instance segmentation; our code is several times faster than the standard MATLAB version of Hariharan et al.

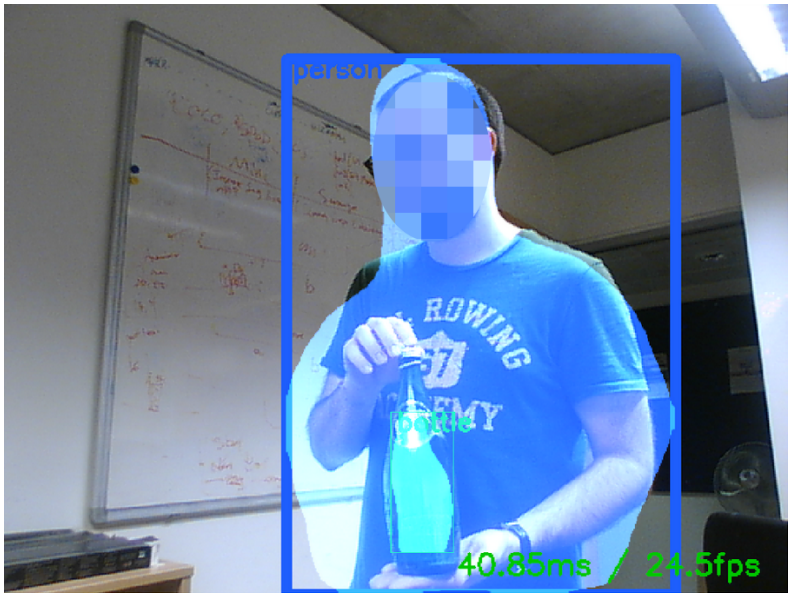


Figure 4.16: Testing direct regression to object shapes in a live webcam stream.

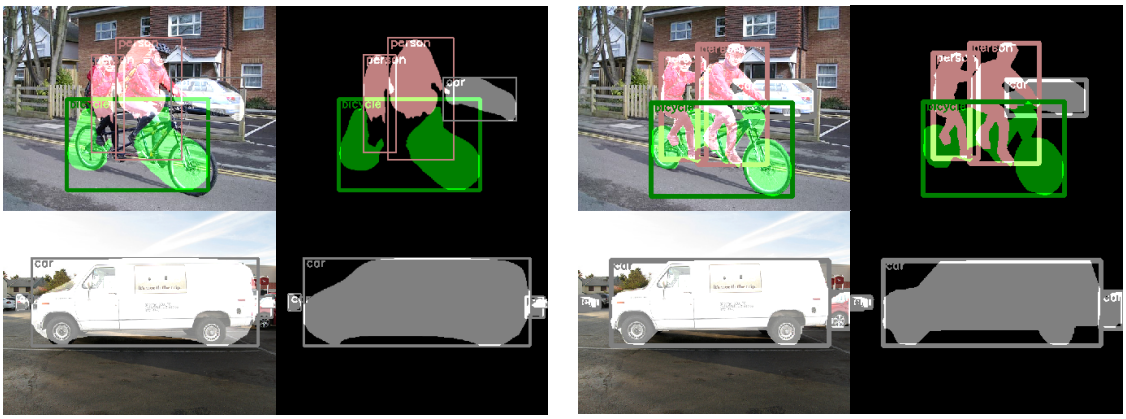
(2014). For ease of development, we use the Torch framework for learning the shape embedding space, and interface the C++ and Lua code using the LuaBridge library*. This interface includes delays that could be avoided by implementing an optimised version of our Lua code in C++.

Real-Time Demo: We also developed some code to demonstrate that our system generalises to scenes captured by a web-camera, as illustrated in Figure 4.16.

Reproducible Results: The code and models used to obtain our results are now available at https://github.com/torrvision/straighttoshapes.

Shape Prediction: In Figures 4.17 and 4.18, we show additional qualitative results of our shape prediction system on the PASCAL VOC (Everingham et al., 2005) (SBD) validation set.

*https://github.com/vinniefalco/LuaBridge



(a) (b)

Figure 4.17: A more detailed look at the results. (a) The results from our shape prediction network and (b) the ground truth. (a) (top) Two people riding a tandem bike. Note that our system predicts shape masks while using bounding boxes for localisation. As such, any error in localisation will affect the location at which the shape mask gets superimposed onto an image (the box does not hug the bike tires tightly). In (b) (top), note the complexity of the shapes that need to be predicted. In the bottom row, multiple cars are detected. The predicted shape contains information on the direction that the car is facing, information that is not available from bounding boxes alone. It is noteworthy that even though the output shape mask is not pixel-accurate, it still contains a lot of information about the object's orientation and shape, and its similarity to other shapes in the shape embedding.

Figure 4.18: Additional qualitative results from the PASCAL VOC (SBD) validation set. Note the huge variety in shapes, even within a single category.

5

Straight to Shapes++: More Accurate Real-time Instance Segmentation

Contents

Instance segmentation is an important problem in computer vision, with applications in autonomous driving, drone navigation and robotic manipulation. However, most existing methods are not real-time, complicating their deployment in time-sensitive contexts. In the last chapter, we proposed a novel real-time instance segmentation approach based on low-dimensional shape embedding spaces. This method, called 'Straight to Shapes'

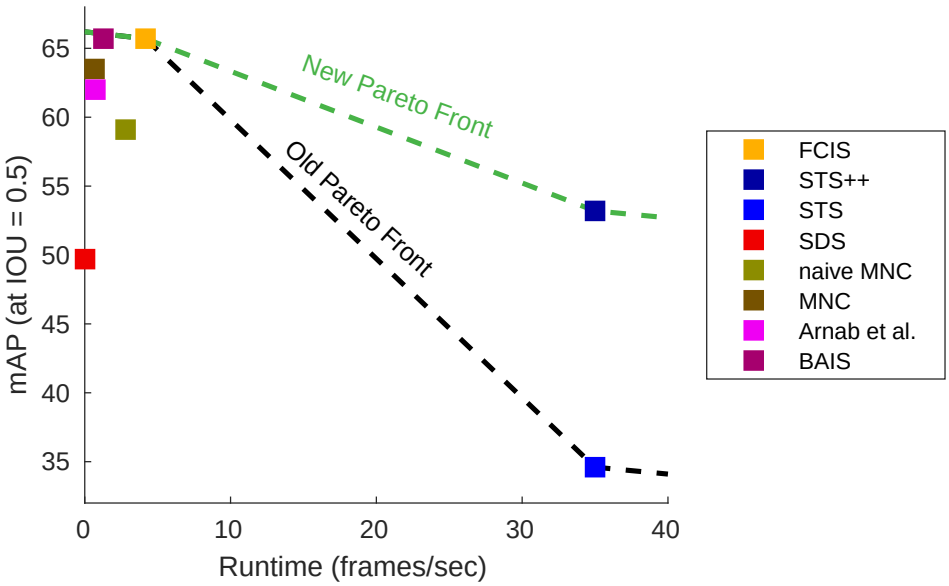


Figure 5.1: mAP vs. runtime trade-off as evaluated on PASCAL VOC dataset (Everingham et al., 2015; Hariharan et al., 2011) across a fleet of existing instance segmentation models: FCIS (Li et al., 2016), STS (Jetley et al., 2017), SDS (Hariharan et al., 2014), naive MNC (Li et al., 2016), MNC (Dai et al., 2016a), Arnab & Torr (2017), BAIS (Hayder et al., 2016). STS++ redefines the performance frontier at real-time speeds.

(STS), can run at 35 FPS on a high-end desktop, but its accuracy is significantly worse than that of offline (non real-time) state-of-the-art methods. In this chapter, we leverage recent advances in the design and training of deep instance segmentation models to improve the performance accuracy of STS model whilst keeping its real-time capabilities intact. In particular, we find that parameter sharing, more aggressive data augmentation and the use of structured loss for shape mask prediction all provide a useful boost to the network performance. Our proposed approach, 'Straight to Shapes++', achieves a remarkable 19.7 point improvement in mAP (at an IOU of 0.5) over the original method as evaluated on the PASCAL VOC dataset, redefining the accuracy frontier for real-time instance segmentation approaches. Since the accuracy of instance segmentation is closely tied to that of the object bounding box prediction, we also study the error profile of the latter and examine the failure modes of the method for future improvements.

5.1 Introduction

Scene understanding deals with the *what* and *where* of objects in a given visual environment. In cases where the scene is described using a set of RGB images, a rather elementary but challenging task of scene understanding is one of identifying and delineating instances of different categories of interest in those images. Popularly known as instance segmentation, this task has a wide applicability in real-time computer vision applications such as autonomous driving (Cordts et al., 2016), drone navigation (Zhu et al., 2018) and robotic manipulation (Schwarz et al., 2018), and the solution methods need to be fast as well as accurate. Towards this, we develop upon a recently proposed instance segmentation model that has demonstrated real-time capabilities but is limited in terms of its performance accuracy (Jetley et al., 2017).

The Straight to Shapes (STS) model extends the real-time object detector YOLO (Redmon et al., 2016) to additionally predict encoded shape representations for multiple object hypotheses. Their model makes use of continuous shape-based representations and learns to map the input image regions to points in this continuous shape space in order to predict instance-level masks for object categories of interest. This use of an intermediate shape embedding space has also been shown to scale to unseen categories at test time that are similar to the training classes, a very useful property when operating in the wild. However, while their simple extension maintains the inference speed, the inaccuracies in regressing to the real-valued vectors of shape representations result in a sub-par mAP accuracy in comparison to existing instance segmentation methods (Li et al., 2016; Hariharan et al., 2014; Li et al., 2016; Dai et al., 2016a; Arnab & Torr, 2017; Hayder et al., 2016). More recently, Liu et al. (2015) (with their SSD model) and Redmon & Farhadi (2016) (with YOLO9000), have demonstrated that state-of-the-art quality can be achieved in real-time detection models by making systematic modifications to the underlying network architecture and training procedure. This suggests that similar improvements may be made in instance segmentation. Thus, we extend the existing STS model through the following contributions:

– We review the recent advances in neural network design and training in the context of object detection and instance segmentation tasks.

– We present a revised model called STS++ with an improved mAP accuracy on PASCAL VOC (Everingham et al., 2015) at real-time speeds, as shown in Figure 5.1.

– We analyse the errors that the proposed model makes in predicting the object bounding boxes, in terms of the taxonomy proposed by Hoiem et al. (2012), and identify avenues for future improvements.

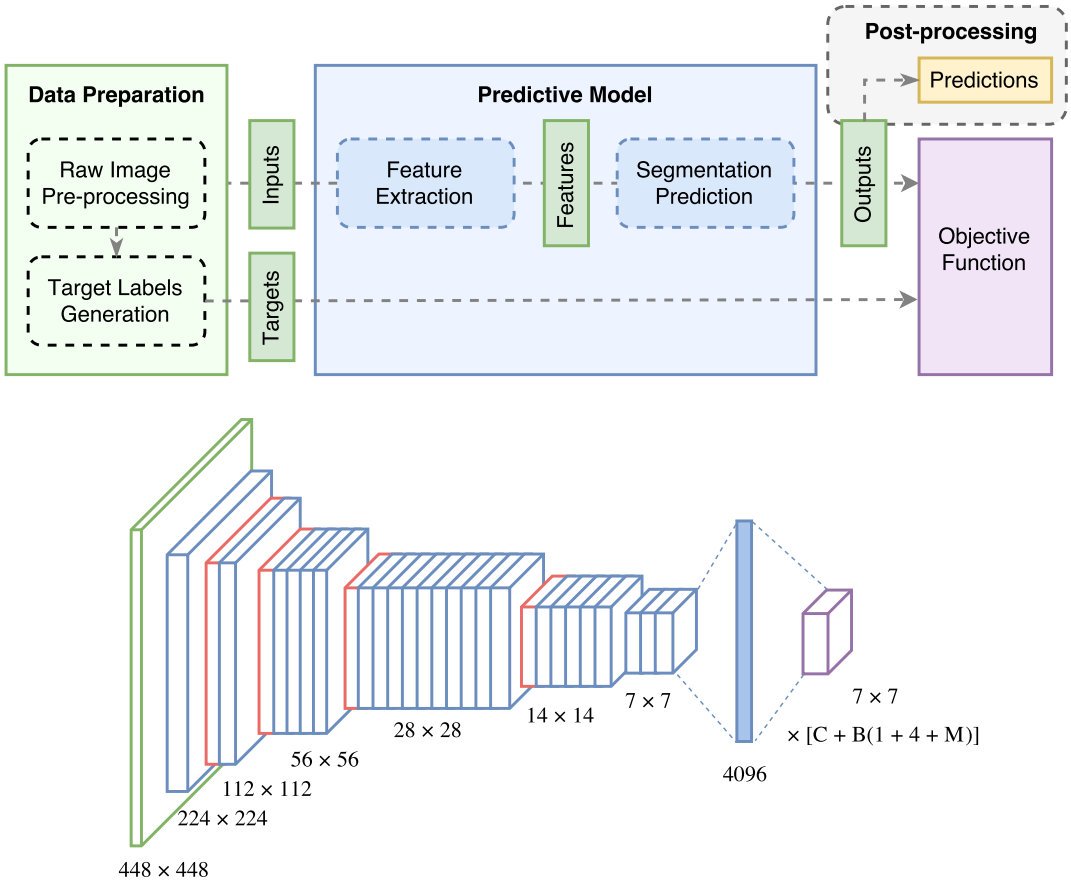


Figure 5.2: Schematic illustration (top) of STS (Jetley et al., 2017) instance segmentation system, and (bottom) its predictive model. The latter is a deep neural regressor that accepts 448×448 sized RGB images as input and encodes them into a fixed 4096-dimensional representation. This representation is then used to predict category, location and shape properties of the underlying object instances. These output properties are matched against ground truth data to train the network via backpropagation. The post-processing step merely takes the predicted shape representations which are then decoded to yield 2D instance shape masks.

5.2 Related Work

Object detection localises individual objects in an image using tight-fitting bounding boxes and classifies them into one of the pre-defined target categories. However, it overlooks the pose and shape details of the objects. Semantic segmentation yields a more fine-grained category map at the pixel-level, but does not contain knowledge about the number of individual objects or the boundaries between them when the objects of the same category are adjoining or overlapping. Instance segmentation attempts to combine the best of both of the above tasks by delineating individual objects at the pixel level. All the above scene understanding tasks have benefited hugely from the availability of large-scale annotated datasets and the advent of deep learning. In particular, there are many deep convolutional neural network based solutions for both object detection e.g. R-CNN (Girshick et al., 2014), Faster R-CNN (Ren et al., 2015), YOLO (Redmon et al., 2016) and SSD (Liu et al., 2015) and semantic segmentation e.g. FCN (Long et al., 2015) and CRF as RNN (Zheng et al., 2015). Following their success, various instance segmentation approaches (Hariharan et al., 2014; Dai et al., 2015, 2016a; Arnab & Torr, 2017; Hayder et al., 2016; Li et al., 2016; He et al., 2017) proposed to combine both top-down detection and bottom-up segmentation results in order to isolate object instances.

For example, the Simultaneous Detection and Segmentation (SDS) approach of Hariharan et al. (2014) adapts the R-CNN (Girshick et al., 2014) detector to the task of instance segmentation. This is achieved by refining the regions within the object boxes of R-CNN via a combination of bottom-up cues from the super-pixels of the Multiscale Combinatorial Grouping (MCG) approach of Arbeláez et al. (2014) and top-down cues from the foreground mask predicted using CNN features. In their approach, the box proposal and mask refinement stages are separately optimised. However, the Multi Network Cascade (MNC) of Dai et al. (2015) incorporates all these steps into a single neural network pipeline using a cascaded structure and joint training. A stack of convolutional layers is shared between the three cascaded tasks of bounding box proposal generation, pixel-wise prediction of an instance mask per proposal using region-of-interest (RoI) CNN features, and classification of the output mask into one

of T target categories. Similarly, the Boundary-Aware Instance Segmentation (BAIS) model (Hayder et al., 2016) uses shared layers for region proposal and RoI feature extraction. This is followed by a pixel-wise prediction of m masks, each corresponding to a specific quantised level of distance transform (DT) values. The DT values directly and redundantly encode the boundary information, which allows the reconstructed instance masks to extend beyond the bounding box and also be robust to prediction noise – two salient features of this approach.

On the other end, Instance-sensitive FCN (Dai et al., 2016a) adapts the segmentation pipeline of FCN (Long et al., 2015) to predict $k \times k$ *relative-position* score maps. Here, each pixel value in a given score map captures the pixel's probability of being at the specified relative position w.r.t to the underlying object. An instance assembly module then reconstructs object masks by using dense sliding windows and copying pixel scores for a given position in an object window (say *top-left*) from the associated relative position (*top-left*) map. The instance mask is thus obtained by combining the masks for each relative location within a specific window. This approach is effective at delineating instances, but doesn't associate those instances with semantic categories: it is hence often used in conjunction with R-FCN (Dai et al., 2016b) to classify the instance proposals. An end-to-end extension of this instance segmentation and classification pipeline is described as the Fully Convolutional Instance-aware Semantic Segmentation (FCIS) model (Li et al., 2016). Unlike Instance-sensitive FCN (Dai et al., 2016a), the position-sensitive score maps here are not predicted at the image level but atop the bounding box predictions of a region proposal network (RPN) (Dai et al., 2016b). The current state-of-the-art model, Mask R-CNN (He et al., 2017), once again extends the RPN pipeline for instance segmentation. The prediction of object bounding boxes and corresponding category scores is followed by the pixel-wise prediction of T binary masks, one for each target category. At test time, the binary mask for the highest-scoring semantic category is selected. The authors remark that the prediction of T class-wise binary masks serves to reduce the competition between pixels to belong to a single category and improves the quality of the output masks. Concurrently, the approach of Arnab & Torr (2017) starts from the semantic segmentation maps of CRF as RNN (Zheng et al., 2015) and

proceeds by allocating each pixel to one of D detections produced by R-FCN (Dai et al., 2016b). They use the CRF framework to optimise this allocation which corresponds to the minimisation of an energy function defined atop the per-pixel semantic score, bounding box score and correlation of the semantic mask with a predefined shape prior.

Two important properties of the above methods are worth noting. Firstly, whilst the accuracy of instance segmentation methods has improved rapidly, runtime has largely remained a secondary concern. The existing best i.e. Mask R-CNN model (He et al., 2017) runs at 5 fps, and cannot trivially be deployed for real-time applications (the cost can be amortised over several frames on a background thread, e.g. Rünz & Agapito (2018), but only if latency is a tolerable issue). Secondly, the bottom-up scheme of pixel-wise instance mask prediction often yields coarse and noisy masks for unseen categories, as noted by Jetley et al. (2017).

In contrast to these methods, STS (Jetley et al., 2017) predicts shape masks via a bottleneck of an encoded, low-dimensional and continuous shape space, which allows test-time generalisation to unseen categories. Unlike all other existing approaches, which cannot run in real time, it is able to run at 35 FPS on a high-end desktop, making it highly desirable, at least in principle, for use in real-time applications (Hicks et al., 2013). However, its practical usefulness is compromised in practice by the accuracy of its predictions, which are significantly worse than those that can be achieved by slower approaches. The goal of this work is to address this problem, inspired by works such as those of Liu et al. (2015) and Redmon & Farhadi (2016), which make systematic modifications to detection networks and achieve improvements in detection accuracy for real-time speeds.

5.3 Methodology

We start by reviewing the original STS model and then discuss the motivation behind the proposed changes and the specifics of those changes.

5.3.1 Original Model

The prediction model of STS (Jetley et al., 2017) is identical to that of YOLO (Redmon et al., 2016). A deep convolutional neural network (comprising 30 layers) encodes an input image of size 448×448 into a fixed 4096 dimensional representation, as shown in Figure 5.2 (bottom). This encoded vector representation is then used for predicting the object instances at all image locations. The STS model predicts object instances relative to the predefined $S \times S$ grid for $S = 7$. At every grid location, the model outputs $C = 20$ conditional class probabilities and makes $B = 2$ class agnostic object instance proposals. Each proposal consists of a confidence score, center coordinates, dimensions of a bounding box, and the additional *instance shape representation* (of size M). More formally, for every grid cell $1 \leq i \leq S^2$, the model outputs a vector

$$\hat{\mathbf{y}}_i = \begin{bmatrix} \hat{\mathbf{p}}_i & \hat{\mathbf{b}}_{i1} & \dots & \hat{\mathbf{b}}_{iB} \end{bmatrix}, \text{ where} \tag{5.1}$$

$$\hat{\mathbf{p}}_i = \begin{bmatrix} \hat{p}_{i1} & \hat{p}_{i2} & \dots & \hat{p}_{iC} \end{bmatrix}, \text{ s.t.} \tag{5.2}$$

$$\hat{p}_{ik} = \mathbb{P}[\text{object of category } k \mid \text{object in cell } i], \text{ and} \tag{5.3}$$

$$\hat{\mathbf{b}}_{ij} = \begin{bmatrix} \hat{c}_{ij} & \hat{x}_{ij} & \hat{y}_{ij} & \hat{\psi}_{ij} & \hat{\omega}_{ij} & \hat{\mathbf{s}}_{ij} \end{bmatrix}, \tag{5.4}$$

are the parameters of the j^th object proposal, where

$$\hat{c}_{ij} = \mathbb{P}[\text{proposal } j \text{ in cell } i \text{ is an object}], \tag{5.5}$$

$[\hat{x}_{ij},\ \hat{y}_{ij},\ \hat{\psi}_{ij},\ \hat{\omega}_{ij}]$ are the bounding box parameters, and $\hat{\mathbf{s}}_{ij} = \begin{bmatrix} \hat{s}_{ij1}, \hat{s}_{ij2}, \dots \hat{s}_{ijM} \end{bmatrix}$ is the M-dimensional shape representation.

Note that here $\psi_{ij},\ \omega_{ij}$ correspond to the square root of the height and width of the object bounding box, respectively. This is done to favour correct predictions of small boxes compared to larger boxes, as a small discrepancy in a large-box prediction has a smaller effect on the output accuracy. Full details of the architecture and loss function design can be found in Table 5.3 of Appendix 5.A.1 and in Appendix 5.A.3 respectively.

The system explicitly constructs a low-dimensional shape embedding space using a denoising auto-encoder. At train time, the deep neural pipeline is adjusted to correctly predict the encoded shape representations $\hat{\mathbf{s}}_{ij}$ corresponding to the underlying scene

objects. At test time, the pipeline estimates these encoded object shape representations which are then mapped to the space of 2D binary shape masks using the standalone decoder block of the denoising auto-encoder. Initial investigation demonstrates that the learned shape space encodes shape information in a semantically meaningful way, where instance masks of objects having similar shape appearances cluster together in the space. It is a continuous space and allows reconstruction of new and realistic shape masks.

5.3.2 Proposed Revisions

We present modifications to the existing system to speed up convergence, reduce memory requirements, minimise computational costs and improve the instance segmentation performance. Following the block diagram of Figure 5.2 (top), we organise the proposed changes into three different groups - changes to the prediction model, the data-preparation step and post-processing operations respectively.

Changes to Prediction Model

Batch Normalisation: Batch normalisation is known to implement regularisation and improve the convergence speed during network training, often resulting in an improved performance at convergence (Ioffe & Szegedy, 2015). Its positive effect on deep neural network training is widely reported in the research literature (He et al., 2016, 2015; Vinyals et al., 2014; Szegedy et al., 2015), and it is demonstrated to improve the original YOLO model in the subsequent work by Redmon & Farhadi (2016). We propose adding batch normalisation after every convolutional layer in the network. Although it increases the total number of learnable parameters in the model, the batch-norm parameters can be combined with convolutional layer's affine transformation once the training is finished. Thus, it does not affect model's computational cost or speed capabilities during inference.

Sharing Prediction Weights: Original model uses a series of convolutional layers followed by a fully-connected layer to construct a representation of the entire image and predict object instances at different grid locations (see Figure 5.2). Firstly, such an architecture is not translation invariant by design, and thus has to be presented with sufficient variety of shifted examples to develop this property. Secondly, fully-connected

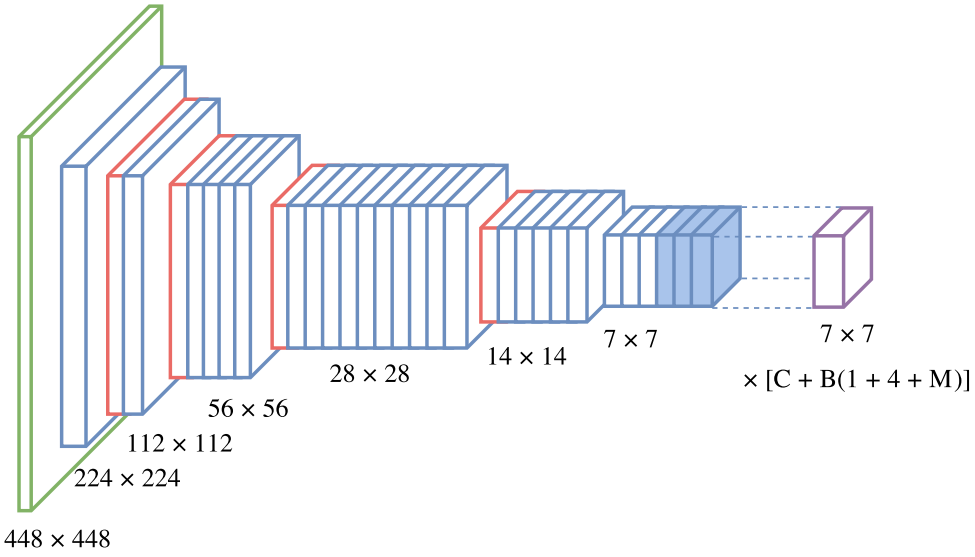


Figure 5.3: Fully connected layers are replaced with convolutional layers (filled blue blocks) to build a translation invariant model.

layers contain the majority of learnable network parameters. Thus, we propose encoding the raw image into a feature map that preserves the spatial information and uses shared convolutional layers to predict object instances at different spatial locations. In order not to reduce model's capacity drastically, we introduce a sequence of 3 convolutional layers ($3 \times 3 \times 2048$, $3 \times 3 \times 2048$ and $1 \times 1 \times 1024$), replacing a single fully-connected layer, before using the shared prediction module (see Figure 5.3 for illustration). Full details of the altered architecture can be found in Table 5.4 in Appendix 5.A.1. Consequently, total number of parameters is reduced from 280×10^6 to 119×10^6 (see Figure 5.4).

Distributing Computations Evenly: Finally, we note that the overall computational cost is rather unevenly distributed across the individual layers of the network, see Figure 5.5. This can slow down the whole network, particularly when the computational resources are limited. In their subsequent work on YOLO, Redmon & Farhadi (2016) identify this issue and propose the Darknet19 architecture. They borrow network design ideas from VGG (Simonyan & Zisserman, 2015) and Network in Network (NiN) (Lin et al., 2013) architectures. For example, the number of filters is doubled when the spatial dimension of the feature map is halved, convolutions of size 1×1 are used at every other layer to compress the feature representations and reduce overall computational

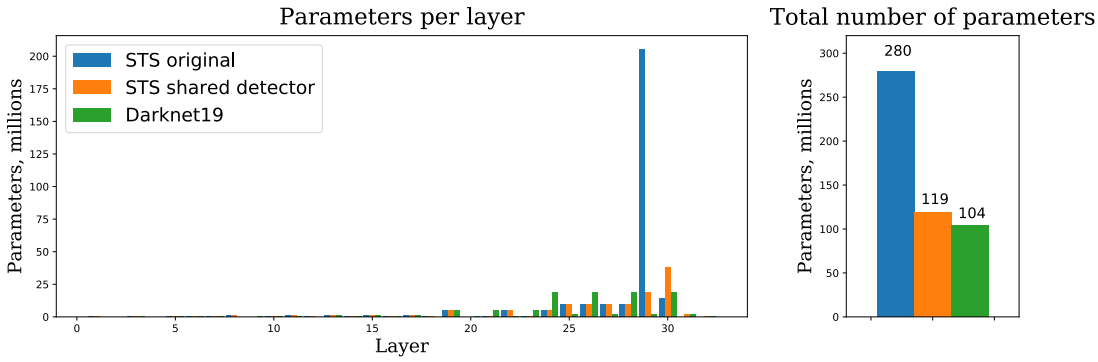


Figure 5.4: Using shared detection module at different locations significantly reduces number of learnable parameters.

costs. We upgrade our model to Darknet19 architecture. Full details of this extended architecture are described in Table 5.5 in Appendix 5.A.1.

Changes to Data Preparation Step

Pre-processing Raw Images: Neural networks used on computer vision tasks usually have their parameter count far exceeding the number of data points available for training (Krizhevsky et al., 2012a). Random data distortions are known to help prevent over-fitting and improve overall performance (Ciresan et al., 2010; Wong et al., 2016). Given also our focus on detecting objects in natural environments, we want to simulate various pose, angle, and lighting conditions via suitable data distortions. This can be achieved, to a limited extent, using affine transformations of 2D images and by performing per-pixel colour distortions. We propose to change the data-augmentation paradigm (often favouring more aggressive transformations) as follows:

- *rotation angle*: uniformly from $[-4°, 4°] \rightarrow [-20°, 20°]$
- *translation in x and y*: uniformly from $[-0.2, 0.2] \rightarrow [-0.15, 0.15]$
- *scaling factor*: uniformly from $[0.97, 1.03] \rightarrow$ log-uniformly from $[\frac{1}{1.2}, 1.2]$
- *random horizontal flip*: from $Bernoulli(0.5)$ (*unchanged*)
- *scale intensity value*: uniform from $[0.8, 1.2] \rightarrow$ log-uniform from $[\frac{1}{1.2}, 1.2]$
- *additive intensity value*: uniform from $[-10, 10]$ (*unchanged*)

Representing Targets: STS (Jetley et al., 2017) uses the target representation approach similar to YOLO (Redmon & Farhadi, 2016) and SSD (Liu et al., 2015). A frame is

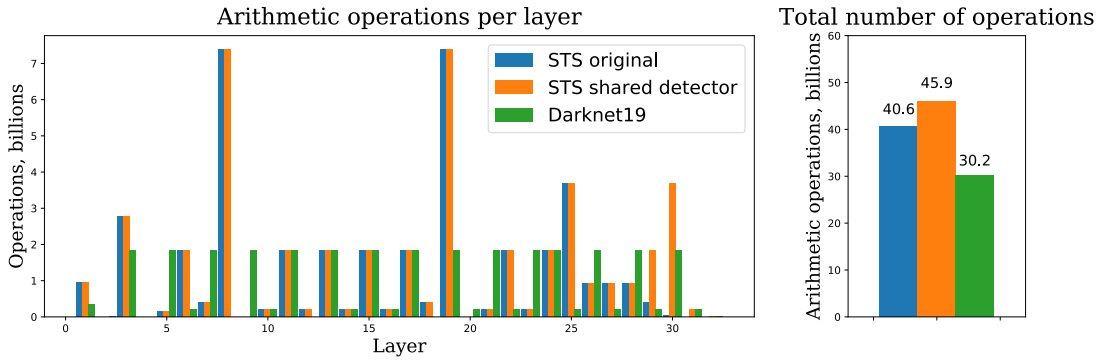


Figure 5.5: Darknet19 network distributes the computations evenly throughout the network.

divided into $S \times S$ grid cells. A cell containing the center of a ground truth bounding box gets assigned all relevant information about the ground truth instance, provided as a vector of size $4 + C + M$, as described in §5.3.1. A nuance of this assignment, however, is that any grid cell is capable of holding information of at most one ground truth object. Whenever another object in the frame has its bounding box centered in the cell i, it is discarded. This leads to the model being penalised for predicting such objects, since they are not represented anywhere in the target.

We propose changing such a target encoding, and employing *anchor boxes* - a technique used in other detection and segmentation methods such as Faster R-CNN (Ren et al., 2015), Mask R-CNN (He et al., 2017), MNC (Dai et al., 2015). We use 3 such boxes per grid cell corresponding to 3 different aspect ratios, namely, 1×1, 1×2 and 2×1. This allows the object instances of non-standard aspect ratios to find their best match, given that we assign an instance to an anchor box when their IOU is the highest amongst other pairs. In addition, the system can now support the prediction of up to 3 closely placed ground truth objects per location.

Representing Shapes: Several different methods to represent shapes are investigated in the STS paper, namely, down-sampled binary mask representation, radial contour description, and an embedding space learned using a parametric model (in particular, a denoising auto-encoder (Vincent et al., 2008)). We use the learned shape embedding in all of our experiments, due to its low dimensionality, better reconstruction accuracy and robustness to noise.

As an alternative, we also propose and investigate the use of the *distance transform* (DT) based representation of binary images (Borgefors, 1986). DT encodes information in a multi-valued format where each pixel value represents its closest distance to the background (w.r.t. some metric d, here we use the Euclidean l_2-distance). This lends DT based representations with a richer structure. Moreover, a corrupted pixel value in DT representation can be recovered given the surrounding pixels values. This property provides a more robust shape reconstruction in the presence of prediction noise.

Changes to Post-processing Step

Mask Decoding: In order to evaluate the quality of proposed instances, binary masks of size $m \times m$ ($m = 64$) are reconstructed from their predicted shape representations and re-scaled to fit their corresponding bounding box predictions. In the case of learned shape embeddings, a trained neural network decoder is used to reconstruct the mask from the predicted embeddings. Detailed architecture of the decoder can be found in the Table 5.6 in Appendix 5.A.1. It is trained separately as part of an auto-encoder pipeline using the per-pixel binary cross-entropy loss:

$$BCE = \sum_{k=1}^{m^2} -t_k \log \hat{t}_k - (1 - t_k) \log (1 - \hat{t}_k), \tag{5.6}$$

where t_k and $\hat{t}_k$ are ground truth and predicted pixel intensities, respectively.

As one of the experiments, we propose incorporating the shape decoder into the full end-to-end trainable pipeline. This simplifies the overall approach and allows the rich semantic structure of the larger PASCAL VOC dataset to tune the reconstruction process. The original decoder has relatively small number of learnable parameters (88 thousand) and uses hard-coded bi-linear spatial up-sampling, which limits the shape decoder's learning ability. Hence, we also propose an alternative decoder architecture with increased learning capacity and learnable up-scaling function (via transposed convolutions). The number of layers and output mask dimensions remains the same, while we increase the number of filters per layer. More details can be found in the Table 5.7 of Appendix 5.A.1. The squared error term in the original loss function design (Eq. 5.11, Appendix 5.A.3),

Figure 5.6: (Left) Object instance binary mask; (Middle) Distance transform based representation of the binary mask, (Right) Illustration of the process of reconstruction of the binary mask from its distance transform based representation. This is done by superimposing discs, at every pixel position, of radii equal to the underlying distance values. (Images taken from `wolfram.com`)

to regress to the encoded shape representations, is replaced with binary cross-entropy (Eq. 5.6) during the holistic training of the full pipeline.

Decoding the Distance Transform (DT): In order to accommodate DT based shape representations, we implement another differentiable decoder for purposes of reconstruction. Given a low dimensional representation for every instance, a neural decoder, designed as a sequence of transposed and ordinary convolutional layers, generates l binary masks of size $m \times m$. Every such mask encodes a specific quantised value representing distance to the background. In particular, the mask $r \in \{1, \ldots, l\}$ has value 1 where the distance is at least r, and 0 elsewhere. Every pixel in every mask is modelled as a binary variable using logistic units. Once such masks are generated a transposed convolutional (deconvolutional) layer is applied with pre-defined non-learnable filters, where every such filter encodes a disk of radius $r \in \{1, \ldots, l\}$. Essentially, a disk of radius r is drawn at the location where encoded distance value is r (see Figure 5.6 for illustration). Noticeably, each pixel value contains information about the distance to the object boundary. This redundancy equips the DT based shape representation to better handle any noise at inference. Final binary mask is obtained by taking a linear combination of such reconstructed masks and thresholding the output, where linear parameters are learned to optimise the objective. The whole decoder is fully differentiable and can be learned together with the full network. For more details refer to the Table 5.8 in Appendix 5.A.1. We use $l = 8$ for our experiments. This is similar to the way that binary masks are reconstructed in the BAIS model by Hayder et al. (2016). Finally, given

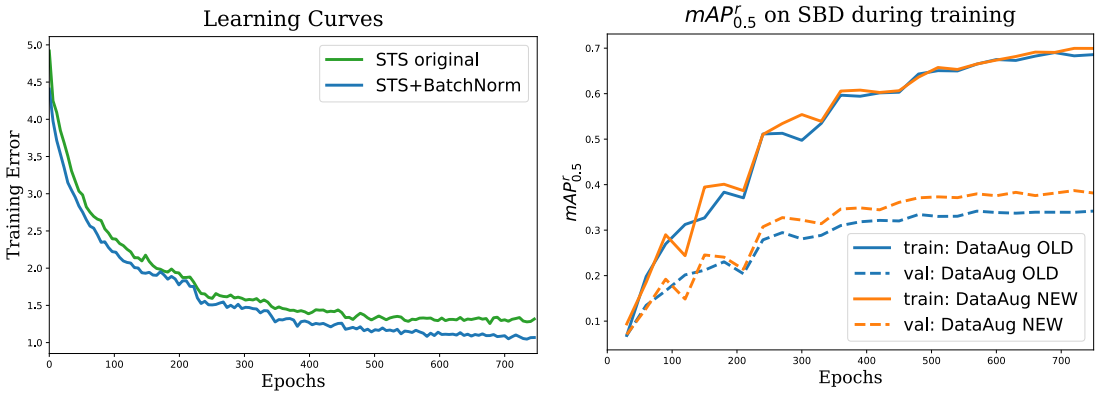


Figure 5.7: (Left) Learning curves during training with and without batch normalisation. (Right) The $mAP^r_{0.5}$ estimates on SBD train and val splits during training with conservative vs. aggressive data augmentation.

the reconstructed binary masks of valid object proposals, non-maximal suppression is performed to filter the overlapping predictions (Neubeck & Gool, 2006).

5.4 Experiments and Results

The details of the experimental setup are provided in Appendix 5.A.2. Here, we present a quantitative analysis of the effects of the proposed changes on the instance segmentation performance. For purposes of comparison, we train our models on the SBD train set and evaluate on the SBD validation set (Hariharan et al., 2011). We further evaluate the models on the task of object detection alone and analyse the results in terms of the error taxonomy proposed by Hoiem et al. (2012).

5.4.1 Evaluating Instance Segmentation Performance

We start with the original STS approach that yields top performance using the 20-dimensional learned shape representations and study the effects of the proposed changes that are introduced incrementally.

Training with Batch Normalisation: As expected and noted in the literature (Ioffe & Szegedy, 2015) batch normalisation improves the convergence speed as well as the performance at convergence, as can be seen in Figure 5.7. On the validation set, it boosts the performance by 5.1% ($mAP^r_{0.5}$). As mentioned before, the processing speed at the time of inference is not affected.

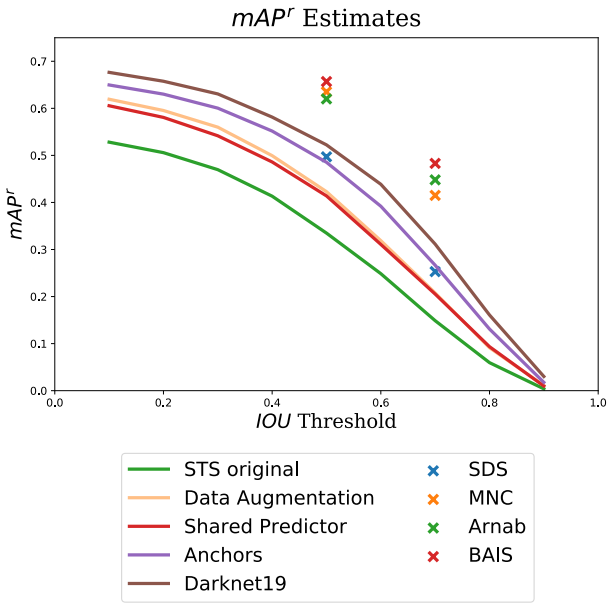
Augmenting Training Dataset: Presenting the network with more aggressively augmented examples improves the model's generalisation. As seen in Figure 5.7, the model has sufficient capacity to fit the increased variability in the training examples, and at the same time it helps to generalise better on unseen validation samples. Building on the previous change of batch normalisation, data augmentation improves our model by an additional 3.7% $mAP^r_{0.5}$.

Sharing Prediction Weights: This change in architecture, from fully-connected to convolutional, introduces translation invariance into the model and reduces the total number of parameters. This has a small negative effect on accuracy decreasing it by 0.9% $mAP^r_{0.5}$, see Figure 5.8 (top).

Anchors as Bounding-box Priors: This impacts how adjacent or possibly overlapping object instances are presented to the model. In the previous setup, the model was discouraged from predicting objects appearing close to each other in the image (by selecting only the ground truth bounding box with the highest IoU when multiple such boxes were centered in the same grid cell). In comparison, our approach provides three anchor boxes per grid cell location with which to match the ground truth instance hypotheses. This leads to a significant 7.1% improvement in $mAP^r_{0.5}$ as can be seen in the performance plots of Figure 5.8.

Distributing Computations Evenly: These architectural changes not only reduce the parameter count and the total computational effort, but also result in a more robust model, offering an overall performance gain of 3.7% in terms of $mAP^r_{0.5}$, refer to Figure 5.8 (top).

Training End-to-End: Training the shape decoder in a unified end-to-end way simplifies the overall optimisation procedure. Moreover, learning the shape decoder in an independent optimisation step can lead to a solution that is sub-optimal overall. Thus, we first train the decoder from a random initialisation using the original architecture discussed in STS (Jetley et al., 2017). This leads to a degradation in the performance, especially, in the high IoU range (see Figure 5.8). This suggests that the model has difficulty in reproducing fine details of the instance shape masks. We then train the decoder with an increased learning capacity (i.e. an increased number of parameters), and call it the Large Decoder model. This model is able to recover from the drop in



	mAP^r_{iou}									mAP^r_{vol}
IoU threshold	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	
STS*	52.8	50.6	47.0	41.3	33.5	24.9	14.9	6.0	0.4	30.1
Darknet19	67.6	65.8	63.1	58.1	52.2	43.9	31.1	16.1	3.0	44.5
End-to-End	68.2	66.2	63.4	59.0	51.7	41.7	27.9	12.8	1.9	43.6
Large Decoder	67.8	65.9	63.1	58.7	52.3	43.8	31.9	**17.3**	**3.9**	45.0
Distance Transform (STS++)	**68.2**	**66.3**	**63.6**	**59.4**	**53.2**	**45.2**	**32.3**	16.9	3.4	45.4

Figure 5.8: mAP^r estimates on SBD validation set for the proposed models incorporating changes to the network architecture and target representations. (Top) mAP^r scores for models with weight sharing, anchor boxes and Darknet19 are compared here with those for SDS (Hariharan et al., 2014), MNC (Dai et al., 2016a), Arnab & Torr (2017) and BAIS (Hayder et al., 2016). (Bottom) Distance transform based shape reconstruction offers a marginal advantage over both pre-trained and end-to-end trained decoders for reconstructing learned shape representations. Note: The model marked with a * has been retrained for this work.

prediction quality noted at high IoU values, and surpass the Darknet19 model with a pre-trained decoder by a small margin.

Distance Transform based Shape Encoding: The alternative shape representation making use of quantised distance transform values offers a slight improvement (up to 1% in terms of mAP^r accuracy) for low IoU values (~ 0.5), as can be seen in Figure 5.8). However, the model still fails to predict the fine details of the object shape masks near the object boundaries, a flaw that manifests itself as low mAP^r scores at high IoU values.

Table 5.1 traces the incremental growth in performance, over the full range of IoU values, for the above discussed series of changes to the network architecture. For a qualitative demonstration of the above results, refer to Figure 5.9.

	STS (Jetley et al., 2017) (darknet)								STS++	SDS (Hariharan et al., 2014) (alexnet)	naive MNC (Li et al., 2016) (resnet-101)	MNC (Dai et al., 2016a) (vgg-16)	Arnab & Torr (2017)	BAIS (Hayder et al., 2016)	FCIS (Li et al., 2016) (resnet-101)
Learned Embedding (20)	+	+	+	+	+	+			+						
Batch Normalisation		+	+	+	+	+	+	+	+						
Data Augmentation			+	+	+	+	+	+	+						
Shared Predictor				+	+	+	+	+	+						
Anchor Boxes					+	+	+	+	+						
Darknet19						+	+	+	+						
End-to-End							+	+	+						
Large Decoder								+	+						
Distance Transform (STS++)									+						
$mAP^r_{0.5}$	34.6	38.6	42.3	41.4	48.5	52.2	51.7	52.3	**53.2**	49.7	59.1	63.5	62.0	65.7	**65.7**
$mAP^r_{0.7}$	15.0	17.4	20.8	20.5	26.7	31.1	27.9	31.9	**32.3**	25.3	36.0	41.5	44.8	48.3	**52.1**
mAP^r_{vol}	31.5	34.3	37.0	36.1	41.4	44.5	43.6	45.0	**45.4**	41.4	-	-	55.4	-	-
runtime/frame (s)	0.028	-	-	-	-	-	-	-	-	48	0.36	1.5	1.37	0.78	**0.24**
frames/sec	~ 35	-	-	-	-	-	-	-	$\sim$ **35**	0.02	2.8	0.67	0.73	1.28	**4.17**

Table 5.1: Effects of the proposed changes on the instance segmentation performance measured in terms of mAP^r on SBD val (Hariharan et al., 2011). The revised STS++ is compared against existing instance segmentation methods in terms of performance accuracy and the processing speed (measured as both runtime/frame in sec. and frames/sec).

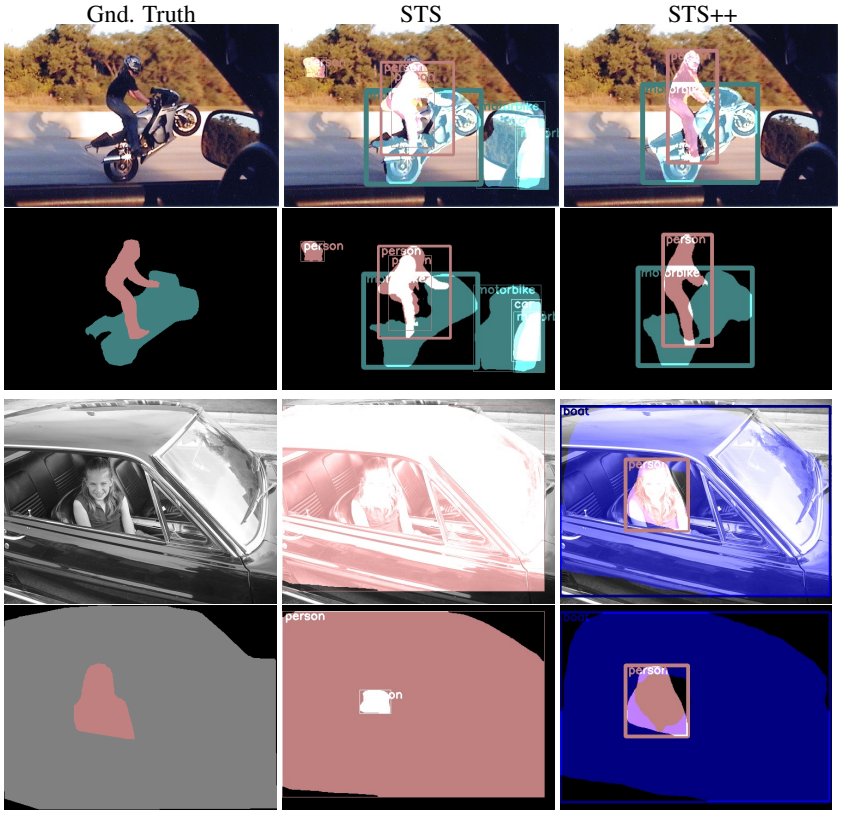


Figure 5.9: Qualitative instance segmentation results: STS (column 2) predicts objects around the rear-view mirror (row 2) and misses the person in the car (row 4). Our proposed model (column 3) provides a more detailed prediction of shape masks for the motorcycle and the rider (row 1 and 2) and correctly delineates the person sitting inside the car (row 3 and 4).

Approach	$mAP^r_{0.5}$	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
STS original	33.5	58.6	33.7	31.2	17.6	11.0	65.5	37.0	62.6	6.2	26.1
Darknet19	52.2	72.0	52.7	54.6	30.6	**28.0**	74.8	**56.8**	81.8	21.1	**56.9**
Large Decoder	52.3	**73.0**	52.5	55.1	**34.8**	24.9	**75.3**	55.3	**82.0**	20.7	54.6
Distance Transform (STS++)	**53.2**	72.9	**53.6**	**58.5**	32.4	25.8	74.9	56.5	81.8	**22.8**	55.0
SDS (Hariharan et al., 2014)	49.7	68.4	49.4	52.1	32.8	33.0	67.8	53.6	73.9	19.9	43.7
(Arnab & Torr, 2017)	62.0	80.3	52.8	68.5	47.4	39.5	79.1	61.5	87.0	28.1	68.3
		dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor
STS original	33.5	13.5	49.7	31.0	37.9	37.7	7.0	31.2	20.0	62.7	29.2
Darknet19	52.2	26.0	71.4	61.6	**62.4**	54.4	19.4	53.2	36.2	76.1	54.5
Large Decoder	52.3	25.2	71.9	**64.8**	58.6	54.0	19.6	54.9	**36.5**	76.9	55.7
Distance Transform (STS++)	**53.2**	**26.2**	**74.1**	62.5	59.8	**57.7**	**22.7**	**56.1**	36.4	**78.5**	**56.4**
SDS (Hariharan et al., 2014)	49.7	25.7	60.6	55.9	58.9	56.7	28.5	55.6	32.1	64.7	60.0
Arnab & Torr (2017)	62.0	35.5	86.1	73.9	66.1	63.8	32.9	65.3	50.4	81.4	71.4

Table 5.2: $AP^r_{0.5}$ scores for individual PASCAL VOC categories on SBD val set.

5.4.2 Evaluating Object Detection Performance

During training, the target masks of object instances are always generated with respect to the ground truth bounding boxes rather than the predicted ones. This prevents the shape predictions from adjusting themselves to the errors in the prediction of the bounding boxes at the time of training. Thus, the accuracy of the overall instance segmentation systems rests on the quality of the object detections. We therefore proceed to analyse how the quality of the detector unit alone evolves as a function of the proposed changes. We also make a quantitative comparison with the STS model proposed by Jetley et al. (2017). We perform the evaluation on PASCAL VOC 2007 test set (Everingham et al., 2015) (which is disjoint from our training set) and use the error analysis methodology and toolkit* developed by Hoiem et al. (2012).

Figure 5.10 defines the taxonomy of prediction errors for a detector unit and compares these errors over the different modifications that are studied in this work. As noted before, the most notable improvement comes from using textitanchor boxes instead of simply the grid cells for target representation during network training. This has the biggest impact on the quality of object localisation. Despite the improvement, the localisation error still remains a main contributing factor, making up more than 9% of the total mAP error. For details of the detection performance on distinct object categories see Tables 5.9 through 5.14 in Appendix 5.A.5. Notice also that the number of background errors is strongly inversely correlated to the localisation error (Figure 5.10). This implies that as the model becomes more precise in localising objects, it is also more prone to detecting

*MATLAB code available at `http://dhoiem.web.engr.illinois.edu/projects/detectionAnalysis/`

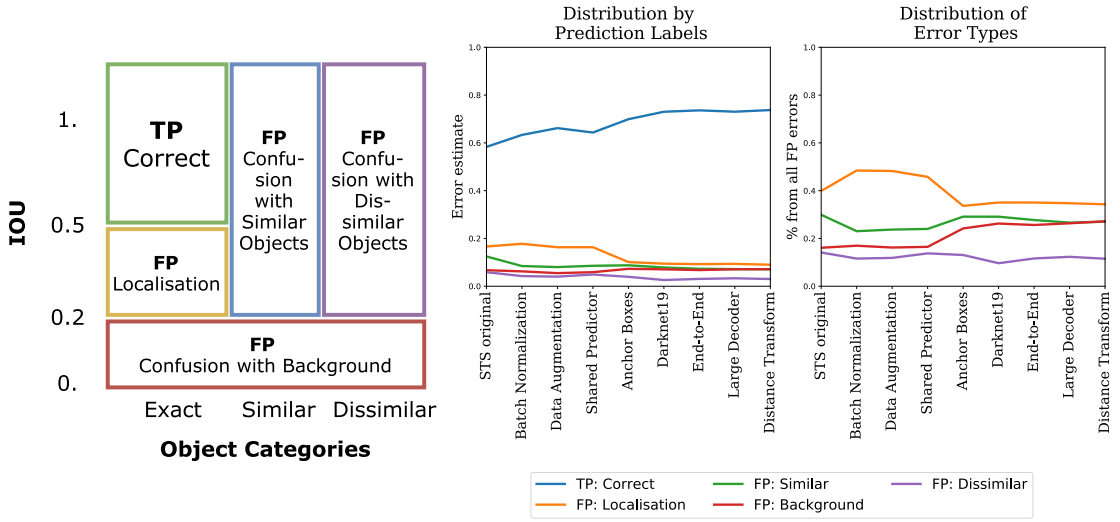


Figure 5.10: (Left) Five types of object detections on PASCAL VOC 2007 test as per Hoiem et al. (2012); (Middle) Typewise break-up of the total mAP error for different architectural modifications; (Right) $\%$-age contribution of each detection type to the total number of false positives.

background regions as likely object instances. Anecdotally, it appears that the model is detecting true objects in the input images which are not actually annotated in the dataset, see Figure 5.12 in Appendix 5.A.4. This is, however, a drawback of the evaluation dataset rather than that of the prediction model.

5.5 Discussion and Conclusion

In this work we tackle the problem of real-time instance segmentation. As shown in Figure 5.1, our revised STS++ model sets a new performance benchmark at real-time processing rates for the task of multiple object instance segmentation. The changes we implement improve the overall model performance by almost 20% in terms of $mAP^r_{0.5}$ and reduce the total number of parameters by 60% via the reuse of network parameters over different spatial locations. Tables 5.1 & 5.2 summarise the effects of the various atomic changes made to the pipeline measured in terms of the mean and individual AP^r scores over 20 distinct object categories of PASCAL VOCdataset. The instance mask visualisations in Appendix 5.A.4 demonstrate that the new model is better at

localising instances and inferring the general body shapes. For more details of the error measurements, refer to Tables 5.13 and 5.14 in Appendix 5.A.5.

Significant improvements have been made to the real-time instance segmentation solution proposed by Jetley et al. (2017) making it more attractive for practical use. Yet there remains an accuracy gap in comparison to the state of the art. We believe that the two main challenges in building more accurate models are as follows. Firstly, models demonstrate poor quality in terms of mAP_t^r for low IoU values (i.e. $IoU \in [0.1, 0.4]$) which indicates issues with correctly localising instance-level object bounding boxes. The analysis of detection errors on PASCAL VOC dataset confirms erroneous localisation as the biggest contributor to the total number of false positives. Moreover, under the current model, instance mask prediction is decoupled from bounding box estimation and has no mechanism to adjust or recover if there is an error in the prediction of the bounding box. Increasing the capacity to represent ground truth instances with different sizes and aspect ratios was demonstrated to yield a great improvement in the results. A further enhancement in the encoding of ground truth information must be sought. Secondly, a poor performance in the high IoU range ($IoU \in [0.7, 1]$) indicates the model's inability to capture intricate details of objects' boundaries. This can be addressed by training pixel-level bottom-up segmentation models such as those making use of conditional random fields (CRFs). The instance segmentation masks obtained from the proposed model can be used as priors for an additional CRF based boundary refinement stage. A similar approach was taken by Arnab & Torr (2016), however, bounding box priors were used instead of segmentation masks. Taking more precise priors in the form of instance masks could further benefit the CRF based post-processing operation.

5.A Appendix

5.A.1 Architectures of Models

Tables 5.3 through 5.8, describe the different neural network architectures used in this project. Every feed-forward network is defined as a sequence of layers. Each layer is described by providing its type, number of output filters (feature maps), filter (kernel) size

dimensions, spatial stride parameter, spatial dimensions of the output layer, number of arithmetic operations performed and the total number of learnable parameters. The layer types include convolutional layer (CONV), transposed convolutional layer (TCONV), max-pooling layer (MAXPOOL) and spatial up-sampling layer (UPSAMPLE). All the convolutional layers are padded with zeros in order to maintain the spatial dimensions of the output feature maps. A single step of addition, multiplication or max comparison is considered to be a single arithmetic operation.

	Type	Filters	Size	Stride	Output	Ops, $\times 10^6$	Params, $\times 10^6$
1:	CONV	64	7×7	2	$224 \times 224 \times 64$	944	0.01
2:	MAXPOOL	1	2×2	2	$112 \times 112 \times 64$	0	0.00
3:	CONV	192	3×3	1	$112 \times 112 \times 192$	2,775	0.11
4:	MAXPOOL	1	2×2	2	$56 \times 56 \times 192$	0	0.00
5:	CONV	128	1×1	1	$56 \times 56 \times 128$	154	0.02
6:	CONV	256	3×3	1	$56 \times 56 \times 256$	1,850	0.30
7:	CONV	256	1×1	1	$56 \times 56 \times 256$	411	0.07
8:	CONV	512	3×3	1	$56 \times 56 \times 512$	7,399	1.18
9:	MAXPOOL	1	2×2	2	$28 \times 28 \times 512$	0	0.00
10:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
11:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
12:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
13:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
14:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
15:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
16:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
17:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
18:	CONV	512	1×1	1	$28 \times 28 \times 512$	411	0.26
19:	CONV	1024	3×3	1	$28 \times 28 \times 1024$	7,399	4.72
20:	MAXPOOL	1	2×2	2	$14 \times 14 \times 1024$	0	0.00
21:	CONV	512	1×1	1	$14 \times 14 \times 512$	206	0.52
22:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
23:	CONV	512	1×1	1	$14 \times 14 \times 512$	206	0.52
24:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
25:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	3,699	9.44
26:	CONV	1024	3×3	2	$7 \times 7 \times 1024$	925	9.44
27:	CONV	1024	3×3	1	$7 \times 7 \times 1024$	925	9.44
28:	CONV	1024	3×3	1	$7 \times 7 \times 1024$	925	9.44
29:	CONV	4096	7×7	7	$1 \times 1 \times 4096$	411	205.52
30:	CONV	3430	1×1	1	$1 \times 1 \times 3430$	28	14.05
					Total:	**40,586**	**279.7**

Table 5.3: Details of the architecture of the original STS model. NOTE: Number of parameters and operations required are given in **millions** (10^6).

	Type	Filters	Size	Stride	Output	Ops, $\times 10^6$	Params, $\times 10^6$
1:	CONV	64	7×7	2	$224 \times 224 \times 64$	944	0.01
2:	MAXPOOL	1	2×2	2	$112 \times 112 \times 64$	0	0.00
3:	CONV	192	3×3	1	$112 \times 112 \times 192$	2,775	0.11
4:	MAXPOOL	1	2×2	2	$56 \times 56 \times 192$	0	0.00
5:	CONV	128	1×1	1	$56 \times 56 \times 128$	154	0.02
6:	CONV	256	3×3	1	$56 \times 56 \times 256$	1,850	0.30
7:	CONV	256	1×1	1	$56 \times 56 \times 256$	411	0.07
8:	CONV	512	3×3	1	$56 \times 56 \times 512$	7,399	1.18
9:	MAXPOOL	1	2×2	2	$28 \times 28 \times 512$	0	0.00
10:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
11:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
12:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
13:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
14:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
15:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
16:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
17:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
18:	CONV	512	1×1	1	$28 \times 28 \times 512$	411	0.26
19:	CONV	1024	3×3	1	$28 \times 28 \times 1024$	7,399	4.72
20:	MAXPOOL	1	2×2	2	$14 \times 14 \times 1024$	0	0.00
21:	CONV	512	1×1	1	$14 \times 14 \times 512$	206	0.52
22:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
23:	CONV	512	1×1	1	$14 \times 14 \times 512$	206	0.52
24:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
25:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	3,699	9.44
26:	CONV	1024	3×3	2	$7 \times 7 \times 1024$	925	9.44
27:	CONV	1024	3×3	1	$7 \times 7 \times 1024$	925	9.44
28:	CONV	1024	3×3	1	$7 \times 7 \times 1024$	925	9.44
29:	CONV	2048	3×3	1	$7 \times 7 \times 2048$	1,850	18.88
30:	CONV	2048	3×3	1	$7 \times 7 \times 2048$	3,699	37.75
31:	CONV	1024	1×1	1	$7 \times 7 \times 1024$	206	2.10
32:	CONV	135	1×1	1	$7 \times 7 \times 135$	14	0.14
					Total:	**45,915**	**119.0**

Table 5.4: Details of the architecture of the STS model with a shared detection layer. NOTE: Number of parameters and operations required are given in **millions** (10^6).

	Type	Filters	Size	Stride	Output	Ops, $\times 10^6$	Params, $\times 10^6$
1:	CONV	32	3×3	1	$448 \times 448 \times 32$	347	0.00
2:	MAXPOOL	1	2×2	2	$224 \times 224 \times 32$	0	0.00
3:	CONV	64	3×3	1	$224 \times 224 \times 64$	1,850	0.02
4:	MAXPOOL	1	2×2	2	$112 \times 112 \times 64$	0	0.00
5:	CONV	128	3×3	1	$112 \times 112 \times 128$	1,850	0.07
6:	CONV	64	1×1	1	$112 \times 112 \times 64$	206	0.01
7:	CONV	128	3×3	1	$112 \times 112 \times 128$	1,850	0.07
8:	MAXPOOL	1	2×2	2	$56 \times 56 \times 128$	0	0.00
9:	CONV	256	3×3	1	$56 \times 56 \times 256$	1,850	0.30
10:	CONV	128	1×1	1	$56 \times 56 \times 128$	206	0.03
11:	CONV	256	3×3	1	$56 \times 56 \times 256$	1,850	0.30
12:	MAXPOOL	1	2×2	2	$28 \times 28 \times 256$	0	0.00
13:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
14:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
15:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
16:	CONV	256	1×1	1	$28 \times 28 \times 256$	206	0.13
17:	CONV	512	3×3	1	$28 \times 28 \times 512$	1,850	1.18
18:	MAXPOOL	1	2×2	2	$14 \times 14 \times 512$	0	0.00
19:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
20:	CONV	512	1×1	1	$14 \times 14 \times 512$	206	0.52
21:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
22:	CONV	512	1×1	1	$14 \times 14 \times 512$	206	0.52
23:	CONV	1024	3×3	1	$14 \times 14 \times 1024$	1,850	4.72
24:	CONV	2048	3×3	2	$7 \times 7 \times 2048$	1,850	18.88
25:	CONV	1024	1×1	1	$7 \times 7 \times 1024$	206	2.10
26:	CONV	2048	3×3	1	$7 \times 7 \times 2048$	1,850	18.88
27:	CONV	1024	1×1	1	$7 \times 7 \times 1024$	206	2.10
28:	CONV	2048	3×3	1	$7 \times 7 \times 2048$	1,850	18.88
29:	CONV	1024	1×1	1	$7 \times 7 \times 1024$	206	2.10
30:	CONV	2048	3×3	1	$7 \times 7 \times 2048$	1,850	18.88
31:	CONV	1024	1×1	1	$7 \times 7 \times 1024$	206	2.10
32:	CONV	70	1×1	1	$7 \times 7 \times 70$	7	0.07
					Total:	**30,155**	**103.8**

Table 5.5: Details of the architecture of the modified Darknet19 model. NOTE: Number of parameters and operations required are given in **millions** (10^6).

	Type	Filters	Size	Stride	Output	Ops, $\times 10^3$	Params, $\times 10^3$
1:	TCONV	100	4×4	1	$4 \times 4 \times 100$	64	32.10
2:	UPSAMPLE	100	2×2	1	$8 \times 8 \times 100$	6	0.00
3:	CONV	50	3×3	1	$8 \times 8 \times 50$	5,760	45.05
4:	UPSAMPLE	50	2×2	1	$16 \times 16 \times 50$	13	0.00
5:	CONV	20	3×3	1	$16 \times 16 \times 20$	4,608	9.02
6:	UPSAMPLE	20	2×2	1	$32 \times 32 \times 20$	20	0.00
7:	CONV	10	3×3	1	$32 \times 32 \times 10$	3,686	1.81
8:	UPSAMPLE	10	2×2	1	$64 \times 64 \times 10$	41	0.00
9:	CONV	1	3×3	1	$64 \times 64 \times 1$	737	0.09
					Total:	**14,936**	**88.1**

Table 5.6: Details of the architecture of the learned shape decoder in STS model. NOTE: Number of parameters and operations required are given in **thousands** (10^3).

	Type	Filters	Size	Stride	Output	Ops, $\times 10^6$	Params, $\times 10^6$
1:	TCONV	1024	5×5	1	$5 \times 5 \times 1024$	7	3.28
2:	TCONV	256	3×3	2	$11 \times 11 \times 256$	118	2.36
3:	CONV	192	3×3	1	$11 \times 11 \times 192$	107	0.44
4:	TCONV	128	3×3	2	$23 \times 23 \times 128$	54	0.22
5:	CONV	96	3×3	1	$23 \times 23 \times 96$	117	0.11
6:	TCONV	64	3×3	2	$47 \times 47 \times 64$	59	0.06
7:	CONV	1	3×3	1	$47 \times 47 \times 1$	3	0.00
					Total:	**463**	**6.5**

Table 5.7: Details of the architecture of the proposed large shape decoder. NOTE: Number of parameters and operations required are given in **millions** (10^6).

	Type	Filters	Size	Stride	Output	Ops, $\times 10^6$	Params, $\times 10^6$
1:	TCONV	1024	5×5	1	$5 \times 5 \times 1024$	7	3.28
2:	TCONV	256	3×3	2	$11 \times 11 \times 256$	118	2.36
3:	CONV	192	3×3	1	$11 \times 11 \times 192$	107	0.44
4:	TCONV	128	3×3	2	$23 \times 23 \times 128$	54	0.22
5:	CONV	96	3×3	1	$23 \times 23 \times 96$	117	0.11
6:	TCONV	64	3×3	2	$47 \times 47 \times 64$	59	0.06
7:	CONV	8	3×3	1	$47 \times 47 \times 8$	20	0.00
8:	DT	8	15×15	1	$47 \times 47 \times 8$	64	0.00
9:	CONV	1	1×1	1	$47 \times 47 \times 1$	0	0.00
					Total:	**545**	**6.5**

Table 5.8: Details of the architecture of the distance transform based neural shape decoder. NOTE: Number of parameters and operations required are given in **millions** (10^6).

5.A.2 Experimental Setup

The original STS model makes use of the Darknet model (Redmon, 2013–2016) as its training and inference workhorse (implemented in plain *C*). The vanilla version of the Darknet model is updated with shape prediction capabilities and additional software layers (in *C++*) for dataset loading and manipulation, shape mask reconstruction, and for evaluating the model on the task of instance segmentation [*]. We ran all our experiments on a single desktop machine with Intel Core i7-4960X CPU (3.6GHz , 6 cores) and NVidia GeForce GTX Titan X GPU (12 GB RAM). SBD dataset (Hariharan et al., 2011) is chosen as the experimental dataset to benchmark our models. This dataset is divided

[*]The project source code can be found at `https://github.com/torrvision/straighttoshapes`.

into 5623 *train* and 5732 validation images for training and evaluation respectively. The performance accuracy is measured in terms of $mAP^r_{0.5}$, $mAP^r_{0.7}$ and mAP^r_{vol} scores.

Due to memory constraints, the data is processed in the form of mini-batches of 8 training examples each. However, the parameters are only updated after accumulating gradients from 8 such mini-batches. In particular, batch-normalisation parameters are computed over 8 training examples, while gradient descent is performed over 64 such examples. The model is trained for 750 epochs (~ 65000 mini-batches) using Stochastic Gradient Descent (SGD) with momentum (0.9) and weight decay (5×10^{-4}). Leaky ReLU with an $\alpha = 0.1$ is the non-linearity used throughout our networks. The parameters in the initial layers of all the neural networks are borrowed from the pre-training of the Darknet model on ImageNet dataset (Russakovsky* et al., 2015) for the task of image classification [*]. The learning rate schedule for the training of the network on the task of instance segmentation is as follows:

Batch number	**Learning rate**
$1 - 200$	1×10^{-3}
$201 - 400$	2.5×10^{-3}
$401 - 20\,000$	5×10^{-3}
$20\,001 - 30\,000$	2.5×10^{-3}
$30\,001 - 40\,000$	1.25×10^{-3}
$40\,001 - 50\,000$	6.25×10^{-4}
$50\,001 - 60\,000$	3.16×10^{-4}
$60\,001 - 65\,000$	1.56×10^{-4}

The learning rate is kept low in the beginning in order to preserve the pre-trained weights. It is subsequently increased in order to speed up convergence and then reduced once more as we fine-tune the solution in the later stages of network training.

5.A.3 Loss Function Design

The model is trained solely as a regression problem and the loss is evaluated depending on where the ground truth objects appear in the input image.

Given a cell $1 \leq i \leq S^2$ containing an object, let $\hat{\mathbf{b}}_{ij}$ denote the parameters of the predicted bounding box at that cell location. We compute iou_{ij} as the IoU between the

[*]Pre-trained Darknet model can be downloaded from `https://pjreddie.com/darknet/yolo/`.

ground truth box and the predicted box j and assign $b_i = \text{argmax}_{1 \leq j \leq B} iou_{ij}$ as the index of the best prediction in cell i. Further, we define indicators $\nu_{ij} = \mathbb{1}\{j = b_i\}$ and $\mu_i = \mathbb{1}\{\text{cell } i \text{ contains an object}\}$ to capture the truth of the predicted box being the best fit and the cell containing the ground truth object respectively.

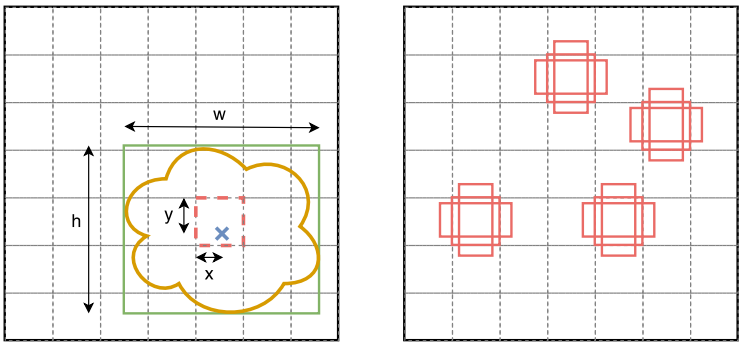


Figure 5.11: (Left) Grid cell (red) containing the center of the object (marked as a blue cross) is responsible for predicting the object's bounding box (green). The center coordinates are normalised relative to the grid cell, while the dimensions of the bounding-box are normalised relative to the dimensions of the complete image. (Right) There are 3 anchors boxes of different aspect ratios per grid cell location. The illustration displays several of these boxes at different locations in the image. Each anchor box gets assigned the ground truth object that shares the highest IoU with the anchor. Clearly, different anchors specialise in predicting objects of different aspect ratios.

Then the different terms of the objective function can be defined as follows:

$$L_{coord}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{S^2} \mu_i \sum_{j=1}^{B} \nu_{ij} \left[(\hat{x}_{ij} - x_i)^2 + (\hat{y}_{ij} - y_i)^2 + (\hat{\psi}_{ij} - \psi_i)^2 + (\hat{\omega}_{ij} - \omega_i)^2 \right] \tag{5.7}$$

where $(x_i, y_i) \in [0, 1)^2$ describes the center coordinates of the ground truth bounding box (see Figure 5.11 above and §3.2.2 for how target values are constructed). Note that the model predicts the square root of the bounding box dimensions, i.e. $\psi_i = \sqrt{h_i}$, $\omega_i = \sqrt{w_i}$, for reasons described in §3.1.The model is penalised for having high confidence predictions when (i) there is no ground truth object in the cell location, or (ii) its bounding box prediction is not the current best for that cell location:

$$L_{conf}^{noobj}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{S^2} \sum_{j=1}^{B} (1 - \mu_i \nu_{ij})(\hat{c}_{ij} - 0)^2 \tag{5.8}$$

In contrast, when cell i does contain a ground truth object, the confidence score of the best overlapping bounding box is penalised for deviating from the associated IoU as follows:

$$L_{conf}^{obj}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{S^2} \mu_i \sum_{j=1}^{B} \nu_{ij} (\hat{c}_{ij} - iou_{ij})^2 \tag{5.9}$$

The conditional class probabilities are modelled independently from the bounding box predictions and also independently for each class (as C binary random variables),

$$L_{class}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{S^2} \mu_i \sum_{j=1}^{B} \nu_{ij} \sum_{k=1}^{C} (\hat{p}_{ik} - p_{ik})^2 \tag{5.10}$$

All the loss terms defined up until this point are used for the task of detection and are exactly as those used for training the YOLO detector. In order to address the task of segmentation the STS model additionally regresses to shape representation values as follows:

$$L_{shape}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{S^2} \mu_i \sum_{j=1}^{B} \nu_{ij} \sum_{k=1}^{M} (\hat{s}_{ijk} - s_{ik})^2 \tag{5.11}$$

, where s_{ik} denotes the target shape representation (see §3.2.2) while $\hat{s}_{ijk}$ denotes the predicted shape mask.

The overall objective of the optimisation problem is then expressed as a weighted sum of the above terms (Eq. 5.7)-(5.11):

$$L = \lambda_{coord} L_{coord} + \lambda_{noobj} L_{conf}^{noobj} + \lambda_{obj} L_{conf}^{obj} + \lambda_{class} L_{class} + \lambda_{shape} L_{shape} \tag{5.12}$$

, where the values $\lambda_{coord} = 5.0$, $\lambda_{noobj} = 0.5$, $\lambda_{obj} = 1.0$, $\lambda_{class} = 1.0$, and $\lambda_{shape} = 0.15$, are chosen via cross-validation.

Remark: Note that the prediction of the object bounding box and shape masks is decoupled in the STS model. This is to say that whenever the former introduces a discrepancy in the bounding box location, the latter is not corrected or adjusted in any way. This is in contrast to other state-of-the-art instance segmentation models (Dai et al., 2015; He et al., 2017), where the target mask is adjusted to the predicted bounding box and thus the error therein.

5.A.4 More Qualitative Results

The section contains some example images from the SBD validation set and their corresponding instance segmentation results. These results as visualised in the following 8 ways: the top row contains images overlaid with predicted instance masks, while the bottom row simply contains these masks on a black background. The segmentation masks (left to right) include those from the ground truth, original STS model, proposed Darknet19 with pre-trained shape decoder, and proposed Darknet19 trained end-to-end using distance transform based shape representations.

These examples, contained in Figures 5.12 to 5.17 demonstrate the instance segmentation capabilities and frequent drawbacks, for example, failure to model object confidence scores, detect particular instances, or predict precise object boundaries.

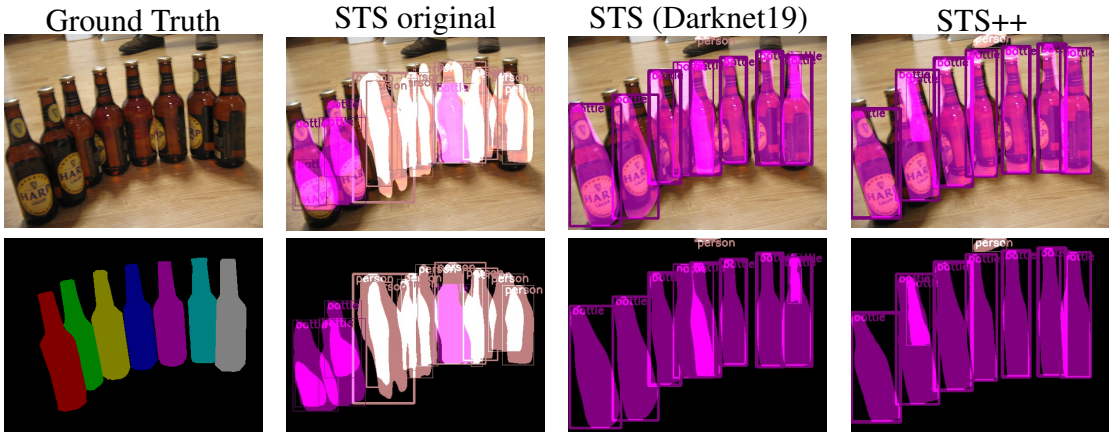


Figure 5.12: The original model confuses tall and narrow bottles with humans in standing pose. Our proposed methods do a better job at delineating overlapping object instances. Note also that the STS++ model is able to segment out the pair of shoes in the background, labelling it of *person* class. These objects are, however, not annotated in the dataset and the model is penalised for segmenting them.

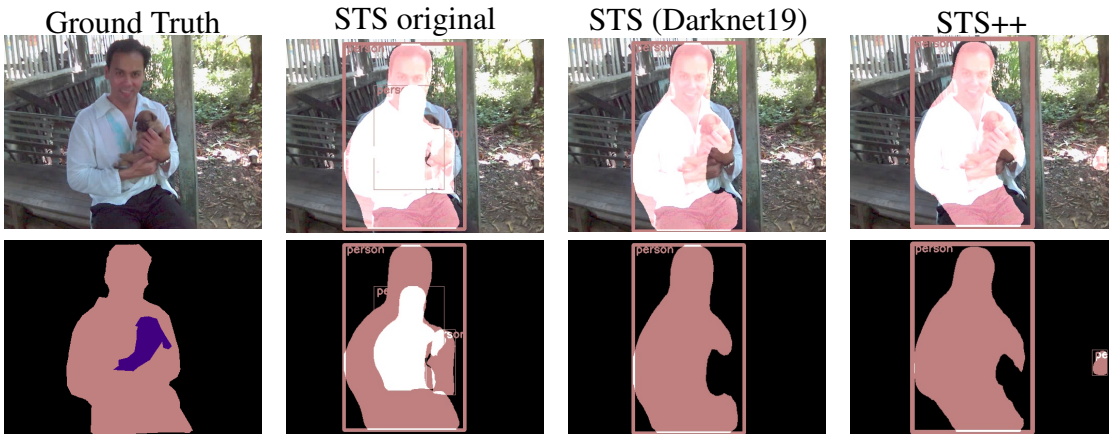


Figure 5.13: The original STS model predicts two high confidence bounding boxes for the same ground truth object. Although our proposed models overcome this flaw, they still fail to detect the puppy in the hands of the person.

Figure 5.14: The proposed models only get the coarse location of the bicycle but do a poor job at capturing the pixel level details.

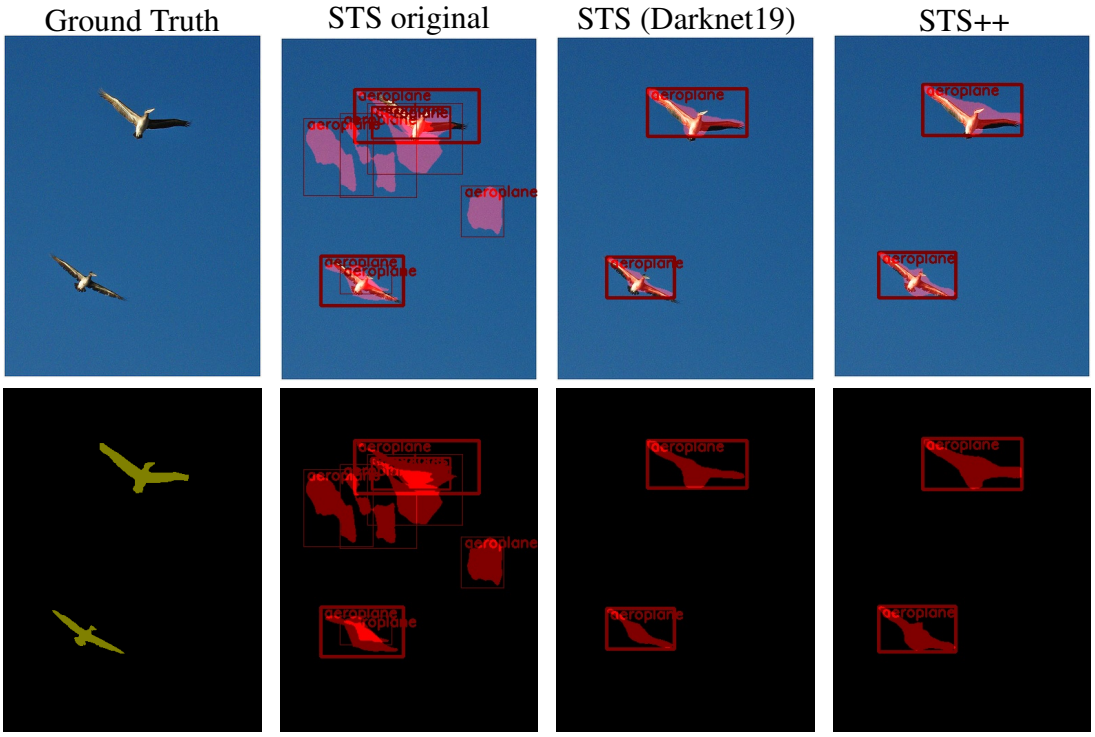


Figure 5.15: The models do better at delineating object boundaries, however, fail to correctly identify the objects as birds and confuse them with aeroplanes.

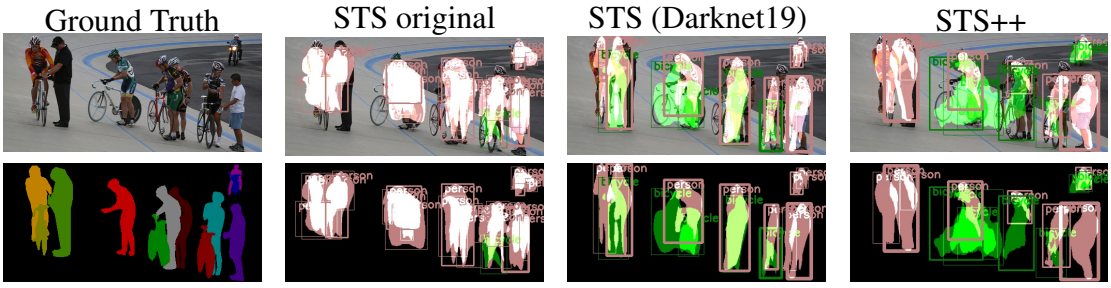


Figure 5.16: In some locations the models output more bounding boxes than there are ground truth objects, while in other places they miss the objects entirely.

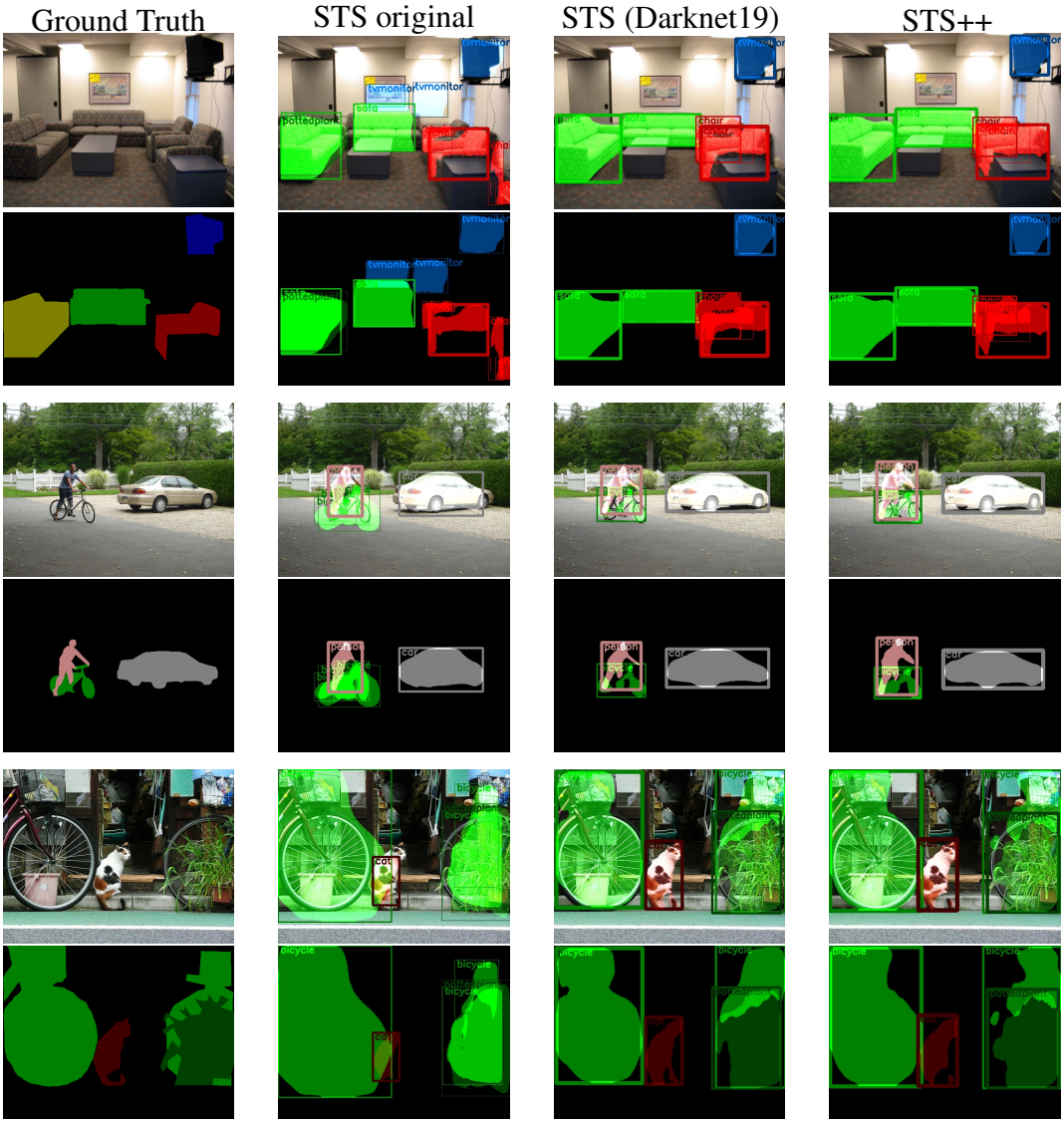


Figure 5.17: More instance segmentation results.

5.A.5 Detailed Error Analysis

	Prediction label					% of all FP error			
Approach	Corr	Loc	Sim	Dissim	Backgr	Loc	Sim	Dissim	Backgr
STS original	58.3	16.6	12.5	5.9	6.7	39.9	29.9	14.1	16.1
Batch Normalisation	63.3	17.8	8.4	4.2	6.2	48.4	23.0	11.6	17.0
Data Augmentation	66.2	16.3	8.0	4.0	5.5	48.2	23.7	11.9	16.2
Shared Predictor	64.3	16.3	8.6	4.9	5.9	45.8	24.0	13.8	16.5
Anchor Boxes	69.9	10.1	8.8	3.9	7.3	33.6	29.1	13.1	24.2
Darknet19	73.0	9.4	7.8	2.6	7.1	35.0	29.1	9.6	26.2
End-to-End	73.6	9.2	7.3	3.1	6.8	35.0	27.7	11.6	25.6
Large Decoder	73.1	9.4	7.2	3.3	7.1	34.7	26.6	12.3	26.3
Distance Transform (STS++)	73.7	9.0	7.1	3.0	7.1	34.3	27.0	11.5	27.2

Table 5.9: Detection errors of our proposed models on Pascal VOC 2007 test dataset. The errors are grouped as per the methodology presented in Hoiem et al. (2012).

STS original	Overall	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
TP: correct	58.3	59.2	61.7	54.5	57.0	39.6	61.0	65.9	66.2	58.8	63.2
FP: localisation	16.6	23.8	15.9	20.5	23.7	19.0	7.9	19.1	7.6	14.0	15.2
FP: similar	12.5	11.6	12.3	10.4	7.9	0.0	24.0	7.1	23.8	7.5	19.5
FP: dissimilar	5.9	1.0	8.7	4.0	2.5	20.9	0.8	2.2	0.8	9.0	0.0
FP: background	6.7	4.5	1.3	10.6	8.9	20.5	6.3	5.6	1.6	10.6	2.1
		dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor
TP: correct	58.3	53.5	62.3	60.8	59.3	56.6	37.0	61.4	68.7	65.9	53.7
FP: localisation	16.6	15.1	6.6	13.7	14.1	36.1	28.5	15.4	7.3	12.6	16.3
FP: similar	12.5	10.4	29.6	22.0	17.1	1.9	0.0	22.2	7.8	14.2	0.0
FP: dissimilar	5.9	12.4	0.2	1.8	8.7	2.5	14.5	0.0	12.6	0.7	14.4
FP: background	6.7	8.7	1.3	1.8	0.8	2.9	19.9	1.0	3.5	6.6	15.5

Table 5.10: Category-level detection errors of the original STS model on Pascal VOC 2007 test dataset.

Darknet19	Overall	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
TP: correct	73.0	70.1	71.0	73.1	68.2	49.8	76.4	79.4	78.9	72.9	79.3
FP: localisation	9.4	11.9	13.6	14.6	17.3	17.2	3.9	9.3	6.2	9.9	5.8
FP: similar	7.8	7.7	9.0	7.5	5.6	0.0	12.2	3.4	14.6	4.4	14.0
FP: dissimilar	2.6	1.9	4.1	0.3	0.8	8.1	0.4	1.0	0.0	3.5	0.0
FP: background	7.1	8.4	2.3	4.5	8.1	25.0	7.1	6.7	0.3	9.3	0.9
		dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor
TP: correct	73.0	77.6	75.7	79.2	75.6	71.4	49.0	75.9	84.6	80.8	71.7
FP: localisation	9.4	3.0	5.1	5.6	7.6	20.6	20.1	4.5	1.3	5.6	5.8
FP: similar	7.8	3.7	17.9	14.4	8.9	1.4	0.0	16.7	6.6	8.9	0.0
FP: dissimilar	2.6	8.4	0.0	0.3	3.8	1.6	6.2	0.3	4.8	0.3	6.1
FP: background	7.1	7.4	1.3	0.5	4.1	5.0	24.7	2.6	2.8	4.3	16.3

Table 5.11: Category-level detection errors of our proposed Darknet19 model on Pascal VOC 2007 test dataset.

Large Decoder	Overall	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
TP: correct	73.1	70.7	74.0	70.7	67.7	50.7	78.3	79.4	80.0	73.2	77.5
FP: localisation	9.4	14.1	10.0	16.0	15.8	16.9	3.1	10.3	7.0	9.6	4.9
FP: similar	7.2	6.4	6.7	8.2	4.1	0.0	9.1	2.7	12.2	4.3	16.1
FP: dissimilar	3.3	2.6	4.4	0.5	1.8	9.7	1.6	0.9	0.5	4.5	0.0
FP: background	7.1	6.1	4.9	4.7	10.7	22.7	7.9	6.6	0.3	8.4	1.5
		dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor
TP: correct	73.1	78.9	77.5	78.2	74.0	71.5	49.3	75.9	83.8	77.5	72.0
FP: localisation	9.4	3.3	4.5	6.3	8.7	21.1	15.7	6.8	2.0	6.3	4.7
FP: similar	7.2	3.0	16.2	14.7	8.9	1.5	0.0	14.8	5.3	9.3	0.0
FP: dissimilar	3.3	7.0	0.2	0.0	6.5	1.4	10.8	0.3	6.8	0.7	6.1
FP: background	7.1	7.7	1.5	0.8	1.9	4.5	24.2	2.3	2.0	6.3	17.2

Table 5.12: Category-level detection errors of our proposed Large Decoder model on Pascal VOC 2007 test dataset.

Approach	$mAP^r_{0.5}$	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
STS original	33.5	58.6	33.7	31.2	17.6	11.0	65.5	37.0	62.6	6.2	26.1
Batch Normalisation	38.6	58.9	39.3	38.2	18.6	15.1	67.9	41.4	70.6	7.6	34.6
Data Augmentation	42.3	64.7	38.4	41.0	23.6	14.5	71.0	42.4	74.3	8.7	44.4
Shared Predictor	41.4	63.0	42.2	41.9	22.4	14.8	69.3	39.8	73.9	8.1	40.3
Anchors	48.5	69.6	47.0	51.9	29.9	22.9	71.9	50.5	77.5	14.9	49.3
Darknet19	52.2	72.0	52.7	54.6	30.6	**28.0**	74.8	**56.8**	81.8	21.1	**56.9**
End-to-End	51.7	65.3	52.6	54.0	31.3	26.5	**77.7**	55.3	80.7	19.0	55.4
Large Decoder	52.3	**73.0**	52.5	55.1	**34.8**	24.9	75.3	55.3	**82.0**	20.7	54.6
Distance Transform (STS++)	**53.2**	72.9	**53.6**	**58.5**	32.4	25.8	74.9	56.5	81.8	**22.8**	55.0
SDS (Hariharan et al., 2014)	49.7	68.4	49.4	52.1	32.8	33.0	67.8	53.6	73.9	19.9	43.7
Arnab & Torr (2017)	62.0	80.3	52.8	68.5	47.4	39.5	79.1	61.5	87.0	28.1	68.3
		dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor
STS original	33.5	13.5	49.7	31.0	37.9	37.7	7.0	31.2	20.0	62.7	29.2
Batch Normalisation	38.6	11.9	58.6	45.5	45.6	36.4	14.1	34.4	23.9	69.8	39.8
Data Augmentation	42.3	15.9	64.1	47.1	53.1	40.9	15.7	44.1	27.0	72.0	43.1
Shared Predictor	41.4	19.3	60.4	50.3	51.2	37.5	13.4	41.0	28.1	69.0	41.9
Anchors	48.5	24.2	68.4	55.0	59.7	49.7	18.6	53.7	31.1	74.5	50.5
Darknet19	52.2	26.0	71.4	61.6	**62.4**	54.4	19.4	53.2	36.2	76.1	54.5
End-to-End	51.7	25.1	72.9	57.9	58.6	51.9	22.5	**56.6**	**37.5**	77.9	56.2
Large Decoder	52.3	25.2	71.9	**64.8**	58.6	54.0	19.6	54.9	36.5	76.9	55.7
Distance Transform (STS++)	**53.2**	**26.2**	**74.1**	62.5	59.8	**57.7**	**22.7**	56.1	36.4	**78.5**	**56.4**
SDS (Hariharan et al., 2014)	49.7	25.7	60.6	55.9	58.9	56.7	28.5	55.6	32.1	64.7	60.0
Arnab & Torr (2017)	62.0	35.5	86.1	73.9	66.1	63.8	32.9	65.3	50.4	81.4	71.4

Table 5.13: Average precision ($AP^r_{0.5}$) estimates for the task of detection on SBD validation set for all the proposed methods compared with state-of-the-art models.

Approach	$mAP^r_{0.7}$	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
STS original	14.9	20.2	10.9	7.8	5.2	5.5	52.3	20.8	41.3	0.6	7.5
Batch Normalisation	17.4	22.4	13.5	12.3	5.3	6.0	55.5	21.6	46.8	0.4	11.4
Data Augmentation	20.8	27.9	11.9	14.2	8.8	6.9	57.3	25.5	53.4	0.8	15.9
Shared Predictor	20.5	28.5	15.2	15.0	9.6	7.0	55.5	24.1	51.1	0.8	17.9
Anchors	26.7	30.2	19.1	18.8	10.9	11.4	65.0	35.9	61.0	2.2	24.8
Darknet19	31.1	40.3	**24.9**	22.7	11.6	14.5	66.7	39.5	67.2	**5.9**	26.3
End-to-End	27.9	23.4	19.3	18.6	13.8	13.4	**70.3**	39.8	60.4	3.2	23.3
Large Decoder	31.9	**42.3**	22.9	24.5	**18.1**	**15.2**	67.5	40.3	**68.8**	5.8	31.1
Distance Transform (STS++)	**32.3**	39.1	23.8	**24.9**	15.8	12.8	65.4	**41.3**	66.8	5.6	**31.3**
Arnab & Torr (2017)	44.8	69.0	27.4	52.7	26.4	22.4	70.3	46.0	74.7	9.6	46.8
		dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor
STS original	14.9	2.0	23.7	3.9	13.7	8.9	1.4	5.5	9.7	45.8	11.0
Batch Normalisation	17.4	2.1	24.9	7.4	15.9	8.4	2.2	9.0	12.4	55.6	14.6
Data Augmentation	20.8	3.3	33.6	10.4	21.8	11.2	3.7	11.9	15.2	57.8	23.7
Shared Predictor	20.5	5.9	33.2	12.0	22.6	11.5	3.0	10.0	13.9	54.3	19.4
Anchors	26.7	7.9	42.5	17.9	26.7	17.9	4.3	19.9	19.2	63.8	33.8
Darknet19	31.1	10.5	48.6	**24.3**	35.1	23.7	4.6	26.8	24.5	64.8	40.5
End-to-End	27.9	11.7	39.6	14.5	25.0	18.8	**6.6**	24.0	**29.1**	61.6	40.6
Large Decoder	31.9	8.5	48.8	22.1	**35.8**	24.7	5.7	25.8	25.4	63.1	**41.9**
Distance Transform (STS++)	**32.3**	**13.2**	**50.8**	22.6	35.7	**27.4**	6.4	**29.0**	26.8	**65.4**	41.0
Arnab & Torr (2017)	44.8	16.9	71.6	48.4	46.3	40.3	14.8	47.6	36.5	69.7	58.2

Table 5.14: Average precision ($AP^r_{0.7}$) estimates for the task of detection on SBD validation set for all the proposed methods compared with state-of-the-art models.

6
Learning to Pay Attention

Contents

We propose an end-to-end-trainable attention module for convolutional neural network (CNN) architectures built for image classification. The module takes as input the 2D feature vector maps which form the intermediate representations of the input image at different stages in the CNN pipeline, and outputs a 2D matrix of scores for each such

map. Standard CNN architectures are modified through the incorporation of this module, and trained under the constraint that a convex combination of the intermediate 2D feature vectors, as parameterised by the score matrices, must *alone* be used for classification. Incentivised to amplify the relevant and suppress the irrelevant or misleading, the scores thus assume the role of attention values. Our experimental observations provide clear evidence to this effect: the learned attention maps neatly highlight the regions of interest while suppressing background clutter. Consequently, the proposed function is able to bootstrap standard CNN architectures for the task of image classification, demonstrating superior generalisation over 6 unseen benchmark datasets. When binarised, our attention maps outperform other CNN-based attention maps, traditional saliency maps, and top object proposals at the task of weakly supervised segmentation as demonstrated on the Object Discovery dataset. We also demonstrate improved robustness against the fast gradient sign method of adversarial attack.

6.1 Introduction

Feed-forward convolutional neural networks (CNNs) have demonstrated impressive results on a wide variety of visual tasks, such as image classification, captioning, segmentation, and object detection. However, the visual reasoning which they implement in solving these problems remains largely inscrutable, impeding understanding of their successes and failures alike.

One approach to visualising and interpreting the inner workings of CNNs is the attention map: a scalar matrix representing the relative importance of layer activations at different 2D spatial locations with respect to the target task (Simonyan et al., 2013). This notion of a nonuniform spatial distribution of relevant features being used to form a task-specific representation, and the explicit scalar representation of their relative relevance, is what we term 'attention'. Previous works have shown that for a classification CNN trained using image-level annotations alone, extracting the attention map provides a straightforward way of determining the location of the object of interest (Cao et al., 2015; Zhou et al., 2016) and/or its segmentation mask (Simonyan et al., 2013), as well as identifying discriminative visual properties across classes (Zhou et al., 2016). More

recently, it has also been shown that training smaller networks to mimic the attention maps of larger and higher-performing network architectures can lead to gains in classification accuracy of those smaller networks (Zagoruyko & Komodakis, 2016).

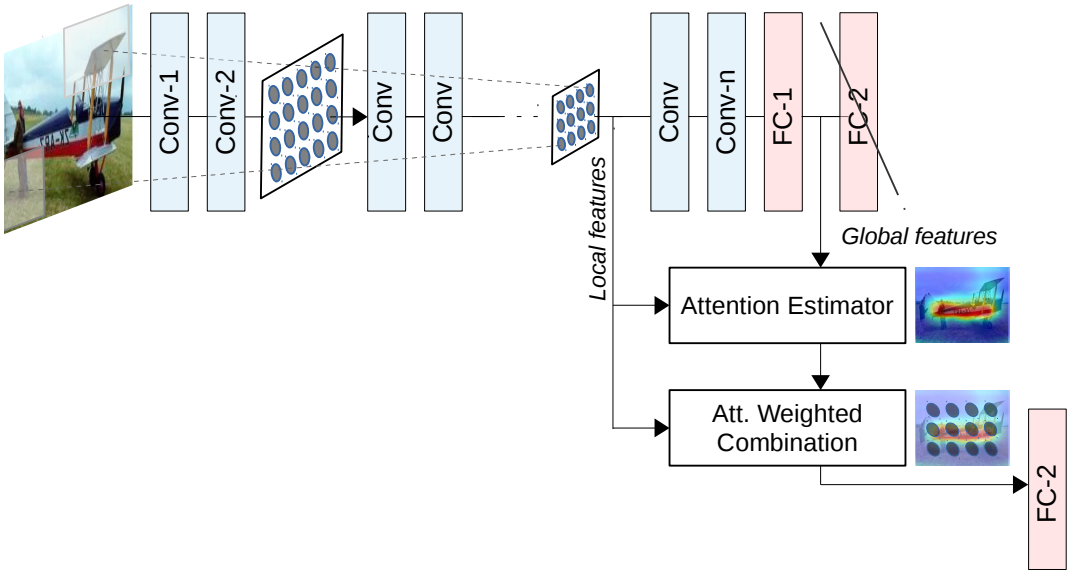


Figure 6.1: Overview of the proposed attention mechanism.

The works of Simonyan et al. (2013); Oquab et al. (2015); Cao et al. (2015); Zhou et al. (2016) represent one series of increasingly sophisticated techniques for estimating attention maps in classification CNNs. However, these approaches share a crucial limitation: all are implemented as post-hoc additions to fully trained networks. On the other hand, integrated attention mechanisms whose parameters are learned over the course of end-to-end training of the entire network have been proposed, and have shown benefits in various applications that can leverage attention as a cue. These include attribute prediction (Seo et al., 2016), machine translation (Bahdanau et al., 2014), image captioning (Xu et al., 2015a; You et al., 2016; Mun et al., 2016) and visual question answering (VQA) (Xu & Saenko, 2016; Yang et al., 2016). Similarly to these approaches, we here represent attention as a probabilistic map over the input image locations, and implement its estimation via an end-to-end framework. The novelty of our contribution lies in repurposing the global image representation as a query to estimate multi-scale attention in classification, a task which, unlike *e.g.* image captioning or VQA, does not naturally involve a query. Our work is also similar in spirit to that of Oquab et al. (2015)

which has sought to attend-to or localise objects in cluttered scenes by boosting and identifying highest scoring image windows corresponding to image-level class labels.

Figure 6.1 provides an overview of the proposed method. Henceforth, we will use the terms 'local features' and 'global features' to refer to the features output by some layer of the CNN whose effective receptive fields are, respectively, contiguous proper subsets of the image ('local') and the entire image ('global'). By defining a compatibility measure between local and global features, we redesign standard architectures such that they must classify an input image using *only* a weighted combination of local features, with the weights represented here by the compatibility scores. The network is thus forced to learn these compatibility scores as a pattern of attention that is relevant to solving the task at hand.

We experiment with applying the proposed attention mechanism to the popular CNN architectures of VGGNet (Simonyan & Zisserman, 2015) and ResNet (He et al., 2016), and capture fine-to-coarse attention maps at multiple levels. We observe that the proposed mechanism can bootstrap baseline CNN architectures for the task of image classification: for example, adding attention to the VGG model offers an accuracy gain of 7% on CIFAR-100. Our use of attention-weighted representations leads to improved fine-grained recognition and superior generalisation on 6 benchmark datasets for domain-shifted classification. As observed on models trained for fine-grained bird recognition, attention aware models offer improved resistance to adversarial fooling at low and moderate L_∞ noise norms. The trained attention maps outperform other CNN-derived attention maps (Zhou et al., 2016), traditional saliency maps (Jiang et al., 2013; Zhang & Sclaroff, 2013), and top object proposals on the task of weakly supervised segmentation as evaluated on the Object Discovery dataset (Rubinstein et al., 2013). In §6.5, we present sample results which suggest that these improvements may owe to the method's tendency to highlight the object of interest while suppressing background clutter.

6.2 Related Work

Attention in CNNs is implemented using one of two main schemes - post hoc network analysis or trainable attention mechanisms. The former scheme has been predominantly

employed to access network reasoning for the task of visual object recognition (Simonyan et al., 2013; Zhou et al., 2016; Cao et al., 2015). Simonyan et al. (2013) approximate CNNs as linear functions, interpreting the gradient of a class output score with respect to the input image as that class's spatial support in the image domain, *i.e.* the class attention map. Importantly, they are one of the first to successfully demonstrate the use of attention for localising objects of interest using image-level category labels alone. The work of Oquab et al. (2015) extends this proposition to a multi-class setting. It attends to image objects in cluttered visual scenes through the analysis of multi-size windows at multiple different scales, and by finding and promoting those that exhibit the highest scores for ground-truth class labels pertaining to the given scene image. Zhou et al. (2016) apply the classifier weights learned for image-level descriptors to patch descriptors, and the resulting class scores are used as a proxy for attention. Their improved localisation performance comes at the cost of classification accuracy. Cao et al. (2015) introduce attention in the form of binary nodes between the network layers of Simonyan et al. (2013). At test time, these binary nodes are adjusted so as to maximise the classification performance of the network architecture in an additional parameter tuning step. Notably, all of the above methods extract attention from fully trained CNN classification models, *i.e.* via post-processing. Subsequently, as discussed shortly, many methods have explored the performance advantages of optimising the weights of the attention unit in tandem with the original network weights.

Trainable Attention in CNNs falls under two main categories - hard (stochastic) and soft (deterministic). In the former, a hard decision is made regarding the use of an image region, often represented by a low-order parameterisation, for inference (Mnih et al., 2014; Xu et al., 2015a). The implementation is non-differentiable and relies on a sampling-based technique called REINFORCE for training, which makes optimising these models more difficult. On the other hand, the soft-attention method is probabilistic and thus amenable to training by backpropagation. The method of Jaderberg et al. (2015) lies at the intersection of the above two categories. It uses a parameterised transform to estimate hard attention on the input image deterministically, where the parameters of the image transformation are estimated using differentiable functions. The soft-attention

method of Seo et al. (2016) demonstrates improvements over the above by implementing nonuniform non-rigid attention maps which are better suited to natural object shapes seen in real images. In comparison, Oquab et al. (2015) implement soft-attention in the form of class scores over image windows of predetermined dimensions. It is in the former direction of non-rigid attention maps that we develop our current work.

Trainable Soft-attention in CNNs has mainly been deployed for query-based tasks (Bahdanau et al., 2014; Xu et al., 2015a; Seo et al., 2016; You et al., 2016; Mun et al., 2016; Xu & Saenko, 2016; Yang et al., 2016). As is done with the exemplar captions of Mun et al. (2016), the questions of Xu & Saenko (2016); Yang et al. (2016), and the source sentences of Bahdanau et al. (2014), we here map an entire image to a high-dimensional representation which is, in turn, matched against patch-level representations in order to highlight those image patches that are relevant for the output prediction and suppress those that are not. We draw a close comparison to the progressive attention approach of Seo et al. (2016) for attribute prediction. However, there are some noteworthy differences. Their method uses a one-hot encoding of category labels to query the image: this is unavailable to us and we hence substitute a learned representation of the global image. In addition, their sequential mechanism refines a single attention map along the length of the network pipeline. This doesn't allow for the expression of a complementary focus on different parts of the image at different scales as leveraged by our method, illustrated for the task of fine-grained recognition in §6.5.

The Applications of Attention, in addition to facilitating the task of output class prediction, are varied. The current work covers the following areas:

· *Domain Shift*: A traditional approach to handling domain shift in CNNs involves fine-tuning on the new dataset, which may require thousands of images from each target category for successful adaptation. We position ourselves amongst approaches that use attention (Zhou et al., 2016; Jaderberg et al., 2015) to better handle domain changes, particularly those involving background content, occlusion, and object pose variation, by selectively focusing on the objects of interest.

· *Weakly Supervised Semantic Segmentation*: This area investigates image segmentation attempted using minimal annotations i.e. in the form of scribbles, bounding boxes,

or image-level category labels. Our work uses category labels and is related to the soft-attention approach of Hong et al. (2016). However, unlike the aforementioned, we do not explicitly train our model for the task of segmentation using any kind of pixel-level annotations. We simply evaluate the binarised spatial attention maps, learned as a by-product of training for image classification, for their ability to segment objects.

· *Adversarial Robustness*: The work by Goodfellow et al. (2015) explores the ease of fooling deep classification networks by adding an imperceptible perturbation to the input image, implemented as an epsilon step in the direction opposite to the predicted class score gradient. The works by Wang et al. (2016) and Gao et al. (2017b) argue that this vulnerability comes from relying on spurious or non-oracle features for classification. Consequently, Gao et al. (2017b) demonstrate increased adversarial robustness by identifying and masking the likely adversarial feature dimensions. We experiment with performing such suppression in the spatial domain.

6.3 Proposed Approach

The core goal of this work is to incorporate attention in CNNs in order to encourage and identify the use of specific input image regions that are most relevant for making classification decisions. This approach is premised on the hypothesis that there is benefit to identifying salient image regions and amplifying their influence, while likewise suppressing the irrelevant and potentially confusing information in other regions. In particular, we expect that enforcing a more focused and parsimonious use of image information should aid in generalisation over changes in the data distribution, as occurs for instance when training on one set and testing on another. Thus, we propose a trainable attention estimator and illustrate how to integrate it into standard CNN pipelines so as to influence their output as outlined above. The method is based on enforcing a notion of compatibility between the local feature vectors and a global image feature. The local features are extracted at intermediate stages in the CNN pipeline, while the global feature is normally a vector that is fed to the linear classification layers at the end of the pipeline.

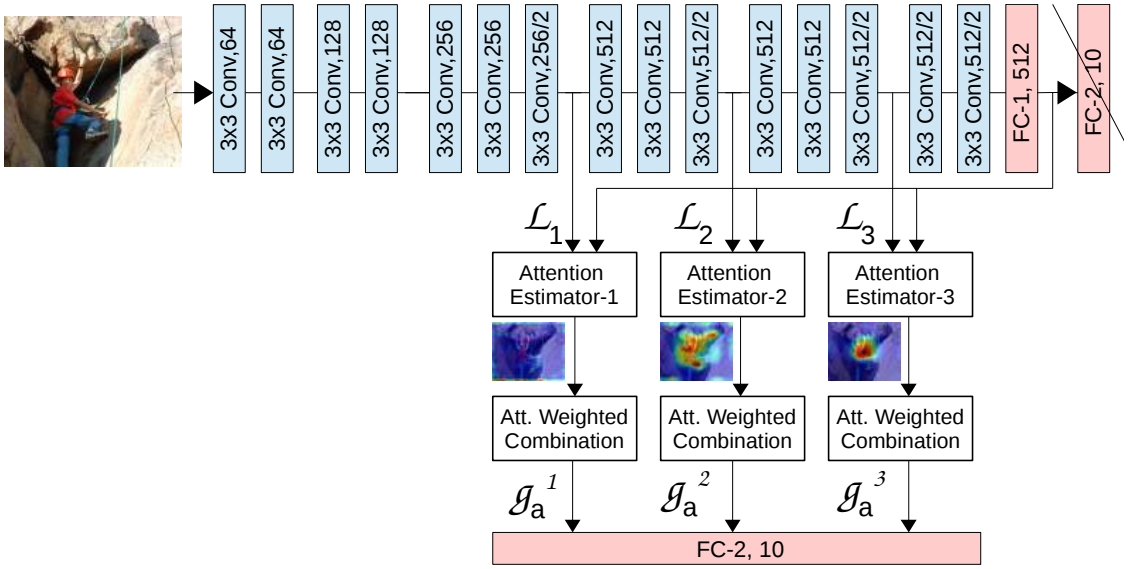


Figure 6.2: Attention introduced at 3 distinct layers of VGG. Lowest level attention maps appear to focus on the surroundings (*i.e.*, the rocky mountain), intermediate level maps on object parts (*i.e.*, harness and climbing equipment) and the highest level maps on the central object.

We implement attention-aware classification by restricting the classifier to use *only* a collection of local feature vectors, as chosen and weighted by the compatibility scores, in classifying examples. Thus, at the time of training, the compatibility scores assume the role of attention, allowing the net to choose which local features to amplify and which to suppress in order to optimise the classification objective. We will first discuss the modification to the network architecture and the method of training it, given a choice of compatibility function. We will then conclude the method description by presenting alternate choices of the compatibility function.

6.3.1 Design and Training of Attention Submodule

The proposed approach is illustrated in Figure 6.2. Denote by $\mathcal{L}^s = \{\boldsymbol{\ell}_1^s, \boldsymbol{\ell}_2^s, \cdots, \boldsymbol{\ell}_n^s\}$ the set of feature vectors extracted at a given convolutional layer $s \in \{1, \cdots, S\}$. Here, each $\boldsymbol{\ell}_i^s$ is the vector of output activations at the spatial location i of n total spatial locations in the layer. The global feature vector $\boldsymbol{g}$ has the entire input image as support and is output by the network's series of convolutional and nonlinear layers, having only to pass through the final fully connected layers to produce the original architecture's class score for that input. Assume for now the existence of a compatibility function $\mathcal{C}$ which

takes two vectors of equal dimension as arguments and outputs a scalar compatibility score: this will be specified in 6.3.2.

The method proceeds by computing, for each of one or more layers s, the set of compatibility scores $\mathcal{C}(\hat{\mathcal{L}}^s, \boldsymbol{g}) = \{c_1^s, c_2^s, ...c_n^s\}$, where $\hat{\mathcal{L}}^s$ is the image of $\mathcal{L}^s$ under a linear mapping of the $\boldsymbol{\ell}_i^s$ to the dimensionality of $\boldsymbol{g}$. The compatibility scores are then normalised by a softmax operation:

$$a_i^s = \frac{\exp(c_i^s)}{\sum_j^n \exp(c_j^s)},\ i \in \{1 \cdots n\}. \tag{6.1}$$

The normalised compatibility scores $\mathcal{A}^s = \{a_1^s, a_2^s, \cdots a_n^s\}$ are then used to produce a single vector $\boldsymbol{g}_a^s = \sum_{i=1}^n a_i^s \cdot \boldsymbol{\ell}_i^s$ for each layer s, by simple element-wise weighted averaging. Crucially, the $\boldsymbol{g}_a^s$ now *replace* $\boldsymbol{g}$ as the global descriptor for the image. For a network trained under the restriction that the $\boldsymbol{g}_a^s$ alone are used to classify the input image, $\mathcal{A}$ corresponds to 'attention' as defined earlier.

In the case of a single layer ($S = 1$), the attention-incorporating global vector $\boldsymbol{g}_a$ is computed as described above and mapped onto a T-dimensional class-score vector, where T is the number of target output classes. This vector is then passed through a softmax layer to obtain class prediction probabilities $\{\hat{p}_1, \hat{p}_2, \cdots \hat{p}_T\}$. In the case of multiple layers ($S > 1$), we compare two options: concatenating the global vectors into a single vector $\boldsymbol{g}_a = [\boldsymbol{g}_a^1, \boldsymbol{g}_a^2, \cdots \boldsymbol{g}_a^S]$ and using this as the input to the linear classification step as above, or, using S different linear classifiers and averaging the output class probabilities. All free network parameters are learned in end-to-end training under a cross-entropy loss function.

6.3.2 Choice of Compatibility Function $\mathcal{C}$

The compatibility score function $\mathcal{C}$ can be defined in various ways. The alignment model of Bahdanau et al. (2014); Xu et al. (2015a) can be re-purposed as a compatibility function as follows:

$$c_i^s = \langle \boldsymbol{u}^s, \boldsymbol{\ell}_i^s + \boldsymbol{g} \rangle,\ i \in \{1 \cdots n\}, \tag{6.2}$$

In practice, given the free parameters between the local and the global image descriptors in a CNN pipeline, we can simplify the concatenation of the two descriptors to an

addition operation as shown in the equation above. This allows us to limit the total number of parameters of the attention unit. We then learn a single fully connected mapping from the resultant descriptors to their compatibility scores. Here, the weight vector $\boldsymbol{u^s}$ can be interpreted as learning the overarching set of features relevant to all the object categories in the dataset. In that sense, the weights may be seen as learning the general concept of objectness.

Alternatively, we can use the dot product between $\boldsymbol{g}$ and $\boldsymbol{\ell}_i^s$ as a measure of their compatibility:

$$c_i^s = \langle \boldsymbol{\ell}_i^s, \boldsymbol{g} \rangle, \; i \in \{1 \cdots n\}. \tag{6.3}$$

In this case, the relative magnitude of the scores would depend on the alignment between $\boldsymbol{g}$ and $\boldsymbol{\ell}_i^s$ in the high-dimensional feature space and the strength of activation of $\boldsymbol{\ell}_i^s$.

6.3.3 Intuition

In a standard CNN architecture, a global image descriptor $\boldsymbol{g}$ is derived from the input image and passed through a fully connected layer to obtain class prediction probabilities. The network must express $\boldsymbol{g}$ via a mapping of the input into a high-dimensional space in which salient higher-order visual concepts are represented by different dimensions, so as to render the classes linearly separable from one another. Our method encourages the filters *earlier* in the CNN pipeline to learn similar mappings, compatible with the one that produces $\boldsymbol{g}$ in the original architecture. This is achieved by allowing a local descriptor $\boldsymbol{\ell}_i$ of an image patch to contribute to the final classification step only in proportion to its compatibility with $\boldsymbol{g}$ as detailed above. That is, $\mathcal{C}(\hat{\boldsymbol{\ell}}_i, \boldsymbol{g})$ should be high if and only if the corresponding patch contains parts of the dominant image category. Note that this implies that the effective filters operating over image patches in the layers s must represent relatively *mature* features with respect to the classification goal. We thus expect to see the greatest benefit in deploying attention relatively late in the pipeline. Further, different kinds of class details are more easily accessible at different scales. Thus, in order to facilitate the learning of diverse and complementary attention-weighted features, we propose the use of attention over different spatial resolutions. The combination of the two factors stated above results in our deploying the attention units after the

convolutional blocks that are late in the pipeline, but before their corresponding max-pooling operations, *i.e.* before a drop in the spatial resolution. Note also that the use of the softmax function in normalising the compatibility scores enforces $0 \leq a_i \leq 1$ $\forall i \in \{1 \cdots n\}$ and $\sum_i a_i = 1$, *i.e.* that the combination of feature vectors is convex. This ensures that features at different spatial locations must effectively compete against one another for their share of the attention map. The compatibility scores thus serve as a robust proxy for attention in the classification pipeline.

6.4 Experimental Setup

To incorporate attention into the VGG network, we move each of the first 2 max-pooling layers of the baseline architecture after each of the 2 corresponding additional convolutional layers that we introduce at the end of the pipeline. By pushing the pooling operations further down the pipeline, we ensure that the local layers used for estimating attention have a higher resolution. Our modified model has 17 layers: 15 convolutional and 2 fully connected. The output activations of layer-16 (fc) define our global feature vector $\boldsymbol{g}$. We use the local feature maps from layers 7, 10, and 13 (convolutional) for estimating attention. We compare our approach with the activation-based attention method of Zhou et al. (2016), and the progressive attention mechanism of Seo et al. (2016).

For Residual Networks (RNs) (He et al., 2016), we use a 164-layered network. We replace the spatial average-pooling step after the computational block-4 with extra convolutional and max-pooling steps to obtain the global feature $\boldsymbol{g}$. The outputs of blocks 2, 3, and 4 serve as the local descriptors for attention. For more details about network architectures refer to §6.A.2. Note that, for both of the above architectures, if the dimensionality of $\boldsymbol{g}$ and the local features of a layer s differ, we project $\boldsymbol{g}$ to the lower-dimensional space of the local features, instead of the reverse. This is done in order to limit the parameters at the classification stage. The global vector $\boldsymbol{g}$, once mapped to a given dimensionality, is then shared by the local features from different layers s as long as they are of that dimensionality.

We refer to the network *Net* with attention at the last level as *Net-att*, at the last two levels as *Net-att2*, and at the last three levels as *Net-att3*. We denote by *dp* the

use of the dot product for matching the global and local descriptors and by *pc* the use of parametrised compatibility. We denote by *concat* the concatenation of descriptors from different levels for the final linear classification step. We use *indep* to denote the alternative of independent prediction of probabilities at different levels using separate linear classifiers: these probabilities are averaged to obtain a single score per class.

Note that the network architectures using the proposed attention mechanism are all trained from a random initialisation i.e. the attention weights are tuned jointly with the rest of the network parameters from a random start. We evaluate the benefits of incorporating attention into CNNs for the primary tasks of image classification and fine-grained object category recognition. We also examine robustness to the kind of adversarial attack discussed by Goodfellow et al. (2015). Finally, we study the quality of attention maps as segmentations of image objects belonging to the network-predicted categories. For more details of the datasets and their pre-processing routines refer to §6.A.1, different network architectures §6.A.2, network training schedules §6.A.3 , and task-specific experimental setups §6.A.4.

6.5 Results and Discussion

6.5.1 Image Classification and Fine-Grained Recognition

Model	Top-1 error with standard deviation. CIFAR-10	CIFAR-100
| Existing architectures |		
VGG (Simonyan & Zisserman, 2015)	7.77 (0.08)	30.62 (0.16)
VGG-GAP (Zhou et al., 2016)	9.87 (0.10)	31.77 (0.13)
VGG-PAN (Seo et al., 2016)	6.29 (0.03)	**24.35 (0.14)**
RN-164 (He et al., 2016)	**6.03 (0.18)**	25.34 (0.16)
| Architectures with attention |		
(VGG-att)-dp	6.14 (0.06)	24.22 (0.08)
(VGG-att2)-indep-dp	5.91 (0.05)	23.24 (0.07)
(VGG-att2)-concat-dp	5.86 (0.05)	23.91 (0.11)
(VGG-att)-pc	5.67 (0.04)	23.70 (0.07)
(VGG-att2)-indep-pc	5.36 (0.06)	24.00 (0.06)
(VGG-att2)-concat-pc	**5.23 (0.04)**	23.19 (0.04)
(VGG-att3)-concat-pc	6.34 (0.07)	**22.97 (0.04)**

Table 6.1: CIFARs: Top-1 classification errors.

Model	Top-1 error with standard deviation. CUB-200-2011	SVHN
| Existing architectures |		
VGG (Simonyan & Zisserman, 2015)	34.64 (0.26)	**4.27 (0.04)**
VGG-GAP (Zhou et al., 2016)	29.50* (–)	5.84 (0.09)
VGG-PAN (Seo et al., 2016)	31.46 (0.16)	8.02 (0.06)
RN-34 (Zagoruyko & Komodakis, 2016)	**26.5* (–)**	–
| Architectures with attention |		
(VGG-att2)-concat-pc	**26.80 (0.16)**	3.74 (0.05)
(VGG-att3)-concat-pc	26.95 (0.10)	**3.52 (0.04)**

Table 6.2: Fine-grained recognition: Top-1 errors. * denotes results from publications.

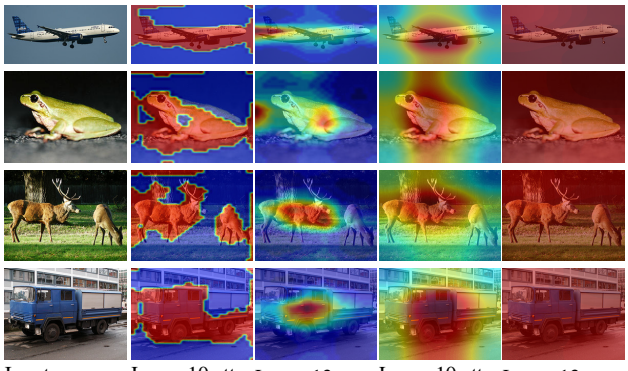


Figure 6.3: Attention maps from VGG-att2 trained on low-res CIFAR-10 dataset focus sharply on the objects in high-res ImageNet images of CIFAR-10 categories; contrasted here with the activation-based attention maps of Zagoruyko & Komodakis (2016).

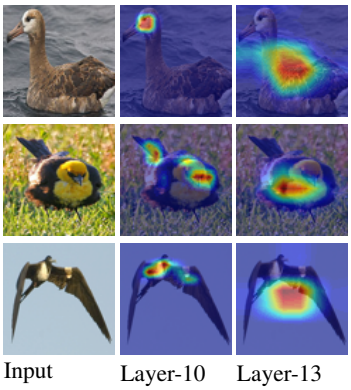


Figure 6.4: VGG-att2 trained on CUB-200 for fine-grained bird recognition task: layer-10 learns to fixate on eye and beak regions, layer-13 on plumage and feet.

When incorporated in a standard VGG architecture, the proposed attention mechanism provides noticeable performance improvement over baseline models (*e.g.* VGG, RN) and existing attention mechanisms (*e.g.* GAP, PAN) for visual recognition tasks, as seen in Table 6.1 & 6.2. Specifically, the *VGG-att2-concat-pc* model achieves a 2.5% and 7.4% improvement over baseline VGG for CIFAR-10 and CIFAR-100 classification, and 7.8% and 0.5% improvement for fine-grained recognition of CUB and SVHN categories. As is evident from Figure 6.3, the attention mechanism enables the network to focus on the object of interest while suppressing the background regions. This selective focus seems to provide greater benefit when there is a larger number of classes to distinguish between, for example in the case of CIFAR-100 and CUB datasets. Moreover, for the task of fine-grained recognition, different layers learn specialised focus on different object

parts as can be seen in Figure 6.4. Note that the RN-34 model for CUB from Table 6.2 is pre-trained on ImageNet. In comparison, our networks are pre-trained using the much smaller and less diverse CIFAR-100. In spite of the low training cost, our networks are on par with the former in terms of accuracy. Importantly, despite the increase in the total number of network parameters due to the attention units, the proposed networks generalise exceedingly well to the test set. We are unable to compare directly with the CUB result of Jaderberg et al. (2015) due to a difference in dataset pre-processing. However, we improve over PAN (Seo et al., 2016) by 4.5%, which has itself been shown to outperform the approach of Jaderberg et al. (2015) at a similar task. For the remaining experiments, *concat-pc* is our implicit attention design unless specified otherwise. When the same attention mechanism is introduced into RNs we observe a marginal drop in performance: 0.9% on CIFAR-10 and 1.5% on CIFAR-100. It is possible that the skip-connections in RNs work in a manner similar to the proposed attention mechanism, *i.e.* by allowing select local features from earlier layers to skip through and influence inference. While this might make the performance improvement due to attention redundant, our method, unlike the skip-connections, is able to provide explicit attention maps which are more informative regarding the visual reasoning of networks, and which can also be used for auxiliary tasks such as weakly supervised object segmentation.

Finally, the global feature vector is used as a query in our attention calculations. By changing the query vector, one could expect to affect the predicted attention pattern. A brief investigation of the extent to which the two compatibility functions allow for such post-hoc control of attention is provided in §6.A.5.

6.5.2 Robustness to Adversarial Attack

From Table 6.3, the fooling rate of attention-aware VGG is 5% less than the baseline VGG at an ℓ_∞ noise norm of 1. As the noise norm increases, the fooling rate for both the networks grows steadily and their performance gap gradually decreases. Interestingly, when the noise begins to be perceptible (see Figure 6.5, col. 5), the fooling rate of VGG-att2 is around a percentage higher than that of VGG.

L_∞ norm of FGSM attack	Fooling rate VGG	VGG-att2
1	**58.58**	**53.78**
2	78.27	76.00
4	89.99	89.90
8	94.89	95.27
16	96.46	97.60

Table 6.3: Network fooling rate measured as a percentage change in the predicted class labels of the adversarially perturbed images w.r.t those predicted for the unperturbed images.

Figure 6.5: Adversarial versions of a sample image for log-linearly increasing perturbation norm (ℓ_∞) from 1 to 16, for VGG and VGG-att2 trained on CUB-200.

6.5.3 Cross-domain Image Classification

CIFAR images cover a wider variety of natural object categories compared to SVHN and CUB dataset images. Hence, we use these to train different network architectures and use the networks as off-the-shelf feature extractors to evaluate the generalisability of their learned features to new unseen datasets. From Table 6.4, attention-aware models consistently improve over the baseline models, with an average margin of 6%. We make two additional observations. Firstly, low-resolution CIFAR images contain useful visual features that are transferable to high-resolution images such as the 600×600 images of the Event-8 dataset (Li & Fei-Fei, 2007). Secondly, training for diversity is better. CIFAR-10 and CIFAR-100 datasets contain the same corpus of images, only organised differently into 10 and 100 categories respectively. From the results in Table 6.4, and the attention maps of Figure 6.6, it appears that while learning to distinguish a larger set of categories the network is able to highlight more nuanced image regions and capture features that classify new datasets better. (Note that the STL dataset shares the same category set as CIFAR-10. Hence, attention-mounted and baseline models are directly evaluated on STL without the use of an SVM.)

Model	Top-1 accuracies using models trained on CIFAR-10 / CIFAR-100.						
	STL-train	STL-test	Caltech-101	Caltech-256	Event-8	Action-40	Scene-67
| Existing architectures |							
VGG (Simonyan & Zisserman, 2015)	45.34 / –	44.91 / –	35.97 / 54.20	13.16 / 25.57	53.62 / 57.05	13.85 / 17.58	11.02 / 16.73
VGG-GAP (Zhou et al., 2016)	43.24 / –	42.76 / –	41.68 / 62.61	16.30 / 31.39	58.83 / 68.11	16.73 / 24.50	12.85 / 23.04
VGG-PAN (Seo et al., 2016)	47.5 / –	**47.21** / –	48.09 / 65.75	19.61 / 33.66	56.93 / 64.49	16.96 / 23.44	18.77 / 23.43
RN-164 (He et al., 2016)	**47.82** / –	47.02 / –	**49.89 / 73.62**	**22.59 / 39.65**	**69.19 / 75.10**	**20.56 / 28.72**	**20.26 / 29.89**
| Architectures with attention |							
VGG-att2	**48.76** / –	48.29 / –	55.96 / 74.40	26.54 / 41.55	67.73 / 80.24	22.58 / 29.95	25.43 / 30.53
VGG-att3	48.42 / –	**48.32** / –	58.34 / 75.39	29.99 / 44.14	77.06 / 82.08	26.75 / 30.96	26.86 / 33.72
RN-164-att2	46.36 / –	46.45 / –	**68.11 / 79.17**	**36.33 / 46.20**	**80.37 / 83.67**	**30.47 / 31.45**	**30.46 / 34.39**

Table 6.4: Cross-domain classification: Top-1 accuracies using models trained on CIFAR-10/100.

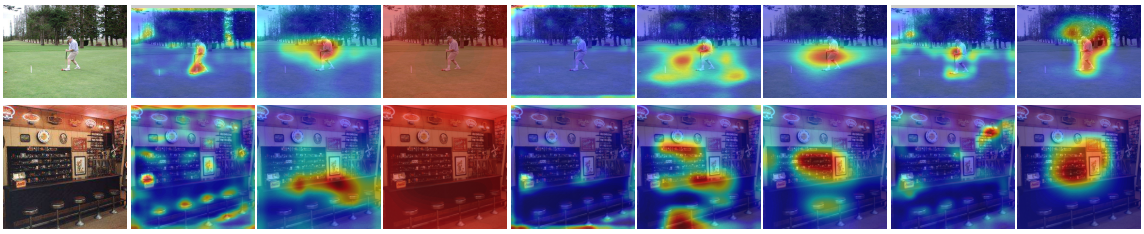


Figure 6.6: Row 1 : Event-8 (croquet), Row 2 : Scene-67 (bar). Attention maps for models trained on CIFAR-100 are more diverse than those from the models trained on CIFAR-10. Note the sharp attention maps in col. 7 versus the uniform ones in col. 4. Attention maps at lower levels appear to attend to part details (*e.g.* the stack of wine bottles in the bar (row 2)) and at a higher level on whole objects owing to a large effective receptive field.

6.5.4 Weakly Supervised Semantic Segmentation

From Table 6.5, the proposed attention maps perform significantly better at weakly supervised segmentation than those obtained using the existing attention methods (Zhou et al., 2016; Seo et al., 2016), outperforming for all three categories by a minimum margin of 11% and 3% respectively. We compare favourably to the non-attentive methods such as the top object proposal method of Arbeláez et al. (2014). Note that we do not compare with the CNN-based object proposal methods as they are trained using additional bounding box annotations. We surpass the saliency-based methods in the car category, but perform less well for the other two categories of airplane and horse. This could be due to the detailed structure and smaller size of objects of the latter two categories, see Figure 6.7. Finally, we perform single-image inference and yet compare well to the joint inference methods that use a group of test images for segmenting the common object category.

Figure 6.7: Weakly supervised segmentation by binarising attention maps.

Models	Airplane	Car	Horse
| Saliency-based |			
Jiang et al. (2013)	37.22	**55.22**	**47.02**
Zhang & Sclaroff (2013)	**51.84**	46.61	39.52
| Top object proposal-based |			
MCG (Arbeláez et al., 2014)	**32.02**	**54.21**	**37.85**
| Joint segmentation-based |			
Joulin et al. (2010)	15.36	37.15	30.16
Object-discovery (Rubinstein et al., 2013)	55.81	64.42	51.65
Chen et al. (2014)	54.62	**69.20**	44.46
Dutt Jain & Grauman (2016)	**58.65**	66.47	**53.57**
| Attention-based |			
VGG-GAP (Zhou et al., 2016)	10.10 / –	19.81 / –	29.55 / –
VGG-PAN (Seo et al., 2016)	31.91 / 31.24	28.21 / 27.45	25.57 / 24.41
VGG-att2	34.98 / 31.45	28.96 / 38.99	29.29 / 29.21
VGG-att3	**48.07** / 35.66	61.19 / 41.23	**40.95** / 29.68
RN-164-att2	41.01 / 45.46	63.12 / **65.05**	36.78 / 40.36

Table 6.5: Jaccard scores (higher is better) for measuring the segmentation quality of binarised attention maps from CIFAR-10/100 trained models tested on the Object Discovery dataset.

6.6 Conclusion

We propose a trainable attention module for generating probabilistic landscapes that highlight where and in what proportion a network attends to different regions of the input image for the task of classification. We demonstrate that the method, when deployed at

multiple levels within a network, affords significant performance gains in classification of seen and unseen categories by focusing on the object of interest. We also show that the attention landscapes can facilitate weakly supervised segmentation of the prominent object. Further, the proposed attention scheme is amenable to popular post-processing techniques such as conditional random fields for refining the segmentation masks, and has shown promise in learning robustness to certain kinds of adversarial attacks.

6.A Appendix

6.A.1 Datasets

We evaluate the proposed attention models on CIFAR-10 (Krizhevsky & Hinton, 2009), CIFAR-100 (Krizhevsky & Hinton, 2009), SVHN (Netzer et al., 2011) and CUB-200-2011 (Wah et al., 2011) for the task of classification. We use the attention-incorporating VGG model trained on CUB-200-2011 for investigating robustness to adversarial attacks. For cross-domain classification, we test on 6 standard benchmarks including STL (Coates et al., 2011), Caltech-256 (Griffin et al., 2007) and Action-40 (Yao et al., 2011). We use the Object Discovery dataset (Rubinstein et al., 2013) for evaluating weakly supervised segmentation performance. A detailed summary of these datasets can be found in Table 6.6.

For all datasets except SVHN and CUB, we perform mean and standard deviation normalisation as well as colour normalisation. For CUB-200-2011, the images are cropped using ground-truth bounding box annotations and resized. For cross-domain image classification we downsample the input images to avoid the memory overhead.

Dataset	Size (total/train/test/extra)	Number of classes / Type	Resolution	Tasks
CIFAR-10 (Krizhevsky & Hinton, 2009)	60,000 / 50,000 / 10,000 / –	10 / natural images	32x32	C, C-c, S
CIFAR-100 (Krizhevsky & Hinton, 2009)	60,000 / 50,000 / 10,000 / –	100 / natural images	32x32	C, C-c, S
SVHN (Netzer et al., 2011)	– / 73,257 / 26,032 / 531,131	10 / digits	32x32	C
CUB-200-2011 (Wah et al., 2011)	– / 5,994 / 5,794 / –	200 / bird images	80x80	C
STL (Coates et al., 2011)	– / 5,000 / 8,000 / –	10 / ImageNet images	96x96	C-c
Caltech-101 (Fei-Fei et al., 2006)	8677 / – / – / –	101 / natural images	∼300x200	C-c
Caltech-256 (Griffin et al., 2007)	29,780/ – / – / –	256 / natural images	∼300x200	C-c
Event-8 (Li & Fei-Fei, 2007)	1574 / – / – / –	8 / in/out door sports	>600x600	C-c
Action-40 (Yao et al., 2011)	9532/ – / – / –	40 / natural images	∼400x300	C-c
Scene-67 (Quattoni & Torralba, 2009)	15613 / – / – / –	67 / indoor scenes	∼500x300	C-c
Object Discovery (Rubinstein et al., 2013)	300/ – / 300 / –	3 / synthetic and natural images	∼340x240	C-c, S

Table 6.6: Summary of datasets used for experiments across different tasks (C: classification, C-c: classification cross-domain, S: segmentation).The natural images span from those of objects in a plain background to images capturing cluttered indoor and outdoor scenes. The objects vary from simple digits to humans involved in complex activities.

6.A.2 Network Architectures

Progressive Attention Networks: We experiment with the progressive attention mechanism proposed by Seo et al. (2016) as part of our 2-level attention-based VGG models. The attention at the lower level (layer-10) is implemented using a 2D map of compatibility scores, obtained using the parameterised compatibility function discussed in §6.3.2. Note

that at this level, the compatibility scores are not jointly normalised using the softmax operation, but are normalised independently using the pointwise sigmoid function. These scores, at each spatial location, are used to scale the corresponding local features before the feature vectors are fed to the next network layer. This is the filtering operation that is at the core of the progressive attention scheme proposed by Seo et al. (2016). For the final level, attention is implemented in the same way as in *VGG-att2-concat-pc*. The compatibility scores are normalised using a softmax operation and the local features, added in proportion to the normalised attention scores, are trained for image classification.

We implement and evaluate the above discussed progressive attention approach as well as the proposed attention mechanism with the VGG architecture using the codebase provided here: https://github.com/szagoruyko/cifar.torch. The code for CIFAR dataset normalisation is included in the repository.

Attention in Residual Networks: We make the attention related modifications to the architecture - https://github.com/szagoruyko/wide-residual-networks/tree/fp16/models. The baseline ResNet implementation consists of 4 distinct levels that project the RGB input onto a 256-dimensional space through 16-, 64-, and 128-dimensional embedding spaces respectively. Each level excepting the first, which contains 2 convolutional layers separated by a non-linearity, contains n-residual blocks. Each residual block in turn contains a maximum of 3 convolutional layers interleaved with non-linearities. This yields a network definition of $9n + 2$ parameterised layers (He et al., 2016). We work with an n of 18 for a 164-layered network. Batch normalisation is incorporated in a manner similar to other contemporary networks.

We replace the spatial average pooling layer after the final (4^{th}) level by convolutional and max-pooling operations which gives us our global feature vector $\boldsymbol{g}$. We refer to the network implementing attention at the output of the last level only as RN-att and with attention at the output of last two levels as RN-att2. Following the results obtained on the VGG network, we train the attention units in the *concat-pc* framework.

6.A.3 Training Routines

VGG networks for CIFAR-10, CIFAR-100 and SVHN are trained from scratch. We use a stochastic gradient descent (SGD) optimiser with a batch size of 128, learning rate decay of 10^{-7}, weight decay of 5×10^{-4}, and momentum of 0.9. The initial learning rate for CIFAR experiments is 1 and for SVHN is 0.1. The learning rate is scaled by 0.5 every 25 epochs and we train over 300 epochs for convergence. For CUB, since the training data is limited, we initialise the model with the weights learned for CIFAR-100. We use the transfer-learning training schedule inspired by Redmon et al. (2016). Thus the training starts at a learning rate of 0.1 for first 30 epochs, is multiplied by 2 twice over the next 60 epochs, and then scaled by 0.5 every 30 epochs for the next 200 epochs.

For ResNet, the networks are trained using an SGD optimiser with a batch size of 64, initial learning rate of 0.1, weight decay of 5×10^{-4}, and a momentum of 0.9. The learning rate is multiplied by 0.2 after 60, 120 and 160 epochs. The network is trained for 200 epochs until convergence. We train the models for CIFAR-10 and CIFAR-100 from scratch.

All models are implemented in Torch and trained using an NVIDIA Titan-X GPU. Training takes around one to two days depending on the model architecture and the dataset in use.

6.A.4 Task-specific Processing

We generate the adversarial images using the fast gradient sign method of Goodfellow et al. (2015) and observe the network fooling behaviour at an increasing L_∞ norm of the perturbations.

For cross-domain classification, we extract features at the input of the final fully connected layer of each model, use these to train a linear SVM with $C = 1$ and report the results of a 5-fold cross validation, similar to the setup used by Zhou et al. (2016). At no point do we fine-tune the networks on the target datasets. We perform ZCA whitening on all the evaluation datasets using the pre-processing Python scripts specified in the following for whitening CIFAR datasets: https://github.com/szagoruyko/wide-residual-networks/tree/fp16.

For weakly supervised segmentation, the evaluation datasets are preprocessed for colour normalisation using the same scheme as adopted for normalising the training datasets of the respective models. For the proposed attention mechanism, we combine the attention maps from the last 2 levels using element-wise multiplication, take the square root of the result to re-interpret it as a probability distribution (in the limit of the two probabilistic attention distributions approaching each other), rescale the values in the resultant map to a range of $[0, 1]$ and binarise the map using the Otsu binarisation threshold. For the progressive attention mechanism of Seo et al. (2016), we simply multiply the attention maps from the two different levels without taking their square root, given that these attention maps are not complementary but sequential maps used to fully develop the final attention distribution. The rest of the operations of magnitude rescaling and binarisation are commonly applied to all the final attention maps, including those obtained using GAP (Zhou et al., 2016).

6.A.5 Query Driven Attention Patterns

In our framework, the global feature vector is used as a query for estimating attention on local image regions. Thus, by changing the global feature vector, one could expect to also affect the attention distribution estimated over fixed local regions. The extent to which the two different compatibility functions, the parameterised and the dot-product based compatibility functions, allow for such external control over the estimated attention patterns may be different.

For the purpose of our analysis, we consider two different network architectures. The first is the *(VGG-att2)-concat-dp* model from Table 6.1 which uses a dot-product (DP) based compatibility function. The other is the *(VGG-att3)-concat-pc* model which makes use of the parameterised compatibility (PC) function. In terms of the dataset, we make use of the extra cosegmentation image dataset available with the Object Discovery dataset package. For each object category we select a single image, centrally focused on an instance of the given object category, and call this the query image. We then gather a few distinct images that contain objects of the same category but in a more cluttered environment with pose and intra-class variations. We call these the target images.

In order to visualise the role of the global feature vector in driving the estimated attention patterns, we perform two rounds of experiments. In the first round, we obtain both the global and local image feature vectors from the target images, shown in column 2 of every row of Figure 6.8. The processing follows the standard procedure and the resulting attention patterns at layer 10 for the two architectures can be seen in columns 3 and 6 of the same figure. In the second round, we obtain the local feature vectors from the target images but the global feature vector is obtained by processing the query image specific to the category being considered, shown in column 1. The new attention patterns are displayed in columns 4 and 7 respectively. The changes in the attention values at different spatial locations as a proportion of the original attention scores at those locations are shown in columns 5 and 8 respectively.

Notably, for the dot-product based attention mechanism, the global vector plays a prominent role in guiding attention. This is visible in the increase in the attention magnitudes at the spatial locations containing image content that is related to the query image object. On the other hand, for the attention mechanism that makes use of the parameterised compatibility function, the global feature vector seems to be redundant. Any change in the global feature vector does not transfer to the resulting attention map. In fact, for this compatibility function, the numerical observations show that the magnitudes of the global features are often a couple of orders of magnitude smaller than those of the corresponding local features. Thus, a change in the global feature vector has little to no impact on the predicted attention scores. Yet, the attention maps are able to consistently highlight object-relevant image regions. Thus, it appears that in the case of parameterised compatibility based attention, the object-centric semantic features are learned as part of the weight vector $\boldsymbol{u}$. These features are adapted to the training dataset and are able to generalise to new images inasmuch as the object categories at test time are similar to those seen during training.

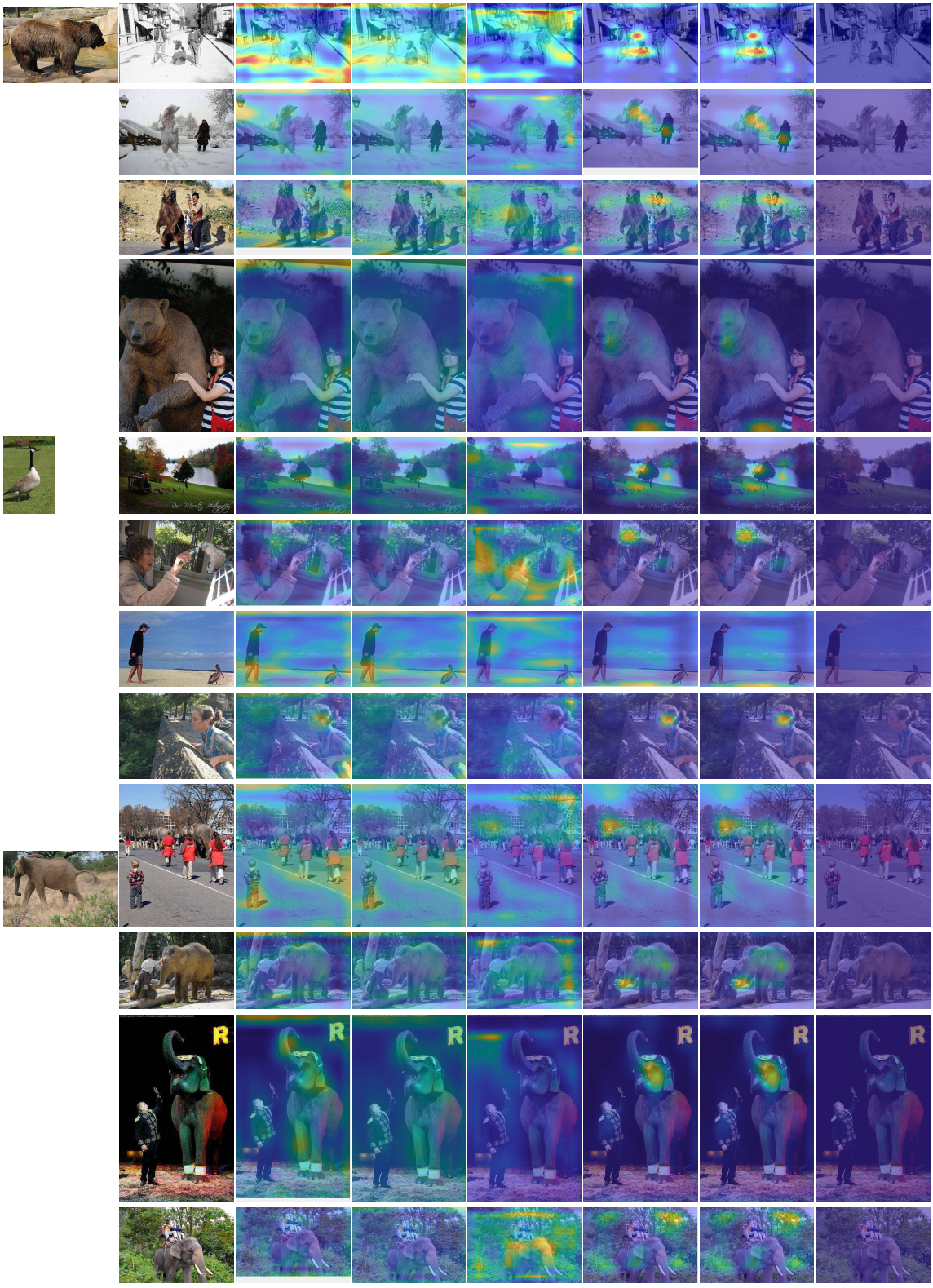


Figure 6.8: Visual analysis of how a global feature vector obtained from a query image affects the attention patterns on the local image regions of another distinct target image.

7

With Friends Like These, Who Needs Adversaries?

Contents

The vulnerability of deep image classification networks to adversarial attack is now well known, but less well understood. Via a novel experimental analysis, we illustrate some facts about convolutional neural networks for image classification that shed new light on their behaviour and how it connects to the problem of adversaries. In short, the

celebrated performance of these networks and their vulnerability to adversarial attack are simply two sides of the same coin: the input image-space directions along which the networks are most vulnerable to attack are the same directions which they use to achieve their classification performance in the first place. We develop this result in two main steps. The first uncovers the fact that classes tend to be associated with specific image-space directions. This is shown by an examination of the class-score outputs of nets as functions of 1D movements along these directions. This provides a novel perspective on the existence of universal adversarial perturbations. The second is a clear demonstration of the tight coupling between classification performance and vulnerability to adversarial attack within the spaces spanned by these directions. Thus, our analysis resolves the apparent contradiction between accuracy and vulnerability. It provides a new perspective on much of the prior art and reveals profound implications for efforts to construct neural nets that are both accurate and robust to adversarial attack.*

7.1 Introduction

Those studying deep networks find themselves forced to confront an apparent paradox. On the one hand, there is the demonstrated success of networks in learning class distinctions on training sets that seem to generalise well to unseen test data. On the other, there is the vulnerability of the very same networks to adversarial perturbations that produce dramatic changes in class predictions despite being counter-intuitive or even imperceptible to humans. A common understanding of the issue can be stated as follows: "While deep networks have proven their ability to distinguish between their target classes so as to generalise over unseen natural variations, they curiously possess an Achilles heel which must be defended." In fact, efforts to formulate attacks and counteracting defences of networks have led to a dedicated competition (NIPS, 2017) and a body of literature already too vast to summarise in total.

*Source code is available at https://github.com/torrvision/whoneedsadversaries. (Elsewhere, * with authors' names represents work with joint first-authorship assertion.)

In the current work we attempt to demystify this phenomenon at a fundamental level. We base our work on the geometric decision boundary analysis approach of Moosavi-Dezfooli* et al. (2018), which we reinterpret and extend into a framework that we believe is simpler and more illuminating with regards to the aforementioned paradoxical behaviour of deep convolutional networks (CNNs) for image classification. Through a fairly straightforward set of experiments and explanations, we clarify what it is that adversarial examples represent, and indeed, what it is that modern CNNs do and do not currently do. In doing so, we tie together work which has focused on adversaries *per se* with other work which has sought to characterise the feature spaces learned by these networks.

Let $\check{\mathbf{i}}$ represent vectorised input images and $\bar{\mathbf{i}}$ be the average vector-image over a given dataset. Then, the mean-normalised version of the dataset is denoted by $\mathbb{I} = \{\mathbf{i}_1, \mathbf{i}_2, \cdots \mathbf{i}_N\}$, where the n^{th} image $\mathbf{i}_n = \check{\mathbf{i}}_n - \bar{\mathbf{i}}$. We define the perturbation of the image $\mathbf{i}_n$ in the direction $\mathbf{d}_j$ as: $\tilde{\mathbf{i}}_n \leftarrow \mathbf{i}_n + s\hat{\mathbf{d}}_j$, where s is the perturbation scaling factor and $\hat{\mathbf{d}}_j$ is the unit-norm vector in the direction $\mathbf{d}_j$. The image is fed through a network parameterised by θ and the output score* for a specific class c is given by $\mathcal{F}_c(\tilde{\mathbf{i}}|\theta)$. This class-score function can be rewritten as $\mathcal{F}_c(\mathbf{i}_n + s\hat{\mathbf{d}}_j|\theta)$, which we equivalently denote by $\tilde{\mathcal{F}}_c(s|\mathbf{i}_n, \mathbf{d}_j, \theta)$. Our work examines the nature of $\tilde{\mathcal{F}}_c$ as a function of movement s in specific image-space directions $\mathbf{d}_j$ starting from randomly sampled natural images $\mathbf{i}_n$, for a variety of classification CNNs. With this novel analysis, we uncover three noteworthy observations about these functions that relate directly to the phenomenon of adversarial vulnerability in these nets, all of which are on display in Figure 7.1. We now discuss these observations in more detail.

Before we begin, we note that these directions $\mathbf{d}_j$ are obtained via the method explained in §7.3 and by design exhibit either positive or negative association with a specific class. In Figure 7.1 we study two such directions for the 'frog' class for the Network-in-Network (Lin et al., 2013) architecture trained on CIFAR10 dataset: similar directions exist for all other classes. Firstly, notice that the score of the corresponding class c ('frog', in this case) as a function of s is often approximately symmetrical about

*The output of the layer just before the softmax operation, commonly known as the logit layer.

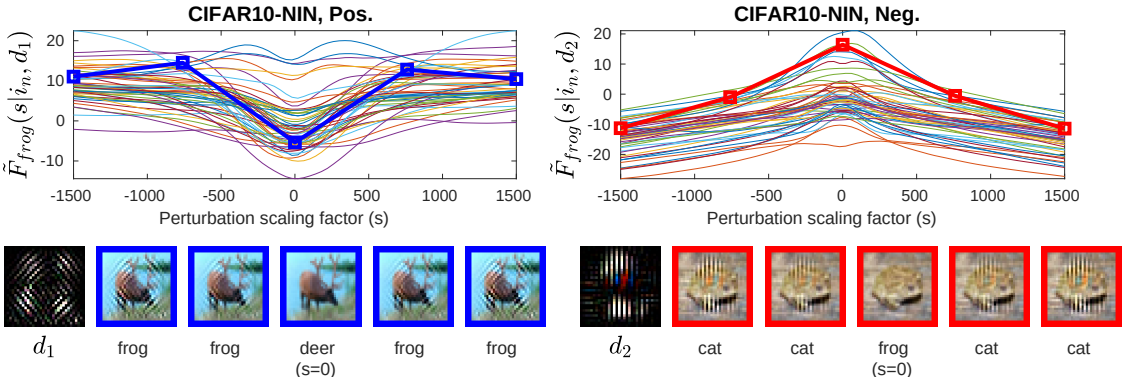


Figure 7.1: Plots of the 'frog' class score $\tilde{\mathcal{F}}_{frog}(s|\mathbf{i}_n, \mathbf{d}_j, \theta)$ for the Network-in-Network (Lin et al., 2013) architecture trained on CIFAR10, associated with two specific image-space directions $\mathbf{d}_1$ and $\mathbf{d}_2$ respectively. These directions are visualised as 2D images in the row below; the method of estimating them is explained in §7.3. Each plot corresponds to a randomly selected CIFAR10 test image $\mathbf{i}_n$. Adding or subtracting components along $\mathbf{d}_1$ causes the network to change its prediction to 'frog': as can be seen, a 'deer' with a mild diamond striping added to it gets classified as a 'frog'. This happens with little regard for the choice of input image $\mathbf{i}_n$ itself. Likewise, perturbations along $\mathbf{d}_2$ change *any* 'frog' to a 'non-frog' class: notice the predicted labels for the sample images along the red curve in the second plot. These class-transition phenomena are predicted by the framework developed in this work. While simplistic functions along directions $\mathbf{d}_1$ and $\mathbf{d}_2$ are used by the network to accomplish the task of classification, perturbations along the very same directions constitute adversarial attacks.

some point s_0, *i.e.* $\tilde{\mathcal{F}}_c(s + s_0|\mathbf{i}_n, \mathbf{d}_j, \theta) \approx \tilde{\mathcal{F}}_c(-s + s_0|\mathbf{i}_n, \mathbf{d}_j, \theta)\ \forall s$, and monotonic in both half-lines. This means that simply increasing the magnitude of correlation between the input image and a single direction causes the net to believe that more (or less) of the class c is present. In other words, the image-space direction sends all images either towards or away from the class c. In the former scenario, the direction represents a class-specific universal adversarial perturbation (UAP). Second, let $i^{\mathbf{d}} = \mathbf{i} \cdot \hat{\mathbf{d}}$, and let $\mathbf{i}^{\mathbf{d}\perp}$ be the projection of $\mathbf{i}$ onto the space normal to $\hat{\mathbf{d}}$, such that $\mathbf{i}^{\mathbf{d}\perp} = \mathbf{i} - i^{\mathbf{d}}\hat{\mathbf{d}}$. Then, our results illustrate that there exists a basis of image space containing $\hat{\mathbf{d}}$ as one of the basis vectors such that the class-score function is approximately additively separable *i.e.* $\mathcal{F}_c(\mathbf{i}|\theta) = \mathcal{F}_c([i^{\mathbf{d}}, \mathbf{i}^{\mathbf{d}\perp}]|\theta) \approx \mathcal{G}(i^{\mathbf{d}}) + \mathcal{H}(\mathbf{i}^{\mathbf{d}\perp})$ for some functions $\mathcal{G}$ and $\mathcal{H}$. This means that the directions under study can be used to alter the nets' predictions almost independently of each other. However, despite these facts, their 2D visualisation reveals low-level structures that are devoid of a clear semantic link to the associated classes, as shown in Figure 7.1. Thus, we demonstrate that the learned functions encode a more simplistic notion of class identity than CNNs are commonly assumed to represent, albeit

one that generalises to the test distribution to an extent. Unsurprisingly, this does not align with the way in which the human visual system makes use of these data dimensions: 'adversarial vulnerability' is simply the name given to this disparity and the phenomena derived from it, with the universal adversarial perturbations of Moosavi-Dezfooli* et al. (2017) being a particularly direct example of this fact.

Finally, we show for a variety of network architectures that the classification performance and adversarial vulnerability of these nets are inextricably linked by virtue of the manner in which they make use of the above directions. Consequently, efforts to improve robustness by "suppressing" nets' responses to components in these directions (*e.g.* Gao et al. (2017a)) cannot simultaneously retain full classification accuracy. The features and functions thereof that CNNs currently rely on to solve the classification problem *are*, in a sense, their own worst adversaries.

7.2 Related Work

7.2.1 Fundamental Developments in Attack Methods

Szegedy et al. (2014) coined the term 'adversarial example', demonstrating the use of box-constrained L-BFGS to estimate a minimal ℓ_2-norm additive perturbation to an input image to cause its label to change to a target class while keeping the resulting image within intensity bounds. Strikingly, they locate a small-norm (imperceptible) perturbation at every point, for every network tested. Further, the adversaries thus generated are able to fool nets trained differently to one another, even when trained with different subsets of the data. Goodfellow et al. (2015) subsequently proposed the 'fast gradient sign method' to demonstrate the effectiveness of the local linearity assumption in producing the same result, calculating the gradient of the cost function and perturbing with a fixed-size step in the direction of its sign (optimal under the linearity assumption and an ℓ_∞-norm constraint). The DeepFool method of Moosavi-Dezfooli et al. (2016) retains the first-order framework of FGSM, but tailors itself precisely to the goal of finding the perturbation of *minimum* norm that changes the class label of a given natural image to any label other than its own. Through iterative attempts to cross the nearest (linear)

decision boundary by a tiny margin, this method records successful perturbations with norms that are even smaller than those of Goodfellow et al. (2015). Moosavi-Dezfooli* et al. (2017) propose an iterative aggregation of DeepFool perturbations that produces "universal" adversarial perturbations: *single* images which function as adversaries over a large fraction of an *entire* dataset for a targeted net. While these perturbations are typically much larger than individual DeepFools, they do not correspond to human perception, and indicate that there are *fixed* image-space directions along which nets are vulnerable to deception *independently* of the image-space locations at which they are applied. They also demonstrate some generalisation over network architectures.

Sabour* et al. (2016) pose an interesting variant of the problem: instead of "label adversaries", they target "feature adversaries" which minimise the distance from a particular guide image in a selected network feature space, subject to a constraint on the ℓ_∞-norm of image-space distance from a source image. Despite this constraint, the adversarial image mimics the guide very closely: not only is it nearly always assigned to the guide's class, but it appears to be an inlier with respect to the guide-class distribution in the chosen feature space. Finally, while "adversaries" are conceived of as small perturbations applied to natural images such that the resulting images are still recognisable to humans, the "fooling images" of Nguyen et al. (2015) are completely unrecognisable to humans and yet confidently predicted by deep networks to be of particular classes. Such images are easily obtained by both evolutionary algorithms and gradient ascent, under direct encoding of pixel intensities (appearing to consist mostly of noise) and under CPPN (Stanley, 2007) regularised encoding (appearing as abstract mid-level patterns).

7.2.2 Analysis of Adversarial Vulnerability and Proposed Defences

Wang et al. (2016) propose a nomenclature and theoretical framework with which to discuss the problem of adversarial vulnerability in the abstract, agnostic of any actual net or attack thereof. They denote an oracle relative to whose judgement robustness and accuracy must be assessed, and illustrate that a classifier can only be both accurate and robust (invulnerable to attack) relative to its oracle if it learns to use exactly the same feature space that the oracle does. Otherwise, a network is vulnerable to adversarial attack

in precisely the directions in which its feature space departs from that of the oracle. Under the assumption that a net's feature space contains some spurious directions, Gao et al. (2017a) propose a subtractive scheme of achieving adversarial robustness by suppressing the neuronal activations (*i.e.* feature responses) that change significantly between the natural and adversarial inputs. Notably, any robustness obtained using this scheme is accompanied by a corresponding loss of performance accuracy. Similar to network feature suppression, some works have explored the suppression of dimensions in the input image space *e.g.* Maharaj (2015); Das et al. (2017); Xie et al. (2017).

Goodfellow et al. (2015) hypothesise that the high dimensionality and excessive linearity of deep networks explain their vulnerability. Tanay & Griffin (2016) begin by taking issue with the above via illustrative toy problems. They then advance an explanation based on the angle of intersection of the separating boundary with the data manifold which rests on overfitting and calls for effective regularisation - which they note is neither solved nor known to be solvable for deep nets. A variety of training-based methods such as Zhang et al. (2018); Goodfellow et al. (2015); Madry et al. (2018); Lu et al. (2017); Metzen et al. (2017) have been proposed to address the premise of the preceding analyses. In the data augmentation scheme of Zhang et al. (2018), convex combinations of input images are mapped to convex combinations of their labels in an attempt to enable the nets to learn smoother decision boundaries. While their approach offers improved resistance to single-step gradient sign attacks, it is no more robust to iterative attacks of the same type. Hardening methods of Goodfellow et al. (2015); Madry et al. (2018) investigate the use of adversarial examples to train robust deep networks. Detection-based methods (Lu et al., 2017; Metzen et al., 2017) view adversaries as outliers to the training data distribution and train detectors to identify them as such in the intermediate feature spaces of nets. Notably, these methods (Lu et al., 2017; Metzen et al., 2017) have not been evaluated on the feature adversaries of Sabour* et al. (2016).

Over the course of the line of work in Moosavi-Dezfooli* et al. (2018), Fawzi* et al. (2016), Fawzi* et al. (2017), and Fawzi et al. (2017), the authors build up an image-space analysis of the geometry of deep networks' decision boundaries, and its connection with adversarial vulnerability. Fawzi* et al. (2017) note that the DeepFool perturbations of

Moosavi-Dezfooli et al. (2016) tend to evince relatively high components in the subspace spanned by the directions in which the decision boundary has a high curvature. Also, the sign of the mean curvature of the decision boundary in the vicinity of DeepFooled images is typically reversed with respect to that of the natural images, which provides a simple scheme to identify and undo the attack. They conclude that a majority of image-space directions correspond to near-flatness of the decision boundary and are insensitive to attack, but along the remaining directions, those of significant curvature, the network is indeed vulnerable. Further, the directions in question are observed to be *shared* over sample images. Moosavi-Dezfooli* et al. (2018) illustrate why a hypothetical network which possessed this property would theoretically be *predicted* to be vulnerable to universal adversaries, and note that the analysis suggests a direct construction method for such universal adversaries as an alternative to the original randomised iterative approach of Moosavi-Dezfooli* et al. (2017): they can be constructed as random vectors in the subspace of shared high-curvature dimensions.

7.3 Method

The analysis begins as in Moosavi-Dezfooli* et al. (2018), with the extraction of the principal directions and principal curvatures of the classifier's image-space class decision boundaries. Put simply, a principal direction vector and its associated principal curvature tell you how much a surface curves as you move along it in a particular direction, from a particular point. Now, it takes many decision boundaries to characterise the classification behaviour of a multiclass net: $\binom{C}{2}$ for a C-class classifier. However, in order to understand the boundary properties that are useful for discriminating a given class from all others, it should suffice to analyse only the C 1-vs.-all decision boundaries. Thus, for each class c, the method proceeds by locating samples very near to the decision boundary $(\mathcal{F}_c - \mathcal{F}_{\hat{c}}) = 0$ between c and the union of remaining classes $\hat{c} \neq c$. In practice, this corresponds to perturbing the samples from class $\hat{c} \neq c$ (target) to the class c (source), and stopping at the decision boundary between the two. Then, the geometry of the

decision boundary is estimated as outlined in the Alg. 1 below*, closely following the approach of Moosavi-Dezfooli* et al. (2018):

Algorithm 1 Computes mean principal directions and principal curvatures for a net's image-space decision surface.

Input: network class score function $\mathcal{F}$, dataset $\mathbb{I} = \{\mathbf{i}_1, \mathbf{i}_2, \cdots \mathbf{i}_N\}$, target class label c
Output: principal curvature basis matrix $\mathbf{V}_b$ and corresponding principal curvature-score vector $\mathbf{v}_s$

```
procedure PRINCIPALCURVATURES(F, I, c)
    H̄ ← null
    for each sample i_n ∈ I s.t. argmax_k(F_k(i_n)) ≠ c do
        ĉ ← argmax_k(F_k(i_n))                    ▷ network predicts i_n to be of class ĉ
        H_cĉ: define as Hessian of function (F_c − F_ĉ)    ▷ subscripts select class scores
        ĩ_n ← DEEPFOOL(i_n, c)                    ▷ approximate nearest boundary point to i_n
        H̄ ← H̄ + H_cĉ(ĩ_n)                         ▷ accumulate Hessian at sample boundary point
    H̄ ← H̄/‖I‖                                    ▷ normalise mean Hessian by number of samples
    (V_b, v_s) = EIGS(H̄)                          ▷ compute eigenvectors and eigenvalues of mean Hessian
    return (V_b, v_s)
```

Moosavi-Dezfooli* et al. (2018) advance a hypothesis connecting positively curved directions with the universal adversarial perturbations of Moosavi-Dezfooli* et al. (2017). Essentially, they demonstrate that if the normal section of a net's decision surface along a given direction can be locally bounded on the outside by a circular arc of a particular positive curvature in the vicinity of a sample image point, then geometry accordingly dictates an upper bound on the distance between that point and the boundary in that direction. If such directions and bounds turn out to be largely common across sample image points (which they do), then the existence of universal adversaries follows directly, with higher curvature implying lower-norm adversaries. This argument is depicted visually in the leftmost subfigure of Figure 7.8 in supplementary §7.A.1. It is from this point that we move beyond the prior art and begin an iterative loop of further analysis, experimentation, and demonstration, as follows.

7.4 Experiments and Analysis

Provided only that the second-order boundary approximation holds up well over a sufficiently wide perturbation range and variety of images, the model implies that the distance of such adversaries from the decision boundary should increase as a function of their norm. Also, the attack along any positively curved direction should in that case be

*For more discussion about the implementation and associated concepts, refer to the supplementary material.

associated with the corresponding target class: the class c in the call to Alg. 1. And while positively curved directions may be of primary interest in Moosavi-Dezfooli* et al. (2018), the extension of the above geometric argument to the negative-curvature case points to an important corollary: as sufficient steps along positive-curvature directions should perturb increasingly into class c, so should steps along *negative*-curvature directions perturb increasingly *away* from class c. Finally, the plethora of approximately zero-curvature (flat) directions identified in Fawzi* et al. (2017); Moosavi-Dezfooli* et al. (2018) should have negligible effect on class identity.

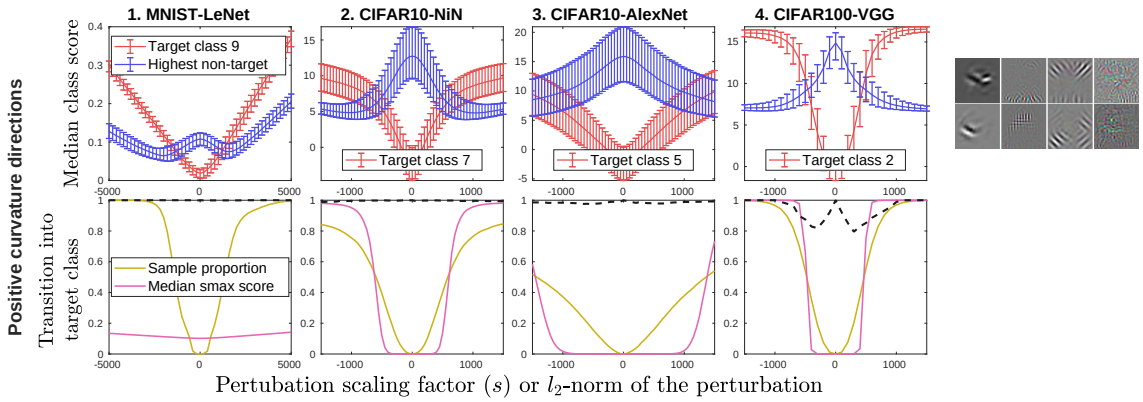


Figure 7.2: Selected class scores plotted as functions of the scaling factor s of the perturbation along the most positively curved direction per net. The 'Median class score' plot compares the score of a randomly selected target class with the supremum of the scores for the non-target classes. Each curve represents the median of the class scores over the associated dataset, bracketed below by the 30th-percentile score and above by the 70th. The 'Transition into target class' plot depicts the fraction of the dataset *not* originally of the target class, but which is transitioned into the target class by the perturbation. Alongside, we graph that population's median softmax target-class score. The black dashed line represents the fraction of the population originally of the target class that remains in the target class under the perturbation. The image grid on the right illustrates the 2D visualisations of the two most-positively curved directions for randomly selected target classes: the columns correspond, from left to right, with the four net-dataset pairs under study.

7.4.1 Class Identity as Function of Component in Specific Image-Space Directions

To test how well the above conjectures hold in practice, we graph statistics of the target and non-target class scores over the dataset as a function of the magnitude of the perturbation applied in directions identified as above. The results are depicted in Figures 7.2 and 7.3, in which the predicted phenomena are readily evident. Along the selected positive-curvature

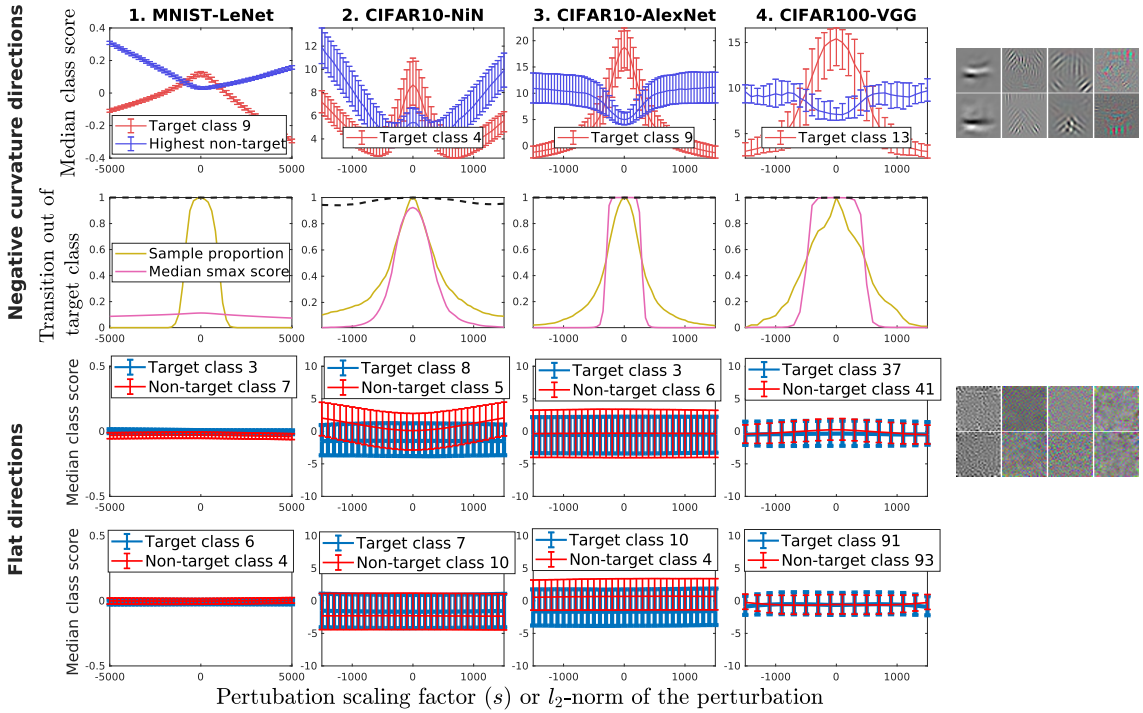


Figure 7.3: Selected class scores plotted as functions of the scaling factor s of the perturbation along the most negatively curved directions and flat directions per net. The 'Median class score' plot compares the score of a randomly selected target class with the supremum of the scores for the non-target classes, for the negatively curved directions. For the flat curvature directions, it plots the score of a randomly selected target class and non-target class respectively. Each curve represents the median of the class scores over the associated dataset, bracketed below by the 30th-percentile score and above by the 70th. For the negative-curvature directions, the 'Transition out of target class' graph works in reverse to the corresponding positive-curvature graph of Figure 7.2: 'sample proportion' represents the fraction of the dataset originally of the target class which retains the target-class label under perturbation, with the median softmax target-class score as before. The 'black dashed line' now represents the fraction of the dataset *not* originally of the target class which remains *outside* of the target class under perturbation. The images in the rightmost column illustrate a sample of these directions as visual patterns. Each block of eight images corresponds to the label (negative, or flat) to its left, and the two-image columns in each block correspond from left to right with the main four net-dataset pairs under study.

directions in Figure 7.2, as the perturbation magnitude increases (with either sign), the population's target class score approaches and then surpasses the highest non-target class score. The monotonicity of this effect is laid bare by graphing the fraction of non-target samples perturbed into the target class, alongside the median target class softmax score. Note, again, that the link between the directions in question and the target class identity is established *a priori* by Alg. 1. We continue in Figure 7.3 and show that, as predicted, the same phenomenon is evident in reverse when using negative-curvature directions

instead. All that changes is that it is the population's non-target class scores that overtake its target class score with increasing perturbation norm, with natural samples of the target class accordingly being perturbed out of it.

We also illustrate the point that flatness of the decision boundary manifests as flatness of both target and non-target class scores: over a wide range of magnitudes, these directions do not influence the network in any way. While Figures 7.2 and 7.3 illustrate these effects at the level of the population, Figure 7.1 shows a disaggregation into individual sample images, with one response curve per sample from a large set. The population-level trends remain evident, but another fact becomes apparent: empirically, the shapes of the curves change very little between most samples. They shift vertically to reflect the class score contribution of the orthonormal components, but they themselves do not otherwise much depend on those components. That is to say that at least some key components are approximately additively separable from one another. This fact connects directly to the fact that such directions are "shared" across samples in the first place, and thus identifiable by Alg. 1.

Figure 7.4: Visualisations of image space directions that are associated with fixed class (or non-class) identities as mentioned in the row below, for Network-in-Network (Lin et al., 2013) architecture trained on CIFAR-10 dataset. The image space directions that take the samples into a given class are that class's positive curvature directions, while those that take the samples away from a given class are the negative curvature directions associated with that class.

A more intuitive picture of what the networks are actually doing begins to emerge: they are identifying the high-curvature image-space directions as features associated with respective class identities, with the curvature magnitude representing the sensitivity of class identity to the presence of that feature. But if this is true, it suggests that what we have thus identified are actually the directions which the net relies on *generally* in predicting the classes of natural images, with the curvatures-cum-sensitivities representing

their relative weightings. Accordingly, it should be possible to disregard the "flat" directions of near-zero curvature without any noticeable change in the network's class predictions. While we perform experiments to explore this hypothesis in the forthcoming sections, we take some time here to visualise sample positive and negative curvature directions associated with CIFAR-10 classes, see Figure 7.4. These directions do exhibit some visual structure, however to a human eye, their overall appearance has little semantic link with the classes that they help to identify.

7.4.2 Network Classification Performance versus Effective Data Dimensionality

To confirm the above hypothesis regarding the relative importance of different image-space directions for classification, we plot the training and test accuracies of a sample of nets as a function of the subspace onto which their input images are projected. The input subspace is parametrised by a dimensionality parameter d, which controls the number of basis vectors selected per class. We use four variants of selection: the d most positively curved directions per class (yielding the subspace S_{pos}); the d most negatively curved directions per class (yielding the subspace S_{neg}); the union of the previous two (subspace $S_{neg \cup pos}$); and the d least curved (flattest) directions per class (subspace S_{flat}). The subspace S so obtained is represented by the orthonormalised basis matrix $\mathbf{Q}_d$ (obtained by QR decomposition of the aggregated directions), and each input image $\mathbf{i}$ is then projected* onto S as $\mathbf{i}^d = \mathbf{Q}_d\mathbf{Q}_d^\top\mathbf{i}$. Accuracies on $\{\mathbf{i}^d\}$ as a function of d are shown in the top row of Figure 7.5.

The outcome is striking: it is evident that in many cases, classification decisions have effectively already been made based on a relatively small number of features, corresponding to the most curved directions. The sensitivity of the nets along these directions, then, is clearly learned purposefully from the training data, and does largely generalise in testing, as seen. Note also that at this level of analysis, it essentially does not matter whether positively or negatively curved directions are chosen. Another important point emerges here. Since it is the high-curvature directions that are largely responsible for

*The mean training-set orthogonal component $(\mathbf{I} - \mathbf{Q}_d\mathbf{Q}_d^\top)\bar{\mathbf{i}}$ can be added, but is approximately 0 in practice for data normalised by mean subtraction, as is the case here.

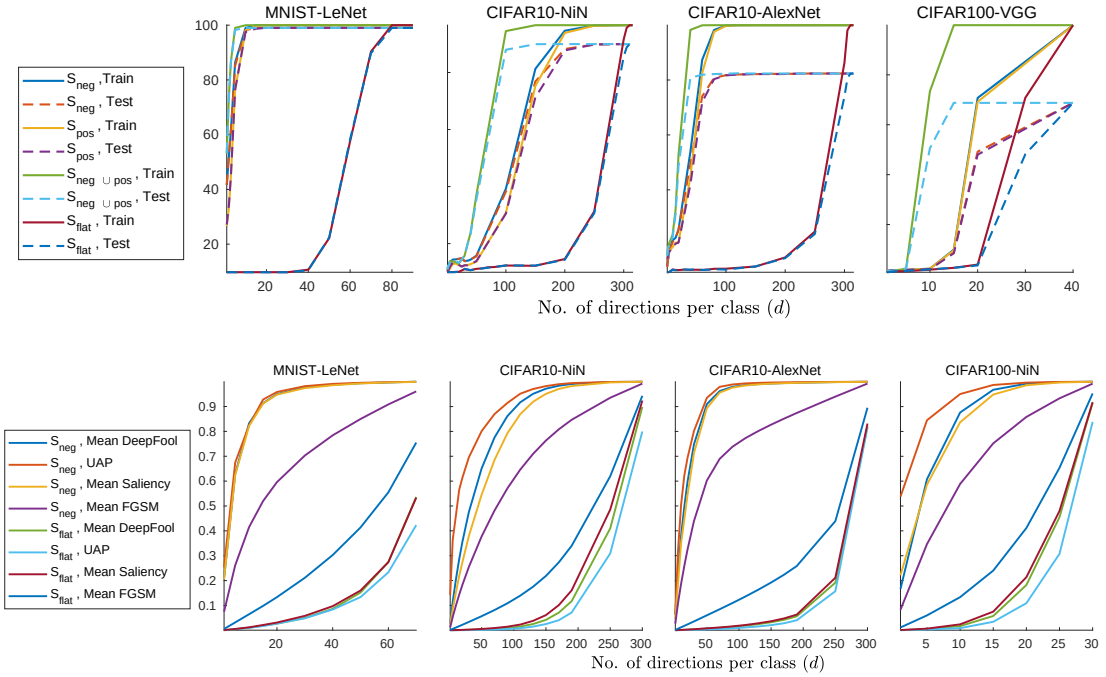


Figure 7.5: *Top row:* Training and test classification accuracies for various CNNs on image sets projected onto the subspaces described in Sec. 7.4.2, as a function of their dimensionality parameter d (from 0 until the input space is fully spanned). The principal directions defining the subspaces are obtained by applying Alg. 1 once for each possible choice of target class c and retaining d directions per class. Note the relationship between the ordering of curvature magnitudes and classification accuracy by comparing the S_{flat} curves to the others. *Bottom row:* Mean ℓ_2-norms of various adversarial perturbations (DeepFool (Moosavi-Dezfooli et al., 2016), FGSM (Goodfellow et al., 2015) and UAP (Moosavi-Dezfooli* et al., 2017)) and saliency maps (Simonyan et al., 2013) when projected onto the same subspaces as above, as a fraction of their original norms.

determining the nets' classification decisions, the nets should be vulnerable to adversarial attack along *precisely these directions*.

7.4.3 Link Between Classification and Adversarial Directions

It has already been noted in Fawzi* et al. (2017) that adversarial attack vectors evince high components in subspaces spanned by high-curvature directions. We expand the analysis by repeating the procedure of Sec. 7.4.2 for various attack methods, to determine whether existing attacks are indeed exploiting the directions in accordance with the classifier's reliance on them. Results are displayed in the bottom row of Figure 7.5, and should be compared against the row above. The graphs in these figures illustrate

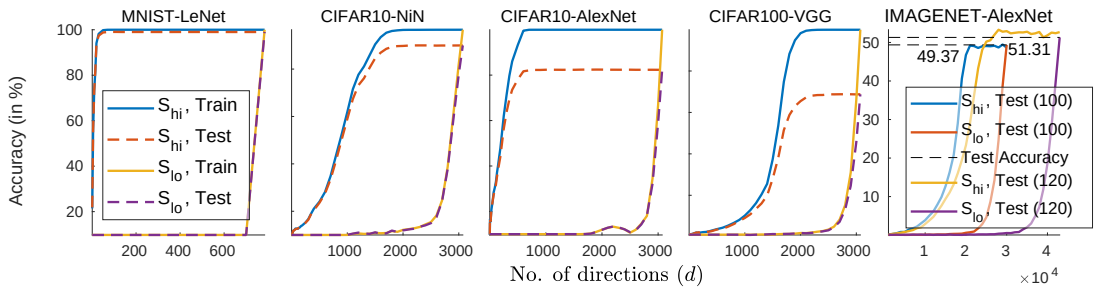


Figure 7.6: Classification accuracies on image sets projected onto subspaces of the spans of their corresponding DeepFool perturbations. For each net-dataset pair, DeepFool perturbations are computed over the image set and assembled into a matrix that is decomposed into its SVD. The singular vectors are ordered as per their singular values: S_{hi} represents the high-to-low ordering, S_{lo} the low-to-high, and d the number of vectors retained. Compare this figure to Figure 7.5 (while noticing how d now counts the total number of directions). For the ImageNet experiments, owing to memory constraints, the SVD is performed on downsampled DeepFools of size $100 \times 100 \times 3$ and $120 \times 120 \times 3$, respectively. The resulting singular vectors span the entire effective classification space of correspondingly downsampled images. This is evinced by the fact that the classification accuracy of images projected onto the singular vectors' subspace saturates to the same performance as that yielded when the net is tested directly on the downsampled images.

the direct relationship between the fraction of adversarial norm in given subspaces and the corresponding usefulness of those subspaces for classification. The inclusion of the saliency images of Simonyan et al. (2013) alongside the attack methods makes explicit the fact that adversaries are themselves an exposure of the net's notion of saliency.

By now, two results hint at a simpler and more direct way of identifying bases of classification/adversarial directions. First, a close inspection of the class-score curves sampled and displayed in Figure 7.1 reveals a direct connection between the curvature of a direction near the origin and its derivative magnitude over a fairly large interval around it. Second, this observation is made more clear in Figure 7.5 where it can be seen that the directions obtained by boundary curvature analysis in Alg. 1 correspond to the directions exploited by various *first*-order methods. Thus, we hypothesise that to identify such a basis, one need actually only perform SVD on a matrix of stacked class-score gradients*. Here, we implement this using a collection of DeepFool perturbations to provide the required gradient information, and repeat the analysis of Sec. 7.4.2, using singular values to order the vectors. The results, in Figure 7.6, neatly replicate the previously seen

* In fact, this analysis is begun in Moosavi-Dezfooli* et al. (2017), but only the singular *values* are examined.

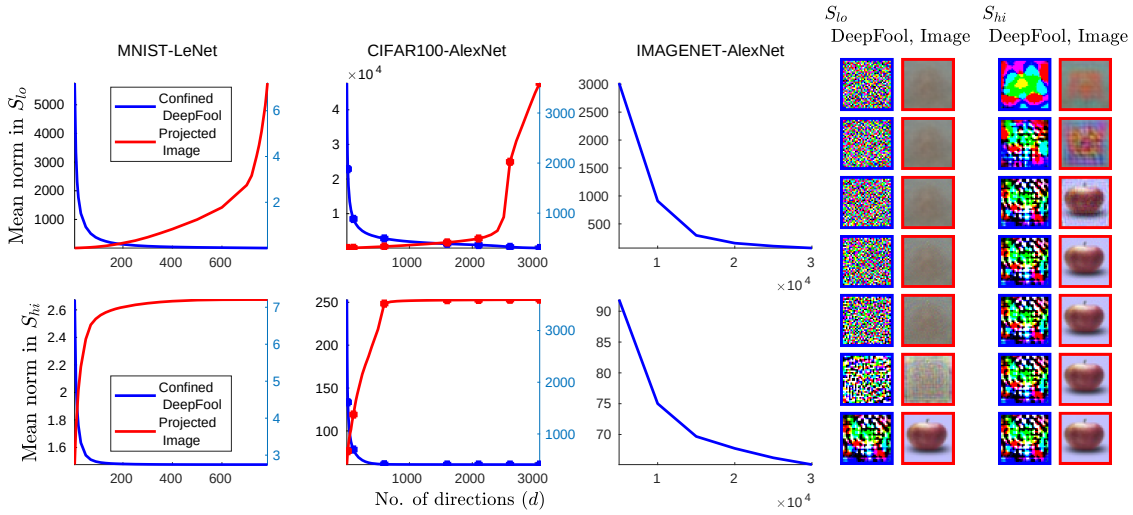


Figure 7.7: Blue curves depict the mean ℓ_2-norms of "confined DeepFool" perturbations: those that are calculated under strict confinement to the respective subspaces of Figure 7.6, also detailed in Sec. 7.4.3. Note the differences in scale of the y-axes of the different plots. For MNIST and CIFAR, we also plot (in red) the mean norms of the projections of the input images onto those subspaces: observe the inverse relationship between the two curves. The columns on the right visualise, from top to bottom, sample images at the indicated points on the curves in the CIFAR100-AlexNet plots, from left to right: blue-bordered images represent confined DeepFool perturbations (rescaled for display), with their red-bordered counterparts displaying the projection of the corresponding sample CIFAR image onto the same subspace. Observe that when the human-recognisable object appearance is captured in any given subspace, the corresponding DeepFool perturbation becomes maximally effective (*i.e.* small-norm). Likewise, when the projected image is not readily recognisable to a human, the DeepFool perturbation is large. The feature space *per se* does not account for adversariality: the issue is in the net's response to the features.

classification accuracy trends for high-to-low and low-to-high curvature traversal of image-space directions. Henceforth, we use these directions directly, simplifying analysis and allowing us to analyse ImageNet networks. The results on ImageNet dataset are shown for the first time in Figure 7.6. The network is use is AlexNet (Krizhevsky et al., 2012b).

While Figure 7.5 displays the magnitudes of components of pre-computed adversarial perturbations in different subspaces, we also design a variation on the analysis to illustrate how effective an efficient attack method (DeepFool) is when *confined* to the respective subspaces. This is implemented by simply projecting the gradient vectors used in solving DeepFool's linearised problem onto each subspace before otherwise solving the problem as usual. The results, displayed in Figure 7.7, thus represent DeepFool's "earnest" attempts to attack the network as efficiently as possible *within* each given subspace. It is evident that the attack *must* exploit genuine classification directions

in order to achieve low norm.

d_{low}	$E_n\{\ell_2\ norm(\mathbf{i}_n)\}$	$E_n\{\ell_2\ norm(\boldsymbol{\delta}\mathbf{i}_n)\}$	Accuracy (%)	Fooling rate (%)					
				$f=1$	$f=2$	$f=3$	$f=4$	$f=5$	$f=10$
227	26798.72	63.96	57.75	100.00	100.00	100.00	100.00	100.00	100.00
200	26515.20	53.19	55.80	32.75	77.25	88.95	92.20	94.35	97.65
150	26327.03	46.86	53.50	35.55	58.35	77.90	85.95	89.25	95.65
120	26159.98	41.92	51.75	36.15	49.80	66.20	76.90	82.95	92.90
100	26008.02	37.98	48.10	41.65	49.25	59.95	68.05	74.80	88.30

Table 7.1: The images $\mathbf{i}_n$ used to train AlexNet operate at the scale of $d_{orig} = 227$ (pixels on a side). In the pre-processing step, these images are downsized to d_{low}, before being upsampled back to the original scale. The reconstructed DeepFool perturbations $\boldsymbol{\delta}\mathbf{i}_n$ lose some of their effectiveness, as seen in the fooling-rate column for $f = 1$. When the effect of downsampling is countered by increasing the value of the ℓ_2-norms of these perturbations (using higher values of f), their efficacy is steadily restored. Note that the mean norms of images and perturbations are estimated in the upscaled space, as are the classification accuracies. The accuracy values for $d_{low} = \{100, 120\}$ should be compared to those at convergence in Figure 7.6. Any difference in the performance scores is strictly due to the random selection of the subset of 2000 test images used for evaluation.

7.4.4 On Image Compression and Robustness to Adversarial Attack

The above observations have made it clear that the most effective directions of adversarial attack are also the directions that contribute the most to the CNNs' classification performance. Hence, any attempt to mitigate adversarial vulnerability by discarding these directions, either by compressing the input data (Maharaj, 2015; Das et al., 2017; Xie et al., 2017) or by suppressing specific components of image representations at intermediate network layers (Gao et al., 2017a), must effect a loss in the classification accuracy. Further, our framework anticipates the fact that the nets must remain just as vulnerable to attack along the remaining directions that continue to determine classification decisions, given that the corresponding class-score functions, which possess the properties discussed earlier, remain unchanged. We use image downsampling as an example data compression technique to illustrate this effect on ImageNet.

We proceed by inserting a pre-processing unit between the CNN and its input at test time. This unit downsamples the input image $\mathbf{i}_n$ to a lower size d_{low} before upsampling it back to the original input size d_{orig}. The resizing (by bicubic interpolation) serves to reduce the effective dimensionality of the input data. For a randomly selected set of 2000 ImageNet (Russakovsky* et al., 2015) test images, we observe the change in classification

accuracy over different values of d_{low}, shown in column 4 of Table 7.1. The fooling rates* for the downsampled versions of these natural images' adversarial counterparts, produced by applying DeepFool to the *original* network (without the resampling unit), follow in column 5 of the table. At first glance, it appears that the downsampling-based pre-processing unit has afforded an increase in the network robustness at a moderate cost in accuracy. Results pertaining to this tradeoff have been widely reported (Maharaj, 2015; Das et al., 2017; Gao et al., 2017a). Here, we take the analysis a step further.

To start, we note the fact that the methodology just described represents a transfer attack from the original net to the net as modified by the inclusion of the resampling unit. As DeepFool perturbations $\boldsymbol{\delta}\mathbf{i}_n$ are not designed to transfer in this manner, we first augment them by simply increasing their ℓ_2-norm by a scalar factor f. We adjust f from unity up to a point at which the mean DeepFool perturbation norm is still a couple of orders of magnitude smaller than the mean image norm, such that the perturbations remain largely imperceptible. The corresponding fooling rates grow steadily with respect to f, as is observable in Table 7.1. Hence, although the original full-resolution perturbations may be suboptimal attacks on the resampling variants of the network (as some components are effectively lost to projection onto the compressed space), sufficient rescaling restores their effectiveness. On the other hand, the modified net continues to be equally vulnerable along the remaining effective classification directions, and can easily be attacked *directly*. To go about this, we simply take the SVD of the stack of downsampled DeepFool perturbations, for d_{low} values of 100 and 120 (owing to memory constraints). The resulting singular vectors span the entire space of classification/adversarial directions of the corresponding resampling network, as can be seen from the accuracy values in the rightmost subplot of Figure 7.6. More crucially, lower-norm DeepFools can be obtained by restricting the attack's iterative linear optimisation procedure to the space spanned by these compressed perturbations, exactly as described in Sec. 7.4.3 and displayed in Figure 7.7. This subspace-confined optimisation is analogous to designing a white-box DeepFool attack for the new network architecture inclusive of the resampling unit, instead of the original network, and is as effective as before. Note that this observation

* Measured as a percentage of samples from the dataset that undergo a change in their predicted label.

is consistent with the results reported in Xie et al. (2017), where the strength of the examined gradient-based attack methods increases progressively as the targeted model better approximates the defending model.

7.5 Discussion

First, regarding the main result of this work, we would like to point out that the discovery of the universal adversarial perturbations of Moosavi-Dezfooli* et al. (2017) strongly hinted at this outcome in advance. Those attacks are "universal" in precisely the sense that certain fixed directions perturb different images across class boundaries irrespective of the diversity in their individual appearances, *i.e.*, irrespective of the input components in all directions orthogonal to the perturbation. In fact, it is precisely *because* the functions are as separable as they are in this sense that the method used is able to identify them as well as it does. Note also that our results explain the "dominant label" phenomenon of Moosavi-Dezfooli* et al. (2017), noted therein as curious but otherwise left unaddressed, in which a given universal perturbation overwhelmingly moves examples into a small number of target classes regardless as to original class identities. This is a manifestation of the targeting of particular classes by particular directions, a phenomenon so strong that it manifests in Moosavi-Dezfooli* et al. (2017) *despite* the fact that their algorithm never explicitly optimises for this property.

In the context of discussing the adversarial vulnerability of CNNs, we would advise caution in using terms 'overfitting' and 'generalisation', particularly if done speculatively. Inspection of Figures 7.5 and 7.6 will convince the reader that the directions of vulnerability produce fits that generalise very *well* to unseen data generated by the same distribution, observable as the near identity between training and test accuracy curves over the directions of highest curvature (or derivative) magnitude. This does *not* correspond to the classical notion of overfitting in machine learning. In fact, overfitting, observable as the divergence between the train and test error curves, happens over less salient (and thus, less vulnerable) directions. This can again be confirmed by inspection. The fact that the nets extract characterisations of their targets that do not correspond to that of humans is a separate issue.

Finally, the approximate symmetry of the feature response functions may appear counter-intuitive: why should the additive inverse of a given perturbation pattern produce such a similar result to the original pattern? We suggest that at least part of the explanation may rest in a fact long known in the hand-engineering of features for computer vision: natural image descriptors must typically neglect sign, because contrast inversion is a fact of the world. For instance, consider how a black bird and a white bird, which share the label *bird*, differ from one another when both are set against the background of a blue sky. One can revisit, for instance, Dalal & Triggs (2005) for a reminder. Empirically, this nonlinearity appears to be particularly important. Note that CNNs trained on MNIST are exceptional in that they may not adhere to this, as MNIST is unnatural in its fixing of contrast sign: *e.g.* the net need never learn that a black vertical stroke against a white background also represents the digit '1'.

7.6 Conclusion

In this work, we expose a collection of directions along which the nets' class-score functions exhibit a strikingly similar behaviour across sample images. These functions are nonlinear, but are *de facto* of a relatively constrained form: axis-symmetric* and typically monotonic over large ranges. We illustrate a close relationship between these directions and class identity: many such directions effectively encode the extent to which the nets believe that a particular target class is or is not present. Thus, as it stands, the predictive power and adversarial vulnerability of studied nets are intertwined owing to the fact that they base their classification decisions on rather simplistic responses to components of the input images in specific directions, *irrespective* of whether the source of those components is natural or adversarial. Clearly, any gain in robustness obtained by suppressing the nets' responses to these components must come at the cost of a corresponding loss of accuracy. We demonstrate this experimentally. We also note that these robustness gains may be lower than they appear, as the networks actually remain vulnerable to a properly designed attack along the remaining directions they continue to use. To conclude, we believe that for any scheme to be truly effective against the

*Though not necessarily so for MNIST, because of its constraints, as discussed in the previous section.

problem of adversarial vulnerability, it must lead to a fundamentally more insightful (and likely complicated) use of features than presently occurs. Until then, those features will continue to be the nets' own worst adversaries.

7.A Appendix

7.A.1 Illustration of Fundamental Geometric Argument

The details are depicted and discussed in Figure 7.8.

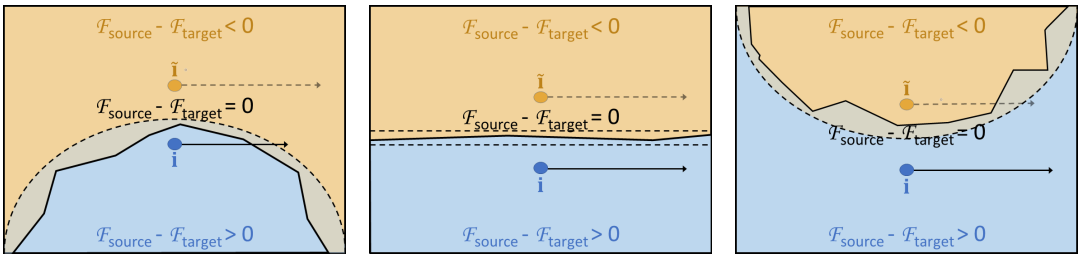


Figure 7.8: The left image illustrates the geometric argument of Moosavi-Dezfooli* et al. (2018) connecting positive curvature to universal perturbations: from the perspective of a source image $\mathbf{i}$, if the actual class boundary (solid black) can be locally bounded on the outside by an arc (dashed black) of fixed positive curvature along a particular direction, then a perturbation along that direction large enough to cross the bounding arc *must* carry $\mathbf{i}$ across the actual boundary, changing its class. If such a direction is "shared" over many samples, it will accordingly represent a universal perturbation over that set. The right image illustrates that this effect is not particular to positive curvatures: in the negative-curvature case, the same holds true for an imagined point $\tilde{\mathbf{i}}$ on the other side of the boundary. The central image shows that a "flat" direction cannot have a material effect on membership of *either* class.

7.A.2 Details of Use of Differential Geometric Concepts

For a more involved treatment of the basic design of the decision boundary curvature analysis algorithm, consult the references Moosavi-Dezfooli* et al. (2018) and Fawzi* et al. (2017) from which it is derived. If still in need of further clarification of fundamental concepts of differential geometry, consult a book such as Carmo (1976). We now address some specifics not discussed elsewhere, including in our own main text.

For one, it may seem strange to speak of the "curvature" of a decision boundary of a ReLU network, as analytically, this is everywhere either zero or undefined. "Curvature" is here (and in the relevant citations) computed via finite-difference numerical methods, implicitly corresponding to a smooth approximation of the actual piecewise linear boundary.

The astute reader may likewise notice that the use of the terms 'principal direction' and 'principal curvature' here (as originally in Moosavi-Dezfooli* et al. (2018)) is somewhat relaxed. Let us clarify this point. Strictly speaking, the principal directions at a point on a manifold in an embedding space (such as an $(N-1)$-dimensional class decision boundary embedded in N-dimensional image space) are a local concept, forming an orthonormal basis spanning the space tangent to the manifold at that point. The principal curvature associated with each principal direction is the curvature, at the point of tangency, of the normal section of the manifold in the principal direction.

Generally, there is no reason to think that tangent spaces at different points on the boundary surface should coincide, and so *a priori*, it may not make any sense to speak of "principal directions" in the embedding space. However, Moosavi-Dezfooli* et al. (2018) are fully aware of this, and base their analysis on the hypothesis that there exist *image*-space directions which, when projected onto the respective tangent spaces of different points sampled from the boundary, correspond to similar curvature patterns. In other words, they assume that these directions are largely shared across sample images. A curvature can then be associated with each such direction by, for instance, taking the mean of the curvatures measured at those sample points. This is the relaxation of the terminology referred to above: the "principal directions" here represent a rotation of the canonical (N-dimensional) image-space axes, and the "principal curvature" associated with each direction represents the sample mean curvature measured along the tangent component of that vector at each sample point in a set.

In practice, the above can in fact be simplified, as is done explicitly in the version of the algorithm given in the main text as Alg. 1. Rather than managing the expense and complexity of the tangent-space projections, it suits our purposes here to work directly with the Hessian of the embedding function, effectively omitting the projection step.

Finally, we note that in contrast to the simplification given in the main text, the (very large) sample mean Hessians $\overline{\mathbf{H}}$ are never actually computed and stored explicitly. Instead, for each $\overline{\mathbf{H}}$, a function $f_{\overline{\mathbf{H}}}(\mathbf{v})$ is constructed that approximates $\overline{\mathbf{H}}\mathbf{v}$ via finite differences of backpropagated gradients. The MATLAB function *eigs* uses $f_{\overline{\mathbf{H}}}(\mathbf{v})$ to compute the eigendecomposition of the approximated $\overline{\mathbf{H}}$.

8 Conclusion

Contents

In this chapter, we begin by summarising the research work presented in the earlier chapters. We then discuss some ripple-effects of that work; the way in which the work has been interpreted, used and extended. Finally, we end with a discussion about 'where we are' and 'where we may be headed' based on our findings and observations.

8.1 Research Review

In this thesis, we proposed and tested a variety of successful extensions to conventional CNN models for an array of scene understanding tasks. These extensions were aimed at incorporating available domain knowledge about the tasks of interest into existing CNN pipelines. Traditional CNN models often rely on general constructs to solve problems. In our work, the inclusion of task-specific knowledge, in the form of network architecture redesign or loss function reformulation, provided a greater (implicit) supervision during network training and inference. This in turn served to improve the predictive performance of the networks while also contributing to their interpretability.

The later part of the thesis analysed the functioning of classification CNNs, first in a qualitative manner via the instantiation of trainable attention modules at different stages in the network pipeline, and then geometrically via the characterisation of network class decision boundaries in the input image space. The learned attention maps revealed that the network classification decision proceeded from a relatively limited region of the input image. This region often coincided with the semantic parts of the inlying objects. For this reason the binarised attention maps could serve as useful priors for foreground object segmentation. Moreover, allowing the network to implement this 'selective focus' yielded consistent gains in the output classification performance.

The next study conducted a principled analysis of the class decision functions implemented by CNNs. It uncovered a slightly disturbing trend. We noticed specific image space directions to be associated with fixed class identities. In other words, an increase in the magnitude of the input image component in a given direction could almost monotonically increase (or decrease) the output score for a predetermined class. This change in the output class score was almost independent of the input image component in the space of directions orthonormal to the one specific direction under study. Moreover, these image space directions did not exhibit any semantic link to the underlying classes. These and other results allowed us to note that the performance capabilities of classification CNNs currently in use are closely entwined with their adversarial vulnerability via the fairly simplistic use of specific image space directions (i.e. visual features) for image classification. In the light of this observation, we then discussed the limited viability of compression-based or subtractive methods for making networks robust to adversarial noise. Overall, we offered a new lens to scrutinise and explain the behaviour of CNNs, with far reaching implications for constructing models that are both robust and accurate.

8.2 Discussion of Impact

Our work on the **use of class prototypical templates** for object recognition prompted the use of a phoneme based embedding space for cross-language knowledge transfer (Arora et al., 2016). Phonemes (the standardised non-decomposable units of spoken language)

can be thought of as the linguistic twins of visual prototypes. Consequently, the use of a phoneme based disentangling space can allow acoustic models to generalise across different languages, as shown by Arora et al. (2016). Another recent work (Drumond et al., 2017) has elaborated on the fact that the use of a prototypical embedding space can help to understand the distribution of class samples in a CNN feature space, thereby making the process more open and interpretable.

The topic of our next research study was **human saliency estimation**. Saliency modelling is increasingly useful for guiding high-precision processing to relevant regions of the input stream in real-time systems such as for surveillance and navigation (Mehmood et al., 2015; Fu et al., 2018; Ip & Varshney, 2011; Chang et al., 2010). Thus, human saliency prediction has become a crucial first-step for a wide range of applications. We proposed a deep convolutional model for this task, treating the output saliency map as a 2D probability distribution, and submitted our top-performing model to the MIT Saliency Benchmark (Bylinskii et al., 2012). It still ranks amongst the top 15 of 86 models in 6 out of 8 metrics evaluated on the benchmark. Consequently, new methods for human saliency estimation are regularly compared against ours. Through our proposed approach, we also argued for a shift from the traditional classification based losses for saliency estimation to losses that are better adapted to the task. More recently, some new approaches have found resonance in this idea. These have in turn explored different network training schemes for the task of saliency estimation such as adversarial training (Pan et al., 2017), and training as a segmentation task (He & Pugeault, 2018). In the former, the saliency prediction function is learned via a duel between a generator and discriminator. The latter treats it as a segmentation task and thus learns to predict distinct salient regions corresponding to different levels of saliency score quantisation.

Further, the **instance segmentation** system proposed in this thesis formed the basis of an innovation pitch titled 'Audio-Guided Navigation for the Partially Sighted' (Oxford, 2017). As a team of 3, we presented a real world application of this system for assisting the partially sighted in navigating everyday outdoor environments. We proposed the use of the instance segmentation system to recognise and localise common nuisance objects on the streets such as trash bins, parked vehicles, and advertisement boards. Once this

information is obtained in real time, it can then be passed on to a 'sonification' unit which could communicate the relevant information to the user via sound signals. For developing ways of mapping objects into pleasing yet functional sounds, we also collaborated with an Art & Technology workshop - 'Reconfigured Vision Workshop' (Sagar, 2017) held in London in June 2017. This workshop was aimed at exploring new and creative ways of mapping a physical environment into a sonic landscape. We presented our work at this workshop and closely engaged with the audience to gather useful insights; many of the members of the audience suffered from partial sight. Ultimately, we would like our observations and deliberations to feed into developing better sound-guided navigation tools for assisting the partially sighted.

For our work on **attention in classifications CNNs**, it is exciting to see that the learned attention mechanism is fast being adopted for augmenting, interpreting and validating the performance of CNN models applied to medical image analysis (Kim et al., 2018; Schlemper et al., 2018; Oktay et al., 2018).

8.3 Future Work

Since the advent of deep learning and big data in computer vision, CNN based models have achieved remarkable success on a variety of tasks (Simonyan & Zisserman, 2015; Redmon & Farhadi, 2016; Zheng et al., 2015; He et al., 2017). However, through all this, a clear understanding of their functional properties was largely elusive. We hope that with our latest work (Jetley et al., 2018a), we have gone some distance in unravelling the magical performance properties of these nets. Classification CNNs rely on proxy visual features in a rather simplistic manner in order to classify input images. This approach is at the heart of their celebrated performance results, but also renders them prone to adversarial attack; features of small magnitude that are undetectable by humans can change the prediction of nets.

Adversarial vulnerability has also been noticed in systems for segmentation (Hendrik Metzen et al., 2017) and speech-to-text conversion (Carlini & Wagner, 2018; Cisse et al., 2017). We hypothesise that similar characteristics underlie the adversarial weakness of the above mentioned systems as those discussed for image classification networks. It

would be useful to confirm this and unify our understanding of the different instantiations of CNN models for different tasks and datasets. In addition, it is yet to be confirmed if the nature of class score functions as uncovered in our work is an outcome of the network architecture, the training algorithm (Bottou, 2010), loss function specification or some combination of those. It is also yet to be confirmed if the networks can learn more complex functions when provided with more data or if they are somehow limited in terms of the complexity of the functions they can implement. Thus, overall, my future aim would be to continue to probe neural network architectures from disparate directions of network compressibility, effects of loss function formulation and architectural or structural constraints on the learned function; use the resultant findings to motivate better design choices with the goal of building models that are both accurate and robust; and in the process, be able to say something about the underlying properties and mechanism of these models in the interest of interpretability and transparency.

Finally, while we have identified and unravelled an apparent flaw in learned CNN models, we would like to note that these observations do not take away from the predictive power of these nets. In fact, they complement the performance merits of CNNs with a greater understanding of the underlying process. This understanding is expected to feed into the development of newer and better computational models for computer vision and related fields. We must see this as an avenue, for as Einstein said - “Nearly every great advance in science arises from a crisis in the old theory, through an endeavour to find a way out of the difficulties created.”

Bibliography

Radhakrishna Achanta and Sabine Süsstrunk. Saliency detection for content-aware image resizing. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, 2009.

T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(12):2037–2041, Dec 2006.

Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. Label-embedding for attribute-based classification. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 819–826, 2013.

Pablo Andrés Arbeláez, Jordi Pont-Tuset, Jonathan T. Barron, Ferran Marqués, and Jitendra Malik. Multiscale combinatorial grouping. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 328–335, 2014.

Anurag Arnab and Philip H S Torr. Bottom-up instance segmentation using deep higher-order crfs. In *Proceedings of British Machine Vision Conference (BMVC)*. BMVA Press, 2016.

Anurag Arnab and Philip P.H. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, and Philip HS Torr. Higher order conditional random fields in deep neural networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 524–540. Springer, 2016.

Vipul Arora, Aditi Lahiri, and Henning Reet. Attribute based shared hidden layers for cross-language knowledge transfer. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pp. 617–623. IEEE, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Pierre Baldi and Yves Chauvin. Neural networks for fingerprint recognition. In *Neural Computation, 5(3)*, 1993.

Evgeniy Bart and Shimon Ullman. Cross-generalization: Learning novel classes from a single example by feature replacement. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pp. 672–679. IEEE, 2005.

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(4):509–522, April 2002. ISSN 0162-8828.

Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 850–865. Springer, 2016.

Gunilla Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34(3):344–371, 1986.

A Borji and L. Itti. State-of-the-art in visual attention modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.

Ali Borji and Jimmy Tanner. Reconciling saliency and object center-bias hypotheses in explaining free-viewing fixations. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 2015.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 737–744, 1994.

Rechele Brooks and Andrew N Meltzoff. The development of gaze following and its relation to language. *Developmental science*, 8(6):535–543, 2005.

N. Bruce and J. Tsotsos. Saliency based on information maximization. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2006.

Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark. http://saliency.mit.edu/, 2012.

Chunshui Cao, Xianming Liu, Yi Yang, Yinan Yu, Jiang Wang, Zilei Wang, Yongzhen Huang, Liang Wang, Chang Huang, Wei Xu, Deva Ramanan, and Thomas S. Huang. Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2015.

Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *Deep Learning and Security Workshop*, 2018.

M.P. Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976. ISBN 9780132125895. URL `https://books.google.co.uk/books?id=1v0YAQAAIAAJ`.

Moran Cerf, Jonathan Harel, Wolfgang Einhäuser, and Christof Koch. Predicting human gaze using low-level saliency combined with face detection. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2008.

Chin-Kai Chang, Christian Siagian, and Laurent Itti. Mobile robot vision navigation & localization using gist and saliency. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4147–4154. IEEE, 2010.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016.

Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Enriching visual knowledge bases via object discovery and segmentation. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2027–2034, 2014.

Yi-Ting Chen, Xiaokai Liu, and Ming-Hsuan Yang. Multi-instance object segmentation with occlusion handling. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3470–3478, 2015.

Dan Cireşan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32:333–338, 2012.

Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep big simple neural nets excel on handwritten digit recognition. *CoRR*, abs/1003.0358, 2010. URL `http://arxiv.org/abs/1003.0358`.

Moustapha M Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 6977–6987, 2017.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 215–223. PMLR, 2011.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016.

Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. *CoRR*, abs/1512.04412, 2015.

Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 534–549. Springer, 2016a.

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 379–387. Curran Associates Inc., 2016b.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pp. 886–893. IEEE, 2005.

Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E. Kounavis, and Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression. *CoRR*, abs/1705.02900, 2017.

Thalita Drumond, Thierry Viéville, and Frédéric Alexandre. Using prototypes to improve convolutional networks interpretability. In *International Conference on Neural Information Processing Systems: Transparent and interpretable machine learning in safety critical environments Workshop*, 2017.

William Du, Michael Fang, and Margaret Shen. Siamese convolutional neural networks for authorship verification. Technical report, Stanford University, `http://cs231n.stanford.edu/reports/2017/pdfs`, No. 801, 2017.

Suyog Dutt Jain and Kristen Grauman. Active image segmentation propagation. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2864–2873, 2016.

S. Eslami and Christopher Williams. A generative model for parts-based object segmentation. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 100–107. Curran Associates, Inc., 2012.

SM Ali Eslami, Nicolas Heess, Christopher KI Williams, and John Winn. The shape boltzmann machine: a strong model of object shape. *International Journal of Computer Vision (IJCV)*, 107 (2):155–176, 2014.

Mark Everingham, Luc Van Gool, Chris Williams, J Winn, and A Zisserman. Pascal visual object classes challenge results. *Available from `www.pascal-network.org`*, 2005.

Mark Everingham, S. M. Ali Eslami, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, 111(1):98–136, 2015.

A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1778–1785. IEEE, June 2009. ISBN 978-1-4244-3992-8.

A. Fawzi*, S. M. Moosavi-Dezfooli*, and P. Frossard. Robustness of classifiers: from adversarial to random noise. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 1632–1640, 2016.

A. Fawzi, S. M. Moosavi-Dezfooli, and P. Frossard. The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine*, 34(6):50–62, 2017.

A. Fawzi*, S. M. Moosavi-Dezfooli*, P. Frossard, and S. Soatto. Classification regions of deep neural networks. *CoRR*, abs/1705.09552, 2017.

Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(4):594–611, 2006.

Jie Feng. Salient object detection for searched web images via global saliency. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3194–3201. IEEE, 2012. ISBN 978-1-4673-1226-4.

Vittorio Ferrari, Frederic Jurie, and Cordelia Schmid. From images to shape models for object detection. *International Journal of Computer Vision (IJCV)*, 2009.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 2121–2129, 2013.

Kui Fu, Jia Li, Hongze Shen, and Yonghong Tian. How drones look: Crowdsourced knowledge transfer for aerial video saliency prediction. *CoRR*, abs/1811.05625, 2018.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr 1980.

J. Gao, B. Wang, Z. Lin, W. Xu, and Y. Qi. Deepcloak: Masking deep neural network models for robustness against adversarial samples. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2017a.

Ji Gao, Beilun Wang, and Yanjun Qi. Deepmask: Masking DNN models for robustness against adversarial samples. *CoRR*, abs/1702.06763, 2017b.

Syed Omer Gilani, Ramanathan Subramanian, Yan Yan, David Melcher, Nicu Sebe, and Stefan Winkler. PET: An eye-tracking dataset for animal-centric pascal object classes. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, 2015.

Ross Girshick. Fast R-CNN. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2015.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587, 2014.

I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.

Ian Goodfellow, Honglak Lee, Quoc V Le, Andrew Saxe, and Andrew Y Ng. Measuring invariances in deep networks. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 646–654, 2009.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

Ya'nan Guo, Chongfan Luo, and Yide Ma. Object detection system based on multimodel saliency maps. *Journal of Electronic Imaging*, 26(2):023022, 2017.

Saurabh Gupta, Ross B. Girshick, Pablo Andrés Arbeláez, and Jitendra Malik. Learning rich features from RGB-D images for object detection and segmentation. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2014.

Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2006.

Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Majik. Semantic contours from inverse detectors. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2011.

Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2014.

Zeeshan Hayder, Xuming He, and Mathieu Salzmann. Boundary-aware instance segmentation. *CoRR*, abs/1612.03129, 2016. URL `http://arxiv.org/abs/1612.03129`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. URL `http://arxiv.org/abs/1703.06870`.

Sen He and Nicolas Pugeault. Salient region segmentation. *CoRR*, abs/1803.05759, 2018.

Marko Heikkilä, Matti Pietikäinen, and Cordelia Schmid. Description of interest regions with local binary patterns. *Pattern Recognition*, 42(3):425–436, March 2009. ISSN 0031-3203.

Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. Universal adversarial perturbations against semantic image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2755–2764, 2017.

Stephen L. Hicks, Iain Wilson, Louwai Muhammed, John Worsfold, Susan M. Downes, and Christopher Kennard. A depth-based head-mounted visual display to aid navigation in partially sighted individuals. *PLOS ONE*, 8(7):1–8, 07 2013.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006.

Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 340–353, 2012.

Seunghoon Hong, Junhyuk Oh, Honglak Lee, and Bohyung Han. Learning transferrable knowledge for semantic segmentation with deep convolutional neural network. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3204–3212, 2016.

Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

Xiaodi Hou, Jonathan Harel, and Christof Koch. Image signature: Highlighting sparse salient regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012.

Xun Huang, Chengyao Shen, Xavier Boix, and Qi Zhao. Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 262–270, 2015.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL `http://arxiv.org/abs/1502.03167`.

Cheuk Yiu Ip and Amitabh Varshney. Saliency-assisted navigation of very large landscape images. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1737–1746, 2011.

L. Itti and P. F. Baldi. Bayesian surprise attracts human attention. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2006.

Laurent Itti and Christof Koch. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision research*, 40(10):1489–1506, 2000.

Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (11):1254–1259, 1998.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 2017–2025, 2015.

Anil K. Jain, Yu Zhong, and Sridhar Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 18(3):267–278, 1996.

Saumya Jetley, Bernardino Romera-Paredes, Sadeep Jayasumana, and Philip Torr. Prototypical priors: From improving classification to zero-shot learning. In *Proceedings of British Machine Vision Conference (BMVC)*, pp. 120.1–120.12. BMVA Press, September 2015.

Saumya Jetley, Naila Murray, and Eleonora Vig. End-to-end saliency mapping via probability distribution prediction. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016a.

Saumya Jetley, Naila Murray, and Eleonora Vig. End-to-end saliency mapping via probability distribution prediction. `https://patents.google.com/patent/US9830529B2/en`, 2016b.

Saumya Jetley, Michael Sapienza, Stuart Golodetz, and Philip H. S. Torr. Straight to shapes: Real-time detection of encoded shapes. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

Saumya Jetley, Nicholas Lord, and Philip Torr. With friends like these, who needs adversaries? In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2018a.

Saumya Jetley, Nicholas A. Lord, Namhoon Lee, and Philip Torr. Learn to pay attention. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2018b. URL `https://openreview.net/forum?id=HyzbhfWRW`.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of ACM International conference on Multimedia (ACM MM)*, 2014.

Bowen Jiang, Lihe Zhang, Huchuan Lu, Chuan Yang, and Ming-Hsuan Yang. Saliency detection via absorbing markov chain. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 1665–1672, 2013.

Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. SALICON: Saliency in Context. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Alexis Joly and Olivier Buisson. Logo retrieval with a contrario visual query expansion. In *Proceedings of ACM International conference on Multimedia (ACM MM)*, pp. 581–584. ACM, 2009.

Armand Joulin, Francis Bach, and Jean Ponce. Discriminative clustering for image co-segmentation. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1943–1950. IEEE, 2010.

Armand Joulin, Francis Bach, and Jean Ponce. Multi-class cosegmentation. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 542–549. IEEE, 2012.

Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Tilke Judd, Frédo Durand, and Antonio Torralba. A benchmark of computational models of saliency to predict human fixations. In *MIT Technical Report*, 2012.

Wolf Kienzle, Felix A Wichmann, Bernhard Schölkopf, and Matthias O Franz. A Nonparametric Approach to Bottom-Up Visual Saliency. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2007.

Seong Tae Kim, Jae-Hyeok Lee, Hakmin Lee, and Yong Man Ro. Visually interpretable deep network for diagnosis of breast masses on mammograms. *Physics in Medicine & Biology*, 63 (23):235025, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2014.

Christof Koch and Shimon Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In *Matters of intelligence*, pp. 115–141. 1987.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *Journal of ICML Deep Learning Workshop*, 2015.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 1097–1105. Curran Associates, Inc., 2012a.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 1097–1105. Curran Associates, Inc., 2012b.

Srinivas S. S. Kruthiventi, Kumar Ayush, and R. Venkatesh Babu. Deepfix: A fully convolutional neural network for predicting human eye fixations. *CoRR*, abs/1510.02927, 2015.

M. Kümmerer, L. Theis, and M. Bethge. Deep Gaze I: Boosting Saliency Prediction with Feature Maps Trained on ImageNet. In *ICLR Workshop*, 2015.

Brenden M Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua B Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of 33rd Annual conference of the Cognitive Science Society*, volume 172, 2011.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 951–958. IEEE, 2009.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(3):453–65, March 2014. ISSN 1939-3539.

Congyan Lang, TamV. Nguyen, Harish Katti, Karthik Yadati, Mohan Kankanhalli, and Shuicheng Yan. Depth matters: Influence of depth cues on visual saliency. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2012.

Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, R. E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a back-propagation network. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 396–404. Morgan-Kaufmann, 1990.

Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 609–616. ACM, 2009.

Li-Jia Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 1–8. IEEE, 2007.

Yangyan Li, Hao Su, Charles Ruizhongtai Qi, Noa Fish, Daniel Cohen-Or, and Leonidas J. Guibas. Joint embeddings of shapes and images via CNN image purification. In *SIGGRAPH Asia*, 2015.

Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. *CoRR*, abs/1611.07709, 2016. URL `http://arxiv.org/abs/1611.07709`.

Z. Li, E. Gavves, T. E. J. Mensink, and C. G. M. Snoek. Attributes make sense on segmented objects. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2014.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2013.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 740–755. Springer, 2014a.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2014b.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015. URL `http://arxiv.org/abs/1512.02325`.

Zhi Liu, Olivier Le Meur, Shuhua Luo, and Liquan Shen. Saliency detection using regional histograms. *Optics letters*, 38(5):700–702, 2013.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, 2015.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, Nov 2004.

J. Lu, T. Issaranon, and D. Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 446–454, 2017.

Yan Luo, Yongkang Wong, and Qi Zhao. Label consistent quadratic surrogate model for visual saliency prediction. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Z. Ma, L. Qing, J. Miao, and X. Chen. Advertisement evaluation using visual saliency based on foveated image. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, pp. 914–917, June 2009.

Laurens Van Der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research (JMLR)*, 9:2579–2605, 2008.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

Akash V Maharaj. Improving the adversarial robustness of convnets by reduction of input dimensionality. Technical report, Stanford, CA: Department of Physics, Stanford University, 2015.

S. Mallat. Wavelets for a vision. *Proceedings of the IEEE*, 84(4):604–614, April 1996.

L. Marchesotti, C. Cifarelli, and G. Csurka. A framework for visual saliency detection with applications to image thumbnailing. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 2232–2239, Sept. 2009.

David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc., 1982. ISBN 0716715678.

Marcin Marszałek and Cordelia Schmid. Accurate object localization with shape masks. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

Stefan Mathe and Cristian Sminchisescu. Action from still image dataset and inverse optimal control to learn task specific visual scanpaths. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2013.

Irfan Mehmood, Muhammad Sajjad, Waleed Ejaz, and Sung Wook Baik. Saliency-directed prioritization of visual data in wireless surveillance networks. *Information Fusion*, 24:16–30, 2015.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *CoRR*, abs/1702.04267, 2017.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

Laurynas Miksys, Saumya Jetley, Michael Sapienza, Stuart Golodetz, and Philip H. S. Torr. Straight to shapes++: Real-time instance segmentation made more accurate. *CoRR*, abs/1905.11358, 2019.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 2204–2212, 2014.

S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

S. M. Moosavi-Dezfooli*, A. Fawzi*, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 86–94. IEEE, 2017.

Seyed-Mohsen Moosavi-Dezfooli*, Alhussein Fawzi*, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Robustness of classifiers to universal perturbations: A geometric perspective. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

Jonghwan Mun, Minsu Cho, and Bohyung Han. Text-guided attention model for image captioning. *CoRR*, abs/1612.03557, 2016.

Naila Murray, Maria Vanrell, Xavier Otazu, and C Alejandro Parraga. Saliency estimation using a non-parametric low-level vision model. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

Naila Murray, Maria Vanrell, Xavier Otazu, and C Alejandro Parraga. Low-level spatiochromatic grouping for saliency estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

Alexander Neubeck and Luc J. Van Gool. Efficient non-maximum suppression. In *Proceedings of International Conference on Pattern Recognition (ICPR)*, pp. 850–855, 2006.

A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 427–436, 2015.

Tam V. Nguyen, Qi Zhao, and Shuicheng Yan. Attentive systems: A survey. *International Journal of Computer Vision (IJCV)*, 126(1):86–110, 2018.

NIPS. 2017 competition on adversarial attacks and defenses. `https://www.kaggle.com/nips-2017-adversarial-learning-competition`, 2017. accessed: 2018-03-12.

Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S Corrado, and Jeffrey Dean. Zero-shot learning by convex combination of semantic embeddings. *CoRR*, abs/1312.5650, 2013.

Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. *CoRR*, abs/1804.03999, 2018.

Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 685–694, 2015.

Oxford. Audio-guided navigation for the partially sighted. `https://www.pmb.ox.ac.uk/news/pembroke-team-win-tri-innovate-2017-innovative-audio-navigation-concept`, 2017. Accessed: 2018-11-21.

Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. Zero-shot learning with semantic output codes. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 1410–1418, 2009.

Junting Pan and Xavier Giró i Nieto. End-to-end convolutional network for saliency prediction. *CoRR*, abs/1507.01422, 2015.

Junting Pan, Cristian Canton Ferrer, Kevin McGuinness, Noel E O'Connor, Jordi Torres, Elisa Sayrol, and Xavier Giro-i Nieto. Salgan: Visual saliency prediction with generative adversarial networks. *CoRR*, abs/1701.01081, 2017.

Seymour Papert. The summer vision project. https://dspace.mit.edu/handle/1721.1/6125, 1966.

Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2751–2758. IEEE, 2012.

Pedro O. Pinheiro, Ronan Collobert, and Piotr Dollár. Learning to segment object candidates. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, 2015.

Pedro O. Pinheiro, Tsung-Yi Lin, Ronan Collobert, and Piotr Dollár. Learning to refine object segments. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 413–420. IEEE, 2009.

Deepak Rao, Quoc V Le, Thanathorn Phoka, Morgan Quigley, Attawith Sudsang, and Andrew Y Ng. Grasping Novel Objects with Depth Segmentation. In *Proceedings of IEEE/RSJ International conference on Intelligent Robots and Systems (IROS)*, pp. 2578–2585, 2010.

Joseph Redmon. Darknet: Open source neural networks in c. `http://pjreddie.com/darknet/`, 2013–2016.

Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016. URL `http://arxiv.org/abs/1612.08242`.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 91–99, 2015.

Bernardino Romera-Paredes and Philip H. S. Torr. Recurrent instance segmentation. *CoRR*, abs/1511.08250, 2015.

Bernardino Romera-Paredes, ENG OX, and Philip HS Torr. An embarrassingly simple approach to zero-shot learning. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 2152–2161, 2015.

Michael Rubinstein, Armand Joulin, Johannes Kopf, and Ce Liu. Unsupervised joint object discovery and segmentation in internet images. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1939–1946, 2013.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, pp. 696–699. MIT Press, 1988. ISBN 0-262-01097-6. URL `http://dl.acm.org/citation.cfm?id=65669.104451`.

Martin Rünz and Lourdes Agapito. MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects. *CoRR*, abs/1804.09194, 2018.

Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

S. Sabour*, Y. Cao*, F. Faghri, and D. J. Fleet. Adversarial manipulation of deep representations. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2016.

Ilona Sagar. Reconfigured vision workshop. http://www.spacestudios.org.uk/art-technology/reconfigured-vision-workshop/, 2017. Accessed: 2018-11-21.

Boris Schauerte and Rainer Stiefelhagen. Quaternion-based spectral saliency detection for eye fixation prediction. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 116–129, 2012.

Jo Schlemper, Ozan Oktay, Michiel Schaap, Mattias P. Heinrich, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention gated networks: Learning to leverage salient regions in medical images. *CoRR*, abs/1808.08114, 2018.

Max Schwarz, Anton Milan, Arul Selvam Periyasamy, and Sven Behnke. Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter. *The International Journal of Robotics Research*, 37(4-5):437–451, 2018.

Kavita Bala Sean Bell. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (SIGGRAPH 2015)*, 2015.

Paul Hongsuck Seo, Zhe Lin, Scott Cohen, Xiaohui Shen, and Bohyung Han. Hierarchical attention networks. *CoRR*, abs/1606.02393, 2016.

Gaurav Sharma, Frédéric Jurie, and Cordelia Schmid. Discriminative spatial saliency for image classification. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Chengyao Shen and Qi Zhao. Webpage saliency. In *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 33–46. Springer International Publishing, 2014.

Chengyao Shen, Mingli Song, and Qi Zhao. Learning high-level concepts by training a deep network on eye fixations. In *NIPS Deep Learning and Unsupervised Feature Learning Workshop*, 2012.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.

K. Simonyan, A. Vedald, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *Proceedings of International conference on Neural Information Processing Systems (NIPS)*, pp. 935–943, 2013.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32: 323–332, 2012.

K. O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162, 2007.

Fred Stentiford. Attention based auto image cropping. In *The 5th International conference on Computer Vision Systems, Bielefeld*, 2007.

Masahiro Suzuki, Haruhiko Sato, Satoshi Oyama, and Masahito Kurihara. Transfer learning based on the observation probability of each attribute. *2014 IEEE International conference on Systems, Man, and Cybernetics (SMC)*, pp. 3627–3631, October 2014.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. URL `http://arxiv.org/abs/1512.00567`.

Michela C. Tacca. Commonalities between perception and cognition. *Front Psychology*, 2011.

T. Tanay and L. Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *CoRR*, abs/1608.07690, 2016.

Bugra Tekin, Isinsu Katircioglu, Mathieu Salzmann, Vincent Lepetit, and Pascal Fua. Structured prediction of 3d human pose with deep neural networks. In *Proceedings of British Machine Vision Conference (BMVC)*. BMVA Press, 2016.

Anne M Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

R. Valenti, N. Sebe, and T. Gevers. Image saliency by isocentric curvedness and color. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2009.

Eleonora Vig, Michael Dorr, and David Cox. Large-scale optimization of hierarchical features for saliency prediction in natural images. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of International Conference on Machine learning (ICML)*, pp. 1096–1103. ACM, 2008.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014. URL `http://arxiv.org/abs/1411.4555`.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011.

B. Wang, J. Gao, and Y. Qi. A theoretical framework for robustness of (deep) classifiers under adversarial noise. *CoRR*, abs/1612.00334, 2016.

Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, volume 11, pp. 2764–2770, 2011.

Sebastien C. Wong, Adam Gatt, Victor Stamatescu, and Mark D. McDonnell. Understanding data augmentation for classification: when to warp? *CoRR*, abs/1609.08764, 2016. URL `http://arxiv.org/abs/1609.08764`.

Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond PASCAL: A benchmark for 3D object detection in the wild. In *IEEE Winter conference on Applications of Computer Vision (WACV)*, 2014.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan L. Yuille. Mitigating adversarial effects through randomization. *CoRR*, abs/1711.01991, 2017.

Huijuan Xu and Kate Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *Proceedings of European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of International Conference on Machine Learning (ICML)*, 2015a.

Pingmei Xu, Krista A. Ehinger, Yinda Zhang, Adam Finkelstein, Sanjeev R. Kulkarni, and Jianxiong Xiao. Turkergaze: Crowdsourcing saliency with webcam based eye tracking. *CoRR*, abs/1504.06755v1, 2015b.

Zichao Yang, Xiaodong He, Jianfeng Gao, li Deng, and Alex Smola. Stacked attention networks for image question answering, 06 2016.

Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 1331–1338. IEEE, 2011.

L. Ye, Z. Liu, L. Li, L. Shen, C. Bai, and Y. Wang. Salient object segmentation via effective integration of saliency and objectness. *IEEE Transactions on Multimedia*, 19(8):1742–1756, Aug 2017.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4651–4659, 2016.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *CoRR*, abs/1612.03928, 2016.

Wolfgang H Zangemeister, HS Stiehl, and C Freksa. *Visual attention and cognition*. Elsevier, 1996.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2017.

Dengsheng Zhang and Guojun Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37, 2004.

H. Zhang, M. Cisse, Y. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2018. URL `https://openreview.net/forum?id=r1Ddp1-Rb`.

Jianming Zhang and Stan Sclaroff. Saliency detection: A boolean map approach. In *Proceedings of IEEE international Conference on computer vision (ICCV)*, pp. 153–160, 2013.

Lingyun Zhang, Matthew H. Tong, Tim K. Marks, Honghao Shan, and Garrison W. Cottrell. SUN: A Bayesian framework for saliency using natural statistics. *Journal of Vision*, 8(7):1–20, 12 2008. ISSN 1534-7362.

Yinda Zhang, Fisher Yu, Shuran Song, Pingmei Xu, Ari Seff, and Jianxiong Xiao. Large-scale scene understanding challenge. `http://lsun.cs.princeton.edu/leaderboard/#saliencysalicon`, 2015.

Ziming Zhang and Venkatesh Saligrama. Zero-shot learning via joint latent similarity embedding. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6034–6042, 2016.

Qi Zhao and Christof Koch. Learning a saliency map using fixated locations in natural scenes. *Journal of Vision*, 11(3), 2011.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 1529–1537, 2015.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of IEEE International conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2921–2929, 2016.

Pengfei Zhu, Longyin Wen, Xiao Bian, Ling Haibin, and Qinghua Hu. Vision meets drones: A challenge. *arXiv preprint arXiv:1804.07437*, 2018.

Prof. Song-Chun Zhu. Research: Are we on the right way? `http://www.stat.ucla.edu/~sczhu/research_blog.html`, 2017. Accessed: 2019-08-13.