

Cross Pixel Optical-Flow Similarity for Self-Supervised Learning

Aravindh Mahendran^[0000–0002–2650–9871], James Thewlis^[0000–0001–8410–2570], and
Andrea Vedaldi

Visual Geometry Group, University of Oxford
{aravindh, jdt, vedaldi}@robots.ox.ac.uk

Abstract. We propose a novel method for learning convolutional neural image representations without manual supervision. We use motion cues in the form of optical-flow, to supervise representations of static images. The obvious approach of training a network to predict flow from a single image can be needlessly difficult due to intrinsic ambiguities in this prediction task. We instead propose a much simpler learning goal: embed pixels such that the similarity between their embeddings matches that between their optical-flow vectors. At test time, the learned deep network can be used without access to video or flow information and transferred to tasks such as image classification, detection, and segmentation. Our method, which significantly simplifies previous attempts at using motion for self-supervision, achieves state-of-the-art results in self-supervision using motion cues, and is overall state of the art in self-supervised pre-training for semantic image segmentation, as demonstrated on standard benchmarks.

Keywords: Self-Supervised Learning · Motion · Convolutional Neural Network

1 Introduction

Self-supervised learning has emerged as a promising approach to address one of the major shortcomings of deep learning, namely the need for large supervised training datasets. All self-supervised learning methods are based on the same basic premise, which is to identify problems that can be used to train deep networks without the expense of collecting data annotations. In this spirit, an amazing diversity of supervisory signals have been proposed, from image generation to colorization, in-painting, jigsaw puzzle solving, orientation estimation, counting, artifact spotting, and many more (see section 2). Furthermore, the recent work of [9] shows that combining several such cues further helps performance.

In this paper, we consider the case of *self-supervision using motion cues* to learn a convolutional neural network (CNN) for static images. Here, a deep network is trained to predict, from a single video frame, how the image *could change* over time. Since predicted changes can be verified automatically by looking at the actual video stream, this approach can be used for self-supervision. Furthermore, predicting motion may induce a deep network to learn about objects in images. The reason is that objects are a major cause of motion regularity and hence predictability: pixels that belong to the same object are much more likely to “move together” than pixels that do not.

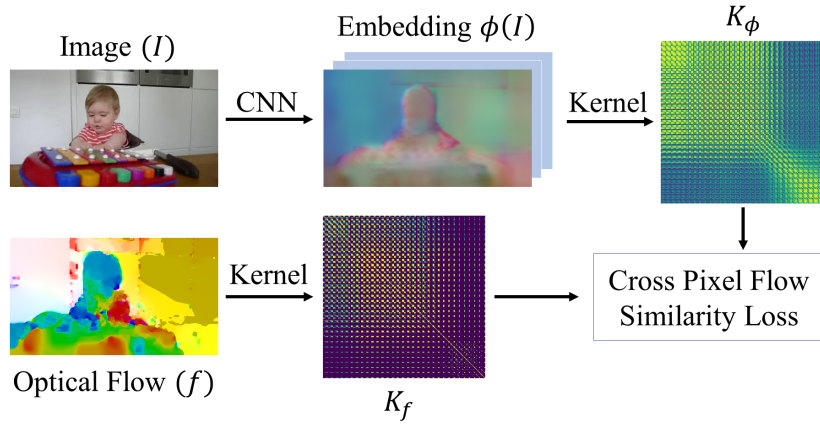


Fig. 1: We propose a novel method to exploit motion information represented as optical-flow, to supervise the learning of deep CNNs. We learn a network that predicts per-pixel embeddings $\phi(I)$ such that the kernel computed over these embeddings (K_ϕ) is similar to that over corresponding optical-flow vectors (K_f). This allows the network to learn from motion cues while avoiding the inherent ambiguity of motion prediction from a single frame.

Besides giving cues about objects, motion has another appealing characteristic compared to other signals for self-supervision. Many other methods are, in fact, based on destroying information in images (e.g. by removing color, scrambling parts) and then tasking a network with undoing such changes. This has the disadvantage of learning the representation on distorted images (e.g. gray scale). On the other hand, extracting a single frame from a video can be thought of as removing information only along the temporal dimension and allows one to learn the network on undistorted images.

There is however a key challenge in using motion for self-supervision: ambiguity. Even if the deep network can correctly identify all objects in an image, this is still not enough to predict the specific direction and intensity of the objects’ motion in the video, given just a single frame. This ambiguity makes the direct prediction of the appearance of future frames particularly challenging [54, 59], and overall an overkill if the goal is to learn a general-purpose image representation for image analysis. Instead, the previous most effective method for self-supervision using motion cues [42] is based on first extracting motion tubes from videos (using off-the-shelf optical-flow and motion tube segmentation algorithms) and then training the deep network to predict the resulting per-frame segments rather than motion directly. Thus they map a complex self-supervision task into one of classic foreground-background segmentation.

While the approach of [42] sidesteps the difficult problem of motion prediction ambiguity, it comes at the cost of pre-processing videos using a complex handcrafted motion segmentation pipeline, which includes many heuristics and tunable parameters. In this paper, we instead propose a new method that can ingest cues from optical-flow *directly*, without the need for any complex data pre-processing.

Our method, presented in section 3 and illustrated in fig. 1, is based on a new cross pixel flow similarity loss layer. As noted above, the key challenge is that specific details about the motion, such as its direction and intensity, are usually difficult if not impossible to predict from a single frame. We address this difficulty in two ways. First, we learn to embed pixels into vectors that cluster together when the model believes that the corresponding pixels *are likely to move together*. This is obtained by encouraging the inner product of the learned pixel embeddings to correlate with the similarity between their corresponding optical-flow vectors. This does not require the model to explicitly estimate specific motion directions or velocities. However, this is still not sufficient to address the ambiguity completely; in fact, while different objects may be *able* to move independently, they *may not do so* all the time. For example, often objects stand still, so their velocities are all zero grouping them together in optical-flow. We attempt to address this second challenge by using a contrastive loss.

In section 4 we extensively validate our model against other self-supervised learning approaches. First, we show that our approach works as well or better than [42], establishing a new state-of-the-art method for self-supervision using motion cues. Second, to put this into context, we also compare the results to all recent approaches for self-supervision that use cues other than motion. In this case, we show that our approach has state-of-the-art performance for semantic image segmentation, at the time of submission.

The overall conclusion (section 5) is that our method significantly simplifies leveraging motion cues for self-supervision and does better than existing alternatives for this modality; it is also competitive with self-supervision methods that use other cues, making motion a sensible choice for self-supervision by itself or in combination with other cues [9].

2 Related Work

Self-supervised learning, of which our method is an instance, has become very popular in the community. We discuss here the methods for training generic features for image understanding as opposed to methods with specific goals such as learning object keypoints. We group them according to the supervision cues they use.

Video/Motion Based: LSTM RNNs can be trained to predict future frames in a video [51]. This requires the network to understand image dynamics and extrapolate it into the future. However, since several frames are observed simultaneously, these methods may learn something akin to a tracker, with limited abstraction. On the other hand, we learn to predict properties of optical-flow from a **single input image**, thus learning a static image representation rather than a dynamic one. Closely related to our work is the use of *video segmentation* by [42]. They use an off-the-shelf video segmentation method [15] to construct a foreground-background segmentation dataset in an unsupervised manner. A CNN trained on this proxy task transfers well when fine-tuned for object recognition and detection. We differ from them in that we do not require a sophisticated pre-existing pipeline to extract video segments, but use optical-flow directly. Also closely related to us is the work of [2]. They train a Siamese style convolutional neural network to predict the transformation between two images. The individual base

networks in their Siamese architecture share weights and can be used as feature extractors for single images at test time. This late fusion strategy forces the learning of abstractions, but our **no-fusion approach** pushes the model even further to learn better features. The polar opposite of these is to do early fusion by concatenating two frames as in FlowNet [11]. This was used as a pretraining strategy by [16] to learn representations for **pairs of frames**. This is different from our objective as we aim to learn a **static image representation**. This difference becomes clearer when looking at the evaluation. While we evaluate on image classification, detection, and segmentation; [16] evaluate on dynamic scene and action recognition.

Temporal context is a powerful signal. [35, 57, 32] learn to predict the correct ordering of frames. [24] exploit both temporal and spatial co-occurrence statistics to learn visual groups. [25] extend slow feature analysis using higher order temporal coherence. [55] track patches in a video to supervise their embedding via a triplet loss while [17] do the same but for spatio-temporally matched region proposals. Temporal context is applied in the imitation learning setting by Time Contrastive Networks [49].

Videos contain more than just temporal information. Some methods exploit audio channels by predicting audio from video frames [48, 40]. [3] train a two stream architecture to classify whether an image and sound clip go together or not. Temporal information is coupled with ego-motion in [26]. [56] use videos along with spatial context pretraining [8] to construct an image graph. Transitivity in the graph is exploited to learn representations with suitable invariances.

Colorization: [31, 60] predict colour information given greyscale input and show competitive pre-training performance. [61] generalize to arbitrary pairs of modalities.

Spatial Context: [41] solve the in-painting problem, where a network is tasked with filling-in partially deleted parts of an image. [8] predict the relative position of two patches extracted from an image. In a similar spirit, [37, 38] solve a jigsaw puzzle problem. [38] also cluster features from a pre-trained network to generate pseudo labels, which allows for knowledge distillation from larger networks into smaller ones. The latest iteration on context prediction [36] obtains state-of-the-art on some benchmarks.

Adversarial/Generative: BiGAN based pretrained models [10] show competitive performance on various recognition benchmarks. [27] adversarially learn to generate and spot defects. [45] obtain self-supervision from synthetic data and adapt their model to the domain of real images by training against a discriminator. [5] predict noise-as-targets via an assignment matrix which is optimized on-line. Their approach is domain agnostic. More in general, generative unsupervised layer-wise pretraining was extensively used in deep learning before AlexNet [30]. An extensive review of these and more recent unsupervised generative models is beyond the scope of this paper.

Transformations: [12] create surrogate classes by applying a set of transformations to each image and learn to become invariant to them. [19] do the opposite and try to estimate the transformation (just one of four rotations in their case) given the transformed image. The crop-concatenate transformation is implicit in the learning by counting method of [39].

Others: A combination of self-supervision approaches was explored by [9]. They report results only with ResNet models making it hard to compare with concurrent work, but closely matching ImageNet-pretrained networks in performance on the PASCAL VOC detection task. A widely-applicable trick that helps in transfer learning is the re-balancing method of [29]. Lastly, our optical-flow classification baseline is based on the work of [4]. They learn a sparse hypercolumn model to predict surface normals from a single image and use this as a pretraining strategy. Our baseline flow classification model is the same but with AlexNet for discretized optical-flow.

3 Method

In this section, we describe our novel method, illustrated in fig. 1, for self-training deep neural networks via direct ingestion of optical-flow. Once learned, the resulting image representation could be used for classification, detection, segmentation and other tasks with minimal supervision.

Our goal is to learn the parameters of a neural network that maps a single image or frame to a field of pixel embeddings, one for each pixel. Notation - Let $\Omega \subset \mathbb{R}^2$ be the set of pixels; $I : \Omega \rightarrow \mathbb{R}^3$ is an image; Our CNN is the per-pixel mapping $\phi(I, p|\Theta) \in \mathbb{R}^D$ producing D dimensional embeddings. In order to learn this CNN, we require the **similarity** between **pairs** of embedding vectors to align with the similarity between the corresponding flow vectors. This is sufficient to capture the idea that things that move together should be grouped together, popularly known as the **Gestalt principle of common fate** [53].

Formally, given D -dimensional CNN embedding vectors $\phi(I, p|\Theta), \phi(I, q|\Theta) \in \mathbb{R}^D$ for pixels $p, q \in \Omega$ and their corresponding flow vectors $f_p, f_q \in \mathbb{R}^2$, we match the kernel matrices

$$\forall p, q \in \Omega : \quad K_\phi(\phi(I, p|\Theta), \phi(I, q|\Theta)) \cong K_f(f_p, f_q) \quad (1)$$

where $K_\phi : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$, $K_f : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$ are kernels that measure the similarity of the CNN embeddings and flow vectors, respectively.

In this formulation, in addition to the choice of CNN architecture ϕ , the key design decisions are the choice of kernels K_ϕ, K_f and how to translate constraint eq. (1) into a loss function. The rest of the section discusses these choices.

3.1 Kernels

In order to compare CNN embedding vectors and flow vectors, we choose the (scaled) cosine similarity kernel and the Gaussian/RBF kernel respectively. Using the shorthand notation $\phi_p = \phi(I, p|\Theta)$ for readability, these are:

$$K_\phi(\phi_p, \phi_q) := \frac{1}{4} \frac{\phi_p^T \phi_q}{\|\phi_p\|_2 \|\phi_q\|_2}, \quad K_f(f_p, f_q) := \exp\left(-\frac{\|f_p - f_q\|_2^2}{2\sigma^2}\right). \quad (2)$$

Note that these kernels, when restricted to the set of pixels Ω , are matrices of size $|\Omega| \times |\Omega|$. Each row or column of this matrix can be thought of as a heatmap capturing

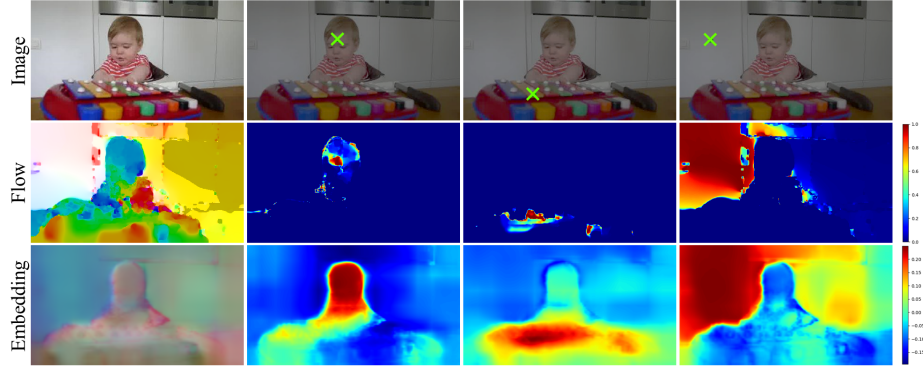


Fig. 2: Visualization of flow and embedding kernels, in the second and third rows respectively. For three pixels p , we plot the row $K_*(p, \cdot)$ reshaped into an image, showing which pixels go together from the kernel’s perspective. Note the localized nature of the flow kernel which is obtain by setting a low bandwidth for the RBF kernel. $\sigma^2 = 0.0036$ in this example. In the first column, optical-flow and embeddings (after a random $16D \rightarrow 3D$ projection) are visualized as color images.

the similarity of a given pixel with respect to all other pixels and thus can be visualized as an image. We present such visualizations for both of our kernels in fig. 2.

We use the Gaussian kernel for the flow vectors as this is consistent with the Euclidean interpretation of optical-flow as a displacement. Reducing kernel bandwidth (σ) would result in a localized kernel that pushes our embeddings to distinguish between different movable objects. In some experiments, the value of σ is learned along with the weights of the CNN in the optimization. This localized kernel, with learned $\sigma^2 = 0.0036$, is shown in the second row of fig. 2.

We use the cosine kernel for the learned embedding as the CNN effectively computes a *kernel feature map*, so that in principle it can approximate any kernel via the inner product. However, note that the expression normalizes vectors in L^2 norm, so that this inner product is the cosine of the angle between embedding vectors.

3.2 Cross Pixel Optical-Flow Similarity Loss Function

The constraint in eq. (1) requires kernels K_ϕ and K_f to be similar. We experiment with three loss functions for this task - kernel target alignment, cross-entropy, and cross-entropy reversed.

Kernel Target Alignment (KTA): KTA [7] is a conventional metric to measure the similarity between kernels. KTA for two kernel matrices K, K' , is given by

$$\mathcal{L}_{KTA}(K, K') = \sum_{pq} K_{pq} K'_{pq} / \sqrt{\sum_{pq} K_{pq}^2 \sum_{pq} K'^2_{pq}} \quad (3)$$

Cross-Entropy (CE): Our second loss function treats pixels as classes and kernel values as logits of a distribution over pixels. The cross entropy of these two distributions measures the distance between them. We compute this loss in two steps. First, we re-normalize each column $K_*(\cdot, q)$ of each kernel matrix into a probability distribution $S_*(\cdot, q)$. $S_f(\cdot, q)$ describes which image pixels p are likely to belong to the same segment as pixel q , according to optical-flow. $S_\phi(\cdot, q)$ describes the same but from the CNN embedding’s perspective. These distributions, arising from CNN and optical-flow kernels, are compared by using cross entropy, summed over columns:

$$\mathcal{L}_{CE}(\Theta) = - \sum_q \sum_p S_f(p, q) \log S_\phi(p, q). \quad (4)$$

Normalization uses the softmax operator. We reduce the contribution of diagonal terms in the kernel matrix before this normalization because each pixel is trivially similar to itself and would skew the softmax. Formally:

$$S_*(p, q) = \begin{cases} 1 / \left(\sum_{q' \neq p} \exp(K_*(p, q')) + 1 \right), & \text{if } p = q, \\ \exp(K_*(p, q)) / \left(\sum_{q' \neq p} \exp(K_*(p, q')) + 1 \right), & \text{if } p \neq q. \end{cases} \quad (5)$$

Cross-Entropy Reversed (CE-rev): Note that the particular ordering of distributions inside the cross entropy loss of eq. (4) treats the distribution induced by the optical-flow kernel (S_f) as ground truth. The embedding is tasked with inducing a kernel such that its corresponding distribution S_ϕ matches S_f . As an ablation study we also experiment with the order of distributions reversed. In other words we use,

$$\mathcal{L}_{CE-rev}(\Theta) = - \sum_q \sum_p S_\phi(p, q) \log S_f(p, q). \quad (6)$$

The intuition here is as follows: For a given pixel p , the distribution $S_\phi(\cdot, q)$ must be a delta distribution around $q' = \operatorname{argmax} S_f(\cdot, q)$. This is the natural effect of a flipped cross entropy loss. This delta distribution can be best approximated by aligning the two embeddings $\phi_p \cong \phi_{q'}$ and making all others anti-correlated $\phi_p \cong -\phi_q \forall q \neq q'$. Note however that this degenerate solution forces all $\phi_q, \phi_{q''}$ such that $q, q'' \neq q'$ to align as well. This coincidental alignment would, in general, significantly increase the loss function. Thus the embedding is forced to induce a non degenerate distribution S_ϕ . We consider it interesting to experiment with this loss.

Thus we have three cross pixel flow similarity losses - Kernel Target Alignment (CPFS-KTA), Cross Entropy (CPFS-CE) and Cross Entropy reversed (CPFS-CE-rev).

3.3 CNN Embedding Function

Lastly, we discuss the architecture of the CNN function ϕ itself. It maps the image into a per-pixel embedding. Recall that $\forall p \in \Omega, \phi(I, p | \Theta) \in \mathbb{R}^D$. We design the embedding CNN as a hypercolumn head [22] over a conventional CNN backbone such as AlexNet. The hypercolumn concatenates features from multiple depths so that our embedding can exploit high resolution details normally lost due to max-pooling layers. For training, we

use the sparsification trick of [31] and restrict prediction and loss computation to a few randomly sampled pixels in every iteration. This reduces memory consumption and improves training convergence as pixels in the same image are highly correlated and redundant; via sampling we can reduce this correlation and train more efficiently [4].

In more detail, the backbone is a CNN with activations at several layers: $\{\phi_{c_1}(I|\Theta), \dots, \phi_{c_n}(I|\Theta)\} \in \mathbb{R}^{H_1 \times W_1 \times D_1} \times \dots \times \mathbb{R}^{H_n \times W_n \times D_n}$. We follow [31] and interpolate values for a given pixel location and concatenate them to form a hypercolumn $\phi_H(I, p|\Theta) \in \mathbb{R}^{D_1 + \dots + D_n}$. The hypercolumn is then projected non-linearly to the desired embedding $\phi(I, p|\Theta) \in \mathbb{R}^D$ using a multi-layer perceptron (MLP). Details of the model architecture are discussed in section 4.1.

4 Experiments

We extensively assess our approach by demonstrating its effectiveness in learning features that we show as useful for several tasks. In order to make our results comparable to most of the related papers in the literature, we consider an AlexNet [30] backbone and four tasks: classification in ImageNet [47] and classification, detection, and segmentation in PASCAL VOC [13, 14].

4.1 Backbone Details

We adapt the AlexNet version used by Pathak *et al.* [42]. The modifications are minor (mostly related to padding), to make it suitable to attach a hypercolumn head. Sparse hypercolumns are built from the conv1, pool1, conv3, pool5 and fc7 AlexNet activations. Embeddings are generated using a multi-layer perceptron (MLP) with a single hidden layer and are L2-normalized. The embeddings are $D = 16$ dimensional (this number could be improved via cross validation, although this is expensive). The exact model specification, in the caffe text protocol buffer format (.prototxt), is included in the supplementary material.

4.2 Dataset

We train the above AlexNet model, using various CPFS losses, on a dataset of RGB-optical-flow image pairs. Inspired by the work of Pathak *et al.* [42], we built a dataset from $\sim 204k$ videos in the YFCC100m dataset [52]. The latter consists of Flickr videos made publicly-available under the creative commons license. We extract 8 random frames from each video and compute optical-flow between those at times t and $t + 5$ using the same (handcrafted) optical-flow method of [42, 33]. Overall, we obtain 1.6M image-flow pairs.¹ Example training sample crops along with optical-flow fields are shown in fig. 3. The noisy nature of both the images and optical-flow in such large-scale non-curated video collections makes it all the more challenging for self-supervision.

¹ Optical-flow is stored in fixed point 16bit PNG files similar to KITTI [18] for compression



Fig. 3: Image and optical-flow training pairs post scale-crop-flip data augmentation. The noisy nature of both images and optical-flows illustrate the challenges in using motion as a self-supervision signal. Optical-flow is visualized as a colour image using the toolbox of [6].

Optical-flow vectors (f_x, f_y) are normalized logarithmically to lie between $[-1, 1]$ during training, so that occasional large flows do not dominate learning. More precisely, the normalization is given by:

$$f' = \begin{bmatrix} \text{sign}(f_x) \min \left(1, \frac{\log(|f_x|+1)}{\log(M+1)} \right) \\ \text{sign}(f_y) \min \left(1, \frac{\log(|f_y|+1)}{\log(M+1)} \right) \end{bmatrix} \quad (7)$$

where M is a loose upper bound on the flow-magnitude set to 56.0 in our experiments.

Despite the large size of this data and aggressive data augmentation during training, AlexNet overfits on our self-supervision task. We use early stopping to reduce overfitting by monitoring the loss on a validation set. The validation set consists of 5000 image-flow pairs computed from the YouTube objects dataset [43]. Epic-Flow [46], with initial matches from Deep-Matching [58], was used to compute optical-flow for these frames.

4.3 Learning Details

We use the Adam optimizer [28] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and initial learning rate 10^{-4} . No weight decay is used because it resulted in our model reaching a worse local minima before over-fitting started. Pixels are sampled uniformly at random for the sparse hypercolumns. Sampling more pixels gives more information per image but also consumes more memory and is computationally expensive. We use 512 pixels per image to balance this trade-off. This allows for a large batch size of 96 frames. Scale, horizontal flip and crop augmentation with crop size 224×224 are applied during training. Color augmentation: shifting the hue by up to 0.1, random contrast between 0.2 – 1.8, random brightness by up to 0.12 (based on 0 – 1 normalised colours); is also applied. Parameter-free batch-normalization [23] is used throughout the network; the moving average mean and variance are absorbed into convolution kernels after self-supervised training, so that, for evaluation, AlexNet does not contain batch normalization layers. The implementation using TensorFlow [1] will be publicly available on GitHub.



Fig. 4: Per-pixel embeddings are visualized by randomly projecting them to RGB colors. From top to bottom: Validation images, RGB-mapped embeddings, optical-flow.

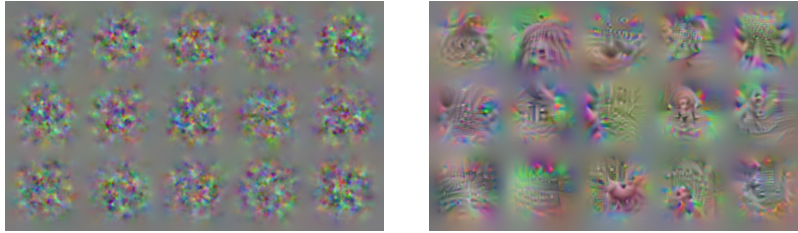


Fig. 5: Neuron maximization results for conv5 features [34]. Left: Neurons in a randomly initialized AlexNet. Right: Neurons in AlexNet trained using our approach: significantly more structure emerges.

4.4 Qualitative Results

In this section we visualize the AlexNet model pre-trained using the CPFS-CE loss function (Equation (4)).

Embedding Visualizations: While our learned pixel embeddings are not meant to be used directly (instead their purpose is to pre-train a neural network parametrization that can be transferred to other tasks), nevertheless it is informative to visualize them. Since embeddings are 16D, we first project them to 3D vectors via random projections and then map the resulting coordinates to RGB space. We show results on the YouTube objects validation set in fig. 4. Note that pixels on a salient foreground object tend to cluster together in the embedding (see, for example, the aircraft in column 4, the motor cyclist in column 3 and the cat in column 6).

Neuron Maximization: We use per-neuron activation maximization [34] to visualize individual neurons in the fifth convolutional layer (fig. 5). This figure presents the estimated optimal stimulus for each of these neurons, made interpretable using a natural image prior. We observe abstract patterns including a human form (row 2, column 4)

Table 1: Pascal VOC Comparison for three benchmarks: VOC2007-classification (column 4) %mAP, VOC2007-Detection (Column 5) %mAP and VOC2012-Segmentation (Column 6) %mIoU. The rows are grouped into four blocks (1) The limits of no-supervision and human supervision, (2) motion/video based self-supervision, (3) Our models and the baseline, (4) others. The third column [ref] indicates which publication the reported numbers are borrowed from. Full table in supplementary material.

	Method	Supervision	[Ref]	Cls.	Detection	Seg.
	Krizhevsky <i>et al.</i> [30]	Class Labels	[60]	79.9	56.8	48.0
	Random	-	[41]	53.3	43.4	19.8
Motion cues	Agrawal <i>et al.</i> [2]	Egomotion	[10]	63.1	43.9	-
	Jayaraman <i>et al.</i> [26]	Egomotion	[26]	-	41.7	-
	Lee <i>et al.</i> [32]	Time-order	[32]	63.8	46.9	-
	Misra <i>et al.</i> [35]	Time-order	[35]	-	42.4	-
	Pathak <i>et al.</i> [42]	Video-seg	[42],Self	61.0	50.2	-
	Wang <i>et al.</i> [55]	Track + Rank	[29, 55]	63.1	47.5	-
	CPFS-CE	Optical-flow	Self	64.2	50.8	41.4
	CPFS-CE-rev	Optical-flow	Self	63.6	49.9	39.5
	CPFS-KTA	Optical-flow	Self	65.3	50.5	41.5
	Ours direct cls.	Optical-flow	Self	63.2	46.1	38.8
	Other Cues - State of the art	Varied	[19, 38, 36]	73.3 [19]	55.5 [38]	40.6 [36]

that are obviously not present in a random network, suggesting that the representation may be learning concepts useful for general-purpose image analysis.

4.5 Quantitative Results

We follow standard practice in the self-supervised learning community and fine-tune the learned representation on various recognition benchmarks. We evaluate our features, pre-trained using various CPFS losses, by transfer learning on PASCAL VOC 2007 detection and classification [13], PASCAL VOC 2012 segmentation [14], and ILSVRC12 linear probing [60] (in the latter case, the representation is frozen). We provide details on the evaluation protocol next and compare against other self-supervised models with results reported for AlexNet-like-architectures in tables 1 and 2. Different from other approaches, we did not benefit from the re-balancing trick of [29] and hence we report results without it. This is possibly due to the use of batch normalization layers.

Baseline: Our main hypothesis is that the cross pixel flow similarity matching method, rather than the direct prediction of optical-flow, is more appropriate for exploiting optical-flow as a self-supervisory signal. We validate this hypothesis by comparing against a direct optical-flow prediction baseline, using the same CNN architecture as our method but a different loss function: while we use flow-similarity matching losses,

Table 2: ImageNet LSVRC-12 linear probing evaluation. A linear classifier is trained on the (downsampled) activations of each layer in the pretrained model. Top-1 accuracy is reported on ILSVRC-12 validation set. The column [ref] indicates which publication the reported numbers are borrowed from. We finetune Pathak *et al.*'s [42] model along with ours as they do not report these benchmark in their paper.

	Method	Supervision	[ref]	Conv1	Conv2	Conv3	Conv4	Conv5
	Krizhevsky <i>et al.</i> [30]	Class Labels	[61]	19.3	36.3	44.2	48.3	50.5
	Random	-	[61]	11.6	17.1	16.9	16.3	14.1
	Random-rescaled [29]	-	[29]	17.5	23.0	24.5	23.2	20.6
Motion	Pathak <i>et al.</i> [42]	Video-seg	Self	15.8	23.2	29.0	29.5	25.4
	CPFS-CE	Optical-Flow	Self	14.9	25.0	29.5	30.1	29.1
	CPFS-CE-rev	Optical-Flow	Self	15.3	24.8	27.7	27.8	26.3
	CPFS-KTA	Optical-Flow	Self	14.8	24.6	29.2	29.5	28.1
	Ours direct cls.	Optical-Flow	Self	14.0	23.0	26.4	26.7	24.8
Other cues	Doersch <i>et al.</i> [8]	Context	[61]	16.2	23.3	30.2	31.7	29.6
	Gidaris <i>et al.</i> [19]	Rotation	[19]	18.8	31.7	38.7	38.2	36.5
	Jenni <i>et al.</i> [27]	-	[27]	19.5	33.3	37.9	38.9	34.9
	Mundhenk <i>et al.</i> [36]	Context	[36]	19.6	31.4	37.0	37.8	33.3
	Noroozi <i>et al.</i> [37]	Jigsaw	[39]	18.2	28.8	34.0	33.9	27.1
	Noroozi <i>et al.</i> [39]	Counting	[39]	18.0	30.6	34.3	32.5	25.7
	Noroozi <i>et al.</i> [38]	Jigsaw++	[38]	18.2	28.7	34.1	33.2	28.0
	Noroozi <i>et al.</i> [38]	CC+Jigsaw++	[38]	18.9	30.5	35.7	35.4	32.2
	Pathak <i>et al.</i> [41]	In-Painting	[61]	14.1	20.7	21.0	19.8	15.5
	Zhang <i>et al.</i> [60]	Colorization	[61]	13.1	24.8	31.0	32.6	31.8
	Zhang <i>et al.</i> [61]	Split-Brain	[61]	17.7	29.3	35.4	35.2	32.8

this baseline does a standard per-pixel softmax cross entropy across 16 discrete optical-flow classes, once for each spatial dimension — x and y . To this end, since the flow is normalized in $[-1, 1]$ (eq. (7)), this interval is discretized uniformly. Note that direct L^2 regression of flow vectors is also possible, but did not work as well in preliminary experiments. This may be because continuous regression is usually harder for deep networks compared to classification, especially for ambiguous tasks. It was beneficial to use a faster initial learning rate of 0.01 for this baseline model.

VOC2007-detection: We finetune our AlexNet backbone end-to-end using the Fast-RCNN model [20] using code from [44] to obtain results for PASCAL VOC 2007 detection [13]. Finetuning follows the protocol of [29] to use multi-scale training and single-scale testing. We report mean average precision (mAP) in table 1 (col. 5) along with results of other self-supervised learning methods. We achieve the state-of-the-art among methods that use temporal information in videos for self-supervision. This table summarizes the state of the art among methods that use cues other than motion. Please see the supplementary material for a complete table of all relevant methods.

VOC2007 classification: We finetune our pretrained AlexNets to minimize the softmax cross-entropy loss over the PASCAL VOC 2007 *trainval* set for image classification across 20 Pascal classes. The initial learning rate is 10^{-3} and drops by a factor of 2 every 10k iterations for a total of 80k iterations and predictions are averaged over 10 random crops at test time in keeping with [29]. We use the code provided by [31] and report mean average precision (mAP) on VOC2007-test in the fourth column of table 1. We achieve state-of-the-art among methods that derive self-supervision from motion cues; in particular, we outperform [42] by a large margin.

ILSVRC12 linear probing: We follow the protocol and code of [61] to train a linear classifier on activations of our pre-trained network. The activation tensors produced by various convolutional layers (after ReLU) are down-sampled using bilinear interpolation to have roughly 9,000-10,000 elements before being fed into a linear classifier. The CNN parameters are frozen and only the linear classifier weights are trained on the ILSVRC-12 training set. Top-1 classification accuracy is reported on the ILSVRC-12 validation set (table 2). We achieve the state-of-the-art among motion-based self-supervision methods, except for “conv1” features.

VOC2012 segmentation: We use the two staged fine-tuning approach of [31] who finetune their AlexNet for semantic segmentation using a sparse hypercolumn head instead of the conventional FCN-32s head. We do so because it is a better fit for our sparse hypercolumn pre-training, although the hypercolumn itself is built using different layers (conv1 to conv5 and fc6 to fc7). Thus the MLP predicting the embedding ϕ from hypercolumn features is replaced with a new one before fine-tuning for segmentation. Also, our model has a fully convolutional structure but is pre-trained on a non-convolutional proxy task. We obtain a mere 31.3 %mIoU using FCN-32s. The training data consists of the PASCAL VOC 2012 [14] training set augmented with additional annotations from [21]. Thus the training-validation split has 10582 training images and 1449 validation images. Test results are reported as mean intersection-over-union (mIoU) scores on the PASCAL VOC 2012 validation set (Column 6 of table 1). We achieve the state of the art on this benchmark among all self-supervised learning methods, even ones that use other supervisory signals than motion (at the time of submission).

Other Architectures: We experimented with a VGG-16 [50] backbone² and followed the protocol of [31] to evaluated on VOC2007-classification and VOC12-segmentation. Our CPFS-CE model achieved 76.4% mAP for VOC2007 classification comparable to Larsson *et al.*’s 77.2% [31]; and 51.7% mIoU for VOC2012-segmentation which fell short of [31]’s 56.0%. VGG-16 has many more parameters than AlexNet. We argue that it may benefit from the extra 2.1M images used by [31] which might explain this performance gap.

4.6 Discussion

We can take home several messages from these experiments. First, in all cases our approach outperforms the baseline of predicting optical-flow directly. This is true for all

² Full model: ‘pool 1-5’, and ‘fc7’ (projected to 256 channels using a 1×1 convolution for faster training) constitute the hypercolumn head for pre-training on the dataset (Section 4.2).

three cross pixel flow similarity loss functions. This supports our hypothesis that direct single-frame optical-flow prediction is either too difficult due to its intrinsic ambiguity or a distraction from the goal of learning a powerful representation. It also supports our claim that predicting pairwise flow similarities partially addresses this ambiguity and allows us to learn useful CNN representations from optical-flow.

Second, the cross entropy loss is comparable in performance to kernel target alignment (KTA). We know that KTA aligns kernels uniformly and doing so is still highly ambiguous. Thus there is more room for improvement in the loss function design. Also, surprisingly, reversed cross entropy performs well although not as well as the other two.

Third, our method is the state of the art for self-supervision using motion cues. This is notable as our approach is significantly simpler than the previous state of the art [42]. By ingesting optical-flow information directly, it does not require data pre-processing via a video segmentation algorithm.

Finally, we also observe that all video/motion based methods for self-supervised learning are generally not as effective as methods that use other cues; particularly in image classification benchmarks. However, our approach still sets the overall state of the art for semantic image segmentation suggesting that the learned representation may be more suitable for per-pixel applications. Therefore further progress in this area of motion based self-supervision may be possible and is worth seeking. At the same time, authors of [9] find that combinations of different cues may result in the best performance.

5 Conclusion

We have presented a novel method for self-supervision using motion cues based on cross-pixel optical-flow similarity loss functions. We trained an AlexNet model using this scheme on a large unannotated video data-set. Visualizations of individual neurons in a deep layer and of the output embedding show that the representation captures structure in the image. We established the effectiveness of the resulting representation by transfer learning for several recognition benchmarks. Compared to the previous state of the art motion based method [42], our method works just as well and in some cases noticeably better despite a significant algorithmic simplification. We also outperform all other self-supervision strategies in semantic image segmentation (VOC12). This is reasonable as we train on a per-pixel proxy task on undistorted RGB images and use a hypercolumn model for fine-tuning. Finally, we see our contribution as an instance of self-supervision using multiple modalities, RGB and optical-flow, which poses our work as a special case of this broader area of research.

Acknowledgements

The authors gratefully acknowledge ERC IDIU, AIMS CDT (EPSRC EP/L015897/1) and AWS Cloud Credits for Research program. The authors thank Ankush Gupta and David Novotný for helpful discussions, and Christian Rupprecht, Fatma Guney and Ruth Fong for proof reading the paper. We thank Deepak Pathak for help with reproducing some of the results from [42].

References

1. Abadi, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
2. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: ICCV (2015)
3. Arandjelović, R., Zisserman, A.: Look, listen and learn. In: ICCV (2017)
4. Bansal, A., Chen, X., Russell, B., Gupta, A., Ramanan, D.: Pixelnet: Representation of the pixels, by the pixels, and for the pixels. arXiv:1702.06506 (2017)
5. Bojanowski, P., Joulin, A.: Unsupervised learning by predicting noise. In: ICML (2017)
6. Butler, D.J., et al.: A naturalistic open source movie for optical flow evaluation. In: ECCV (2014)
7. Cristianini, N., et al.: An Introduction to Support Vector Machines. Cambridge: CUP (2000)
8. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV (2015)
9. Doersch, C., et al.: Multi-task self-supervised visual learning. In: ICCV (2017)
10. Donahue, J., et al.: Adversarial feature learning. ICLR (2017)
11. Dosovitskiy, A., et al.: Flownet: Learning optical flow with convolutional networks. In: ICCV (2015)
12. Dosovitskiy, A., et al.: Discriminative unsupervised feature learning with exemplar convolutional neural networks. IEEE PAMI **38**(9), 1734–1747 (Sept 2016)
13. Everingham, M., et al.: The PASCAL Visual Object Classes Challenge 2007 Results (2007)
14. Everingham, M., et al.: The PASCAL Visual Object Classes Challenge 2012 Results (2012)
15. Faktor, A., Irani, M.: Video segmentation by non-local consensus voting. In: BMVC (2014)
16. Gan, C., Gong, B., Liu, K., Su, H., Guibas, L.J.: Geometry guided convolutional neural networks for self-supervised video representation learning. In: CVPR (2018)
17. Gao, R., Jayaraman, D., Grauman, K.: Object-centric representation learning from unlabeled videos. In: Proc. ACCV (2016)
18. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: CVPR (2012)
19. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised Representation Learning by Predicting Image Rotations. In: Proc. ICLR (2018)
20. Girshick, R.B.: Fast R-CNN. In: ICCV (2015)
21. Hariharan, B., et al.: Semantic contours from inverse detectors. In: ICCV (2011)
22. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: CVPR. pp. 447–456 (2015)
23. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
24. Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Learning visual groups from co-occurrences in space and time. ICLR Workshop (2015)
25. Jayaraman, D., Grauman, K.: Slow and steady feature analysis: higher order temporal coherence in video. In: CVPR. pp. 3852–3861 (2016)
26. Jayaraman, D., et al.: Learning image representations tied to ego-motion. In: ICCV (2015)
27. Jenni, S., Favaro, P.: Self-supervised feature learning by learning to spot artifacts. In: CVPR (2018)
28. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014)
29. Krähenbühl, P., et al.: Data-dependent initializations of convolutional neural networks. ICLR (2016)
30. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS. pp. 1106–1114 (2012)

31. Larsson, G., Maire, M., Shakhnarovich, G.: Colorization as a proxy task for visual understanding. In: CVPR (2017)
32. Lee, H.Y., Huang, J.B., Singh, M.K., Yang, M.H.: Unsupervised representation learning by sorting sequence. In: ICCV (2017)
33. Liu, C.: Beyond Pixels: Exploring New Representations and Applications for Motion Analysis. Ph.D. thesis, Massachusetts Institute of Technology, USA (2009)
34. Mahendran, A., Vedaldi, A.: Visualizing deep convolutional neural networks using natural pre-images. IJCV pp. 1–23 (2016)
35. Misra, I., Zitnick, C.L., Hebert, M.: Shuffle and Learn: Unsupervised Learning using Temporal Order Verification. In: ECCV (2016)
36. Mundhenk, T., Ho, D., Y. Chen, B.: Improvements to context based self-supervised learning. In: CVPR (2017)
37. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV. pp. 69–84. Springer (2016)
38. Noroozi, M., Vinjimoor, A., Favaro, P., Pirsiaavash, H.: Boosting self-supervised learning via knowledge transfer. In: CVPR (2018)
39. Noroozi, M., et al.: Representation learning by learning to count. In: ICCV (2017)
40. Owens, A., et al.: Ambient sound provides supervision for visual learning. In: ECCV (2016)
41. Pathak, D., et al.: Context encoders: Feature learning by inpainting. In: CVPR (2016)
42. Pathak, D., et al.: Learning features by watching objects move. In: CVPR (2017)
43. Prest, A., et al.: Learning object class detectors from weakly annotated video. In: CVPR (2012)
44. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS (2015)
45. Ren, Z., Lee, Y.J.: Cross-domain self-supervised multi-task feature learning using synthetic imagery. In: CVPR (2018)
46. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: Epicflow: Edge-preserving interpolation of correspondences for optical flow. In: CVPR (2015)
47. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
48. de Sa, V.R.: Learning classification with unlabeled data. In: NIPS. pp. 112–119 (1994)
49. Sermanet, P., et al.: Time-contrastive networks: Self-supervised learning from video. In: Proc. Intl. Conf. on Robotics and Automation (2018)
50. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Arxiv 1409.1556 (2014)
51. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: ICML (2015)
52. Thomee, B., et al.: Yfcc100m: the new data in multimedia research. ACM (2016)
53. Todorovic, D.: Gestalt principles. Scholarpedia **3**(12), 5345 (2008), revision #91314
54. Walker, J.: Data-Driven Visual Forecasting. Ph.D. thesis, Carnegie Mellon University (2018)
55. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: ICCV. pp. 2794–2802 (2015)
56. Wang, X., He, K., Gupta, A.: Transitive invariance for self-supervised visual representation learning. In: ICCV. pp. 2794–2802 (2017)
57. Wei, D., et al.: Learning and using the arrow of time. In: CVPR. pp. 8052–8060 (2018)
58. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: ICCV. pp. 1385–1392 (2013)
59. Xue, T., Wu, J., Bouman, K.L., Freeman, W.T.: Visual dynamics: Stochastic future generation via layered cross convolutional networks. In: IEEE PAMI (2018)
60. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV (2016)
61. Zhang, R., Isola, P., Efros, A.A.: Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In: CVPR (2017)