Cross Pixel Optical Flow Similarity for Self-Supervised Learning

Aravindh Mahendran, James Thewlis, and Andrea Vedaldi

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

Abstract. We propose a novel method to supervise convolutional neural networks without manual supervision, using instead motion cues obtained by measuring optical flow in video streams. 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, competitive results for self-supervision in general, and is overall state of the art in self-supervised pretraining for semantic image segmentation, as demonstrated on standard benchmarks.

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. While there is a remarkable variety of self-supervised learning methods, they are all based on the same basic premise, which is to identify problems that can be used to train good 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 (section 2). Furthermore, the recent work of [1] shows that combining several such cues further helps performance.

In this paper, we consider the case of *self-supervision using motion cues*. 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

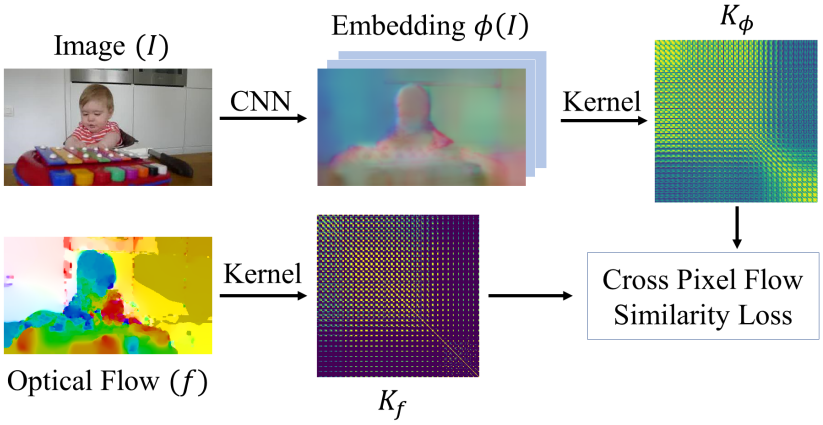

Fig. 1: We propose a novel method to exploit motion information represented as optical flow, to supervise the learning of deep CNNs. We learn a network that predicts per-pixel embeddings $\phi(I)$ such that the kernel computed over these embeddings (K_ϕ) is similar to that over corresponding optical-flow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

motion regularity and hence predictability: pixels that belong to the same object are much more likely to "move together" than pixels that do not.

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, and overall an overkill if the goal is to learn a good general-purpose image representation for image analysis. Instead, the previous most effective method for self-supervision using motion cues [2] 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 [2] 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 to address 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*, regardless of the specific direction or velocity. This is obtained by encouraging the inner product of the learned pixel embeddings to correlate with the similarity between their corresponding optical flow vectors. 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 times (for example, often objects stand still, so their velocities are all zero). We addressed this second challenge by using a robust loss that captures pixel grouping probabilistically rather than deterministically.

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 [2], 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 strong performance in some cases and state-of-the-art performance for semantic image segmentation.

The overall conclusion (section 5) is that our method significantly simplifies leveraging motion cues for self-supervision and does so 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 [1].

2 Related Work

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

Video/Motion Based: LSTM RNNs can be trained to predict future frames in a video [3]. 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 [2]. They use an off-the-shelf video segmentation method [4] 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 [5]. 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 [6]. This was used as a pretraining strategy by [7] 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; [7] evaluate on dynamic scene and action recognition.

Temporal context is a powerful signal. [8,9,10] learn to predict the correct ordering of frames. [11] exploit both temporal and spatial co-occurrence statistics to learn visual groups. [12] extend slow feature analysis using higher order temporal coherence. [13] track patches in a video to supervise their embedding via a triplet loss while [14] do the same but for spatio-temporally matched region proposals. Temporal context is applied in the imitation learning setting by Time Contrastive Networks [15].

Videos contain more than just temporal information. Some methods exploit audio channels by prediction audio from video frames [16,17]. [18] 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 [19,20]. [21] use videos along with spatial context pretraining [22] to construct an image graph. Transitivity in the graph is exploited to learn representations with suitable invariances.

Colorization: [23,24,25] predict colour information given greyscale input and show competitive pre-training performance. A generalization to arbitrary pairs of modalities was proposed in [26].

Spatial Context: [27] solve the in-painting problem, where a network is tasked with filling-in partially deleted parts of an image. [22] predict the relative position of two patches extracted from an image. In a similar spirit, [28,29] solve a jigsaw puzzle problem. [29] 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 by [30] obtains state-of-the-art results on several benchmarks.

Adversarial/Generative: BiGAN based pretrained models [31] show competitive performance on various recognition benchmarks. [32] adversarially learn to generate and spot defects. [33] obtain self-supervision from synthetic data

and adapt their model to the domain of real images by training against a discriminator. [34] 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 [35]. An extensive review of these and more recent unsupervised generative models is beyond the scope of our paper.

Transformations: [36] create surrogate classes by applying a set of transformations to each image and learn to become invariant to them. [37] 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 [38]. [39] use correspondences obtained from synthetic warps to learn a dense image representation.

Others: A combination of self-supervision approaches was explored by [1]. 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. [40] propose a mix-and-match tuning strategy as a precursor to finetuning on the target domain. Their approach can be applied to any pretrained model and achieves impressive results for PASCAL VOC 2012 semantic segmentation. Another widely-applicable trick that helps in transfer learning is the re-balancing method of [41]. Lastly, our optical-flow classification baseline is based on the work of [42]. 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 can be used for classification, detection, segmentation and other tasks with minimal supervision.

Our goal is to learn the parameters Θ of a neural network ϕ that maps a single image or frame $I : \mathbb{R}^2 \supset \Omega \rightarrow \mathbb{R}^3$ to a field of pixel embeddings $\phi(I, p|\Theta) \in \mathbb{R}^D$, one for each pixel $p \in \Omega$. In order to learn this embedding, which is extracted from a *single frame*, we task our neural network with *predicting* the motion present in the corresponding video, represented as optical flow. However, since predicting flow vectors directly is too ambiguous, we propose instead to require the *similarity* between *pairs* of embedding vectors to align to 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's principle of *common fate*.

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

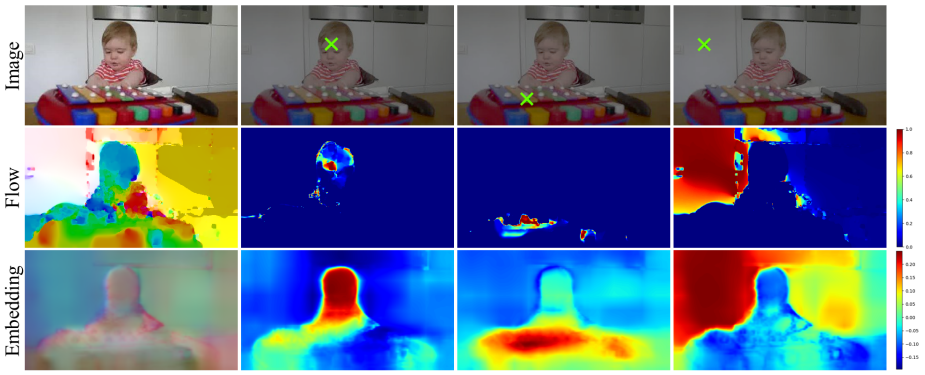

Fig. 2: Visualization of flow (second row) and embedding (third row) kernels. For three pixels p, we plot the row $K(p, \cdot)$ reshaped as an image, showing which pixels go together from the kernel's perspective. Note the localized nature of the flow kernel which is obtain by setting a low bandwidth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

the kernel matrices

$$\forall p, q \in \Omega : \quad K_\phi\Big(\phi(I, p|\Theta), \phi(I, q|\Theta)\Big) \approxeq K_f(f_p, f_q) \tag{1}$$

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 (1) into a loss function. The rest of the section discusses these choices.

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

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

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 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. Observe that reducing kernel bandwidth (σ) results in a localized kernel that pushes our embeddings harder to distinguish between different movable objects. The value of σ is learned along with the weights of the CNN in the optimization. This localized kernel, with learned $\sigma^2 = 0.0067$, 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. This normalization is key as it guarantees that K_Φ is maximum when the embeddings being compared are identical (the Gaussian kernel does so automatically).

Cross Pixel Optical-Flow Similarity Loss function: Constraint (1) requires kernels K_ϕ and K_f to be similar. A conventional metric to measure the similarity between kernels is *kernel target alignment* (KTA) [43] which, for two kernel matrices K, K', is given by $\sum_{pq} K_{pq} K'_{pq} / \sqrt{\sum_{pq} K_{pq}^2 \sum_{pq} {K'_{pq}}^2}$. In this manner, KTA and analogous metrics require kernels to match *everywhere*. In our case, however, this is too strong a requirement. Even if the embedding function ϕ correctly groups pixels that belong to different movable objects, K_ϕ may still not match the optical flow kernel K_f. The reason is that in many video frames objects may not move with a distinctive pattern or may not move at all, so that no corresponding grouping can be detected in the measured flow field. KTA failed to produce reasonable embeddings in our preliminary experiments.

This motivated us to research a kernel similarity criterion more suitable for our task. The key idea is to relax the correspondence between kernels to hold probabilistically. We do so in two steps. First, we re-normalize each column $K_*(\cdot, q)$ of each kernel matrix into a probability distribution $S_*(\cdot, q)$. Each distribution $S_*(\cdot|q)$ tells which image pixels p are likely to belong to the same segment as pixel q according to a particular cue, either the CNN embedding or the optical flow. Then, the distributions arising from CNN and optical flow kernels are compared by using cross entropy, summed over columns:

$$\mathcal{L}(\Theta) = -\sum_q \sum_p S_f(p,q) \log S_\phi(p,q). \tag{3}$$

Normalization uses the softmax operator after reducing the contribution of diagonal terms in the kernel matrix (because each pixel is trivially similar to itself and would skew the softmax normalization). Formally, for optical flow we have:

$$S_f(p,q) = \begin{cases} \frac{1}{\sum_{q' \neq p} \exp(K_f(p,q')) + 1}, & \text{if } p = q, \\ \frac{\exp(K_f(p,q'))}{\sum_{q' \neq p} \exp(K_f(p,q')) + 1}, & \text{if } p \neq q. \end{cases} \tag{4}$$

Similarly, for the CNN embedding we have:

$$S_\phi(p,q) = \begin{cases} \frac{e^{-3/4}}{\sum_{q' \neq p} \exp(K_\phi(p,q')) + e^{-3/4}}, & \text{if } p = q, \\ \frac{\exp(K_\phi(p,q'))}{\sum_{q' \neq p} \exp(K_\phi(p,q')) + e^{-3/4}}, & \text{if } p \neq q. \end{cases} \tag{5}$$

Note that the $p = q$ and $p = q'$ cases contribute $e^{-3/4}$ for the embedding kernel's softmax. This is because of the $1/4$ scaling used in the cosine similarity kernel (2).

CNN embedding function: Lastly, we discuss the architecture of the CNN function ϕ itself. We design the embedding CNN as a hypercolumn head 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 [23] 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 [42].

In more detail, the backbone is a CNN with activations at several layers: $\{\phi_{c_1}(I|\Theta), \cdots, \phi_{c_n}(I|\Theta)\} \in \mathbb{R}^{H_1 \times W_1 \times D_1} \times \cdots \times \mathbb{R}^{H_n \times W_n \times D_n}$. We follow [24] and interpolate values at a given pixel location and concatenate them to form a hypercolumn $\phi_H(I, p|\Theta) \in \mathbb{R}^{D_1 + \cdots + D_n}$. 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 useful for several tasks. In order to make our results comparable to most of the related papers in the literature, we consider an AlexNet [35] backbone and four tasks: classification in ImageNet [44] and classification, detection, and segmentation in PASCAL VOC [45].

4.1 Backbone details

We adapt the AlexNet version used by Pathak et al. [2]. The modifications are minor (mostly related to padding) and required to make it possible 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 was selected after a preliminary investigation and could be improved via cross validation, although this is expensive). The exact model specifications are included in supplementary material.

4.2 Dataset

We train the above AlexNet model on a dataset of RGB-optical flow image pairs. Inspired by the work of Pathak et al. [2], we built a dataset from $\sim 204k$ videos in the YFCC100m dataset [46]. 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 [2,47]. Overall, we obtain 1.6M

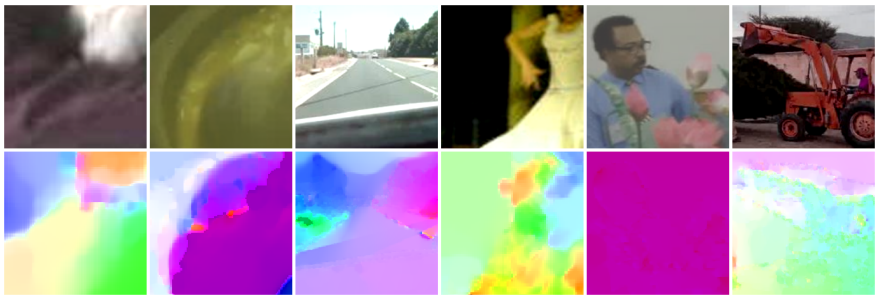

Fig. 3: Image and optical-flow training pairs post scale-crop-flip data augmentation. The noisy nature of both images and optical-flows illustrate the challenges in using motion as a self-supervision signal.

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.

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:

$$f' = \begin{bmatrix} \text{sign}(f_x) \min\left(1, \frac{\log(|f_x|+1)}{\log(M+1)}\right) \\ \\ \text{sign}(f_y) \min\left(1, \frac{\log(|f_y|+1)}{\log(M+1)}\right) \end{bmatrix} \tag{6}$$

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 over-fitting by monitoring the loss on a validation set. The validation set consists of 5000 image-flow pairs computed from the YouTube objects dataset [48]. Epic-Flow [49] + Deep-Matching [50] was used to compute optical-flow for these frames.

4.3 Learning Details

AlexNet is trained using the Adam optimizer [51] with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ and initial learning rate set to 10^{-4}. No weight decay is used because it worsened the minima reached by our models before overfitting started. Pixels are sampled uniformly at random for the sparse hypercolumns. Sampling more

[1] The dataset occupies more than 1TB of space and does not easily fit in fast memory such as an SSD for training. We addressed this problem by using a fixed point 16bit representation of optical flow and storing it as PNG images, with dramatic compression and negligible residual error. The compressed data occupies 431GB.

Fig. 4: Per-pixel embeddings are visualized by randomly projecting them to RGB colors. We show example embeddings generated by our model on frames from the YouTube objects dataset. From top to bottom: The original validation images, RGB-mapped embeddings, and optical-flow fields. Best viewed in color on screen.

pixels gives more information per image but also consumes more memory and is computationally expensive making each iteration slower. We use 512 pixels per image to balance this trade-off. This reduces memory consumption and allows for a large batch size of 96 frames. Scale, horizontal flip and crop augmentation are applied during training. The network is trained on crops of size 224×224. Parameter-free batch-normalization [52] 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 full implementation using TensorFlow [53] will be released to the public upon acceptance.

4.4 Qualitative Results

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 them 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 cats in columns 1, 3 and 4, the aircraft in column 6 and the motor-cyclist in column 8).

We also use per-neuron activation maximization [54] to visualize individual neurons in the fifth convolutional layer (fig. 5). We observe abstract patterns including a human form (row 1, column 5) that are obviously not present in a random network, suggesting that the representation may be learning concepts useful for general-purpose image analysis.

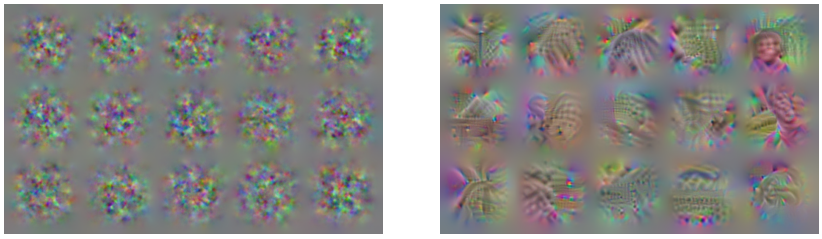

Fig. 5: Each image in the grid corresponds to the estimated optimal excitatory stimulus for a neuron in conv5 feature space. This visualization is obtained using the regularized neuron maximization method of [54]. Left: Neurons in a randomly initialized AlexNet. Right: Neurons in AlexNet trained using our approach: significantly more structure emerges.

4.5 Quantitative results

We follow the standard practice in the self-supervised learning community and fine-tune the learned representation on various recognition benchmarks. We evaluate our features on the PASCAL VOC 2007 detection and classification [45], PASCAL VOC 2012 segmentation, and ILSVRC12 linear probing benchmarks [25] (in the latter case, the representation is frozen). We provide details on the evaluation protocol next and compare against other self-supervised models in table 1 and table 2. Differently from other approaches, we did not benefit from the re-balancing trick of [41] and report results without it.

Baseline: Our main hypothesis is that our similarity matching method, rather than a direct prediction of optical flow, is a more appropriate mechanism to exploit optical flow information as a supervisory signal. We validate this hypothesis by comparing against a direct optical-flow prediction baseline, using the same CNN architecture but a different loss function: while we use a flow-similarity matching loss, 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. (6)), 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 as 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.

VOC2007-Detection: We finetune our AlexNet backbone end-to-end using the Fast-RCNN model [56] to obtain results for PASCAL VOC 2007 detection [45]. Finetuning is performed for 150k iterations, were the learning rate drops by a 10^{th} every 50k iterations. The initial learning rate is set to 0.002, weight decay to 5×10^{-4}, train-set is VOC2007-train+val, test-set is VOC2007-test. Following the guidelines of [41], we 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 state-of-the-art

Table 1: Pascal VOC Comparison for three benchmarks: VOC2007-classification (column 4) %mAP, VOC2007-Detection (Column 5) %mAP and VOC2012-Segmentation (Column 6) %mIU. The rows are grouped into four blocks (0) The limits of no-supervision and human supervision, (1) motion/video based self-supervision, (2) Our model and the baseline, (3) others. The third column [ref] indicates which publication the reported numbers are borrowed from. *Zhan et al. contribute a different mix-and-match tuning strategy which transfers better to the target domain compared to finetuning. This is orthogonal to the efforts of finding a good self-supervision method and is therefore not counted when marking the state-of-the-art in **bold.**

	Method	Supervision	[Ref]	Cls.	Detection	Seg.
	Krizhevsky et al. [35]	Class Labels	[25]	79.9	56.8	48.0
	Random	-	[27]	53.3	43.4	19.8
Motion cues	Agrawal et al. [5]	Egomotion	[31]	63.1	43.9	-
	Jayaraman et al. [20]	Egomotion	[20]	-	41.7	-
	Lee et al. [10]	Time-order	[10]	63.8	46.9	-
	Misra et al. [8]	Time-order	[8]	-	42.4	-
	Pathak et al. [2]	Video-seg	[2],Self	61.0	50.2	-
	Wang et al. [13]	Ranking	[41,13]	63.1	47.5	-
	Ours	Optical-flow	Self	**64.4**	**50.3**	**41.4**
	Ours direct reg.	Optical-flow	Self	61.4	44.0	37.1
Other cues	Bojanowski et al. [34]	-	[34]	65.3	49.4	-
	Doersch et al. [55]	Context	[31]	65.3	51.1	-
	Donahue et al. [31]	-	[31]	60.3	46.9	35.2
	Gidaris et al. [37]	Rotation	[37]	**73.0**	54.4	39.1
	Krahenbuhl et al. [41]	-	[41,31]	56.6	45.6	32.6
	Larssons et.al. [24]	Colorization	[24]	65.9	-	38.4
	Mundhenk et al. [30]	Context	[30]	69.3	55.2	**40.6**
	Noroozi et al. [28]	Jigsaw	[28]	67.6	53.2	37.6
	Noroozi et al. [38]	Counting	[38]	67.7	51.4	36.6
	Noroozi et al. [29]	Jigsaw++	[29]	69.8	**55.5**	38.1
	Noroozi et al. [29]	CC+Jigsaw++	[29]	69.9	55.0	40.0
	Owens et al. [17]	Sound	[26,17]	61.3	44.1	-
	Pathak et al. [27]	In-painting	[27]	56.5	44.5	29.7
	Jenni et al. [32]	-	[32]	69.8	52.5	38.1
	Zhang et al. [25]	Colorization	[25]	65.9	46.9	35.6
	Zhang et al. [26]	Split-Brain	[26]	67.1	46.7	36.0
	Zhan et al. [40]*	Colorization	[40]	-	-	42.8
	Zhan et al. [40]*	Puzzle	[40]	-	-	44.5

performance among methods that use temporal information in videos for self-supervision.

VOC2007 classification: We finetune our pretrained AlexNet to minimize the softmax cross-entropy loss over the PASCAL VOC 2007 *trainval* set. The

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 model along with ours as they do not report these benchmark in their paper.

	Method	Supervision	[ref]	Conv1	Conv2	Conv3	Conv4	Conv5
	Krizhevsky et.al. [35]	Class Labels	[26]	19.3	36.3	44.2	48.3	50.5
	Random	-	[26]	11.6	17.1	16.9	16.3	14.1
	Random-rescaled [41]	-	[41]	17.5	23.0	24.5	23.2	20.6
Motion	Pathak et.al. [2]	Video-seg	Self	**15.8**	23.2	**29.0**	**29.5**	25.4
Motion	Ours	Optical-Flow	Self	15.0	**24.8**	28.9	29.4	**28.0**
Motion	Ours direct reg.	Optical-Flow	Self	14.0	22.6	25.3	25.0	23.0
Other cues	Doersch et.al. [22]	Context	[26]	16.2	23.3	30.2	31.7	29.6
Other cues	Gidaris et.al. [37]	Rotation	[37]	18.8	31.7	**38.7**	38.2	**36.5**
Other cues	Jenni et.al. [32]	-	[32]	19.5	**33.3**	37.9	**38.9**	34.9
Other cues	Mundhenk et.al. [30]	Context	[30]	**19.6**	31.4	37.0	37.8	33.3
Other cues	Noroozi et.al. [28]	Jigsaw	[38]	18.2	28.8	34.0	33.9	27.1
Other cues	Noroozi et.al. [38]	Counting	[38]	18.0	30.6	34.3	32.5	25.7
Other cues	Noroozi et.al. [29]	Jigsaw++	[29]	18.2	28.7	34.1	33.2	28.0
Other cues	Noroozi et.al. [29]	CC+Jigsaw++	[29]	18.9	30.5	35.7	35.4	32.2
Other cues	Pathak et.al. [27]	In-Painting	[26]	14.1	20.7	21.0	19.8	15.5
Other cues	Zhang et.al. [25]	Colorization	[26]	13.1	24.8	31.0	32.6	31.8
Other cues	Zhang et.al. [26]	Split-Brain	[26]	17.7	29.3	35.4	35.2	32.8

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 [41]. We use the code provided by [24] and report mean average precision on VOC2007-test in the fourth column of table 1. We achieve state-of-the-art self-supervision using motion cues; in particular, we outperform [2] by a good margin.

ILSVRC12 linear probing: We follow the protocol and code of [26] 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 perform comparably to the best motion-based self-supervision method of [2] (slightly worse or better depending on the layer), but using other types of cues achieves stronger results in this case.

VOC2012 segmentation: We use the two staged finetuning approach of [24] who finetune their AlexNet for semantic segmentation using a sparse hypercol-

umn head instead of the conventional FCN-32s head. We do so because it is a better fit for our sparse hypercolumn pretraining, 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 discarded before finetuning for segmentation. The training data consists of the PASCAL VOC 2012 train set augmented with annotations from [57]. Test results are reported as mean intersection-over-union (mIU) scores on the PASCAL VOC 2012 validation set (Column 6 of table 1). We are the state of the art on this benchmark among all self-supervised learning methods, even ones that use other supervisory signals than motion.

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 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 good representation. It also shows that our approach of predicting pairwise flow similarities successfully addresses this ambiguity and allows to learn good CNN representations from optical flow.

Second, our method is generally as strong as the state of the art for self-supervision using motion cues represented by the approach of [2]. In fact, our approach outperforms the latter by a good margin in PASCAL VOC07 classification. This is notable as our approach is significantly simpler: by ingesting optical flow information directly, it does not require to pre-process the data via an ad-hoc video segmentation algorithm.

Finally, we also found out that all video/motion based methods for self supervised learning are sometimes not as good as methods that use other cues (but our approach still sets the overall state-of-the-art for semantic image segmentation). This suggests that further progress in this area is possible and worth seeking. At the same time, [1] find that the *combination* of different cues may in practice result in the best performance; in this sense, our approach, by significantly simplifying the use of motion cues, can make it much easier to design multi-task networks that can leverage motion together with other complementary methods.

5 Conclusion

We have presented a novel method for self-supervision using motion cues based on a cross-pixel optical-flow similarity loss function. 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 [2], 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.

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.

Acknowledgments

We are grateful for support by ERC 638009-IDIU and EPSRC AIMS CDT.

References

1. Doersch, C., et al.: Multi-task self-supervised visual learning. In: ICCV. (2017)
2. Pathak, D., et al.: Learning features by watching objects move. In: CVPR. (2017)
3. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: Proc. ICML. (2015)
4. Faktor, A., Irani, M.: Video segmentation by non-local consensus voting. In: Proc. BMVC. Volume 2. (2014) 8
5. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: ICCV. (2015)
6. Dosovitskiy, A., et al.: Flownet: Learning optical flow with convolutional networks. In: Proc. ICCV. (2015)
7. 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)
8. Misra, I., Zitnick, C.L., Hebert, M.: Shuffle and Learn: Unsupervised Learning using Temporal Order Verification. In: Proc. ECCV. (2016)
9. Wei, D., Lim, J.J., Zisserman, A., Freeman, W.T.: Learning and using the arrow of time. In: Proc. CVPR. (2018) 8052–8060
10. Lee, H.Y., Huang, J.B., Singh, M.K., Yang, M.H.: Unsupervised representation learning by sorting sequence. In: Proc. ICCV. (2017)
11. Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Learning visual groups from co-occurrences in space and time. ICLR Workshop (2015)
12. Jayaraman, D., Grauman, K.: Slow and steady feature analysis: higher order temporal coherence in video. In: Proc. CVPR. (2016) 3852–3861
13. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: Proc. ICCV. (2015) 2794–2802
14. Gao, R., Jayaraman, D., Grauman, K.: Object-centric representation learning from unlabeled videos. In: Proc. ACCV. (2016)
15. Sermanet, P., et al.: Time-contrastive networks: Self-supervised learning from video. In: Proc. Intl. Conf. on Robotics and Automation. (2018)
16. de Sa, V.R.: Learning classification with unlabeled data. In: NIPS. (1994) 112–119
17. Owens, A., Wu, J., McDermott, J.H., Freeman, W.T., Torralba, A.: Ambient sound provides supervision for visual learning. In: Proc. ECCV, Springer (2016) 801–816
18. Arandjelović, R., Zisserman, A.: Look, listen and learn. In: Proc. ICCV. (2017)
19. Jayaraman, D., Grauman, K.: Learning image representations tied to egomotion from unlabeled video. IJCV **125** (2017) 136–161
20. Jayaraman, D., Grauman, K.: Learning image representations tied to ego-motion. In: Proc. ICCV. (2015) 1413–1421

21. Wang, X., He, K., Gupta, A.: Transitive invariance for self-supervised visual representation learning. In: Proc. ICCV. (2017) 2794–2802
22. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: Proc. ICCV. (2015) 1422–1430
23. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: Proc. ECCV, Springer (2016) 577–593
24. Larsson, G., Maire, M., Shakhnarovich, G.: Colorization as a proxy task for visual understanding. In: Proc. CVPR. (2017)
25. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV. (2016)
26. Zhang, R., Isola, P., Efros, A.A.: Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In: Proc. CVPR. (2017)
27. Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: Proc. CVPR. (2016)
28. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: Proc. ECCV, Springer (2016) 69–84
29. Noroozi, M., Vinjimoor, A., Favaro, P., Pirsiavash, H.: Boosting self-supervised learning via knowledge transfer. In: Proc. CVPR. (2018)
30. Mundhenk, T., Ho, D., Y. Chen, B.: Improvements to context based self-supervised learning. In: Proc. CVPR. (2017)
31. Donahue, J., et al.: Adversarial feature learning. ICLR (2017)
32. Jenni, S., Favaro, P.: Self-supervised feature learning by learning to spot artifacts. In: Proc. CVPR. (2018)
33. Ren, Z., Lee, Y.J.: Cross-domain self-supervised multi-task feature learning using synthetic imagery. In: Proc. CVPR. (2018)
34. Bojanowski, P., Joulin, A.: Unsupervised learning by predicting noise. In: Proc. ICML, PMLR (2017) 517–526
35. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS. (2012) 1106–1114
36. Dosovitskiy, A., et al.: Discriminative unsupervised feature learning with exemplar convolutional neural networks. IEEE PAMI **38** (2016) 1734–1747
37. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised Representation Learning by Predicting Image Rotations. In: Proc. ICLR. (2018)
38. Noroozi, M., et al.: Representation learning by learning to count. In: ICCV. (2017)
39. Novotný, D., Albanie, S., Larlus, D., Vedaldi, A.: Self-supervised learning of geometrically stable features through probabilistic introspection. In: CVPR. (2018)
40. Zhan, X., Liu, Z., Luo, P., Tang, X., Loy, C.C.: Mix-and-match tuning for self-supervised semantic segmentation. In: AAAI. (2018)
41. Krähenbühl, P., Doersch, C., Donahue, J., Darrell, T.: Data-dependent initializations of convolutional neural networks. ICLR (2016)
42. 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)
43. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge: CUP (2000)
44. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
45. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results (2007)
46. Thomee, B., et al.: Yfcc100m: the new data in multimedia research. ACM (2016)
47. Liu, C.: Beyond Pixels: Exploring New Representations and Applications for Motion Analysis. PhD thesis, Massachusetts Institute of Technology, USA (2009)
48. Prest, A., Leistner, C., Civera, J., Schmid, C., Ferrari, V.: Learning object class detectors from weakly annotated video. In: CVPR. (2012)

49. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: Epicflow: Edge-preserving interpolation of correspondences for optical flow. In: Proc. CVPR. (2015)
50. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: Proc. ICCV. (2013) 1385–1392
51. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
52. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proc. ICML. (2015)
53. Abadi, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
54. Mahendran, A., Vedaldi, A.: Visualizing deep convolutional neural networks using natural pre-images. IJCV (2016) 1–23
55. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: Proc. CVPR. (2015)
56. Girshick, R.B.: Fast R-CNN. In: Proc. ICCV. (2015)
57. Hariharan, B., et al.: Semantic contours from inverse detectors. In: ICCV. (2011)