Hyperpixel Flow: Semantic Correspondence with Multi-layer Neural Features

Juhong Min, Jongmin Lee, Jean Ponce, Minsu Cho

Hyperpixel Flow: Semantic Correspondence with Multi-layer Neural Features

Juhong Min[1,2] Jongmin Lee[1,2] Jean Ponce[3,4] Minsu Cho[1,2]
[1]POSTECH [2]NPRC* [3]Inria [4]DI ENS†

http://cvlab.postech.ac.kr/research/HPF/

Abstract

Establishing visual correspondences under large intra-class variations requires analyzing images at different levels, from features linked to semantics and context to local patterns, while being invariant to instance-specific details. To tackle these challenges, we represent images by "hyperpixels" that leverage a small number of relevant features selected among early to late layers of a convolutional neural network. Taking advantage of the condensed features of hyperpixels, we develop an effective real-time matching algorithm based on Hough geometric voting. The proposed method, hyperpixel flow, sets a new state of the art on three standard benchmarks as well as a new dataset, SPair-71k, which contains a significantly larger number of image pairs than existing datasets, with more accurate and richer annotations for in-depth analysis.

1. Introduction

Establishing visual correspondences under large intra-class variations, *i.e.*, matching scenes depicting different instances of the same object categories, remains a challenging problem in computer vision. It requires analyzing scenes at different levels, from features linked to semantics and context to local image patterns, while being invariant to irrelevant instance-specific details. Recent methods have addressed this problem using deep convolutional features. Many of them [5, 16, 24, 42] formulate this task as local region matching and learn to assign a local region in an image to a correct match in another image. Others [23, 41, 42, 45] cast it as image alignment and learn to regress the parameters of global geometric transformation, *e.g.*, using an affine or thin plate spline model [8]. These methods, however, mainly perform the prediction based on the output of the last convolutional layer, and fail to fully exploit the different levels of semantic features available to resolve the severe ambiguities in matching linked with intra-class variations.

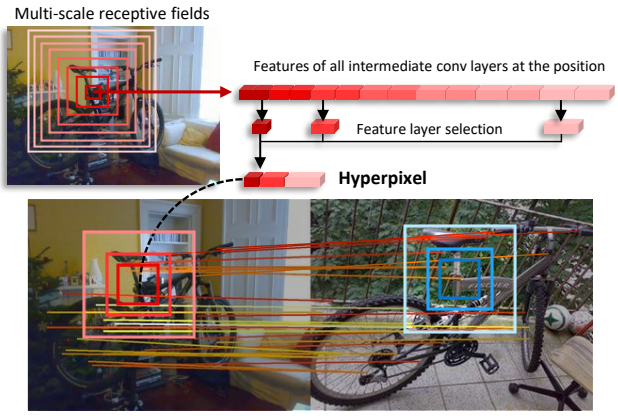

Figure 1: Hyperpixel flow. Top: The *hyperpixel* is a multi-layer pixel representation created with selected levels of features optimized for semantic correspondence. It provides multi-scale features, resolving local ambiguities. Bottom: The proposed method, *hyperpixel flow*, establishes dense correspondences in real time using hyperpixels.

We propose a novel dense matching method, dubbed *hyperpixel flow* (Figure 1). Inspired by the hypercolumns [18] used in object segmentation and detection, we represent images by "hyperpixels" that leverage different levels of features among early to late layers of a convolutional neural network and disambiguate parts of images in multiple visual aspects. The corresponding feature layers for hyperpixels are selected by a simple yet effective search process which requires only a small validation set of supervised image pairs. We show that the resultant hyperpixels provide both fine-grained and context-aware features suited for semantic correspondence and that only a few layers are sufficient and even better for the purpose, thus making hyperpixels an effective representation for light-weight computation. To obtain a geometrically consistent flow of hyperpixels, we present a real-time dense matching algorithm, regularized Hough matching (RHM), building on a recent region matching method using geometric voting [4]. Furthermore,

*The Neural Processing Research Center, Seoul, Korea
†Département d'informatique de l'ENS, ENS, CNRS, PSL University, Paris, France

we also introduce a new large-scale dataset, SPair-71k, with more accurate and richer annotations, which facilitates in-depth analysis for semantic correspondence.

Our paper makes four main contributions:

- We propose *hyperpixels* for establishing reliable dense correspondences between two images, which provide multi-layer features robust to local ambiguities.
- We present an efficient matching algorithm, regularized Hough matching (RHM), that achieves a speed of more than 50 fps on a GPU for 300×200 image pairs.
- We introduce a new dataset, *SPair-71k*, which contains a significantly larger number of image pairs with richer annotations than existing ones.
- The proposed method, *hyperpixel flow*, sets a new state of the art on standard benchmarks as well as SPair-71k.

2. Related Work

Local region matching. Early methods commonly tackle semantic correspondence by matching two sets of local regions based on handcrafted features. Liu *et al*. [32] and Kim *et al*. [22] use dense SIFT descriptors to establish a flow of local regions across similar but different scenes by leveraging a hierarchical optimization technique in a coarse-to-fine manner. Bristow *et al*. [1] use LDA-whitened SIFT descriptors, making correspondence more robust to background clutter. Cho *et al*. [4] introduce an effective voting-based algorithm based on region proposals and HOG features [6] for semantic matching and object discovery. Ham *et al*. [14] further extend the work with a local-offset matching algorithm, and introduce a benchmark dataset with keypoint-level annotations. Taniai *et al*. [46] tackle semantic correspondence jointly with cosegmentation, introducing a benchmark dataset annotated with dense flows and segmentation masks. All these hand-crafted representation fails to capture high-level semantics enough to discriminate complex patterns with large intra-class deformations.

In this context, CNN features have emerged as good alternatives for semantic matching. Long *et al*. [34] show that convolutional features from a CNN pretrained on classification are transferable to correspondence problems. Choy *et al*. [5] attempt to learn a similarity metric based on a CNN using a contrastive loss with hard negative mining. Han *et al*. [16] propose to learn a CNN end-to-end with geometric matching, which uses region proposals as matching primitives. Kim *et al*. [24] introduce a CNN-based self-similarity feature for semantic correspondence, and also use it to estimate dense affine-transformation fields by an iterative discrete-continuous optimization [25]. Novotny *et al*. [37] train a geometry-aware feature in an unsupervised regime and use it for part matching and discovery by measuring confidence scores. Rocco *et al*. [43] propose a neighbourhood consensus network that computes robust matching similarity using 4D convolution filters.

Global image alignment. Some methods have cast semantic correspondence as global alignment. Rocco *et al*. [41] propose a CNN architecture which takes a correlation tensor and directly predicts global transformation parameters for geometric matching. Seo *et al*. [45] improve it using offset-aware correlation kernels with attention. Rocco *et al*. [42] develop a weakly-supervised learning framework using differentiable soft-inlier count loss function. Jeon *et al*. [20] propose a pyramidal affine transformation regression network to compute the correspondence hierarchically from high-level semantics to pixel-level points. Kim *et al*. [23] introduce a recurrent alignment network that performs iterative local transformations with a global constraint.

Multi-layer neural features. Hariharan *et al*. [18] have shown that *hypercolumns* that combine features from multiple layers of CNN, improve object detection, segmentation, and part labeling. Following this work, several methods [26, 30] have used multi-layer neural features with additional modules on object detection task. Fathy *et al*. [10] propose coarse-to-fine stereo matching method that uses multi-layer features in sequence. In semantic correspondence, multi-layer neural features have rarely been explored despite its relevance. Novotny *et al*. [39] use residual hypercolumn features to learn a set of diverse filters for object parts. Ufer and Ommer [47] employ pyramids of pre-trained CNN features to localize salient feature points guided by object proposals, and match them across images using sparse graph matching. In these methods, multi-layer features are mainly used to localize salient parts and the feature layers are manually selected following previous methods [12, 19]. Unlike these approaches and the hypercolumn [18], we use a multi-layer neural feature as a pixel representation for dense matching and optimize feature layers via layer search for the purpose. We show that specific combinations of layers significantly affect matching performance and using only a small number of layers can achieve a remarkable performance.

Neural architecture search (NAS). The layer search for hyperpixels can be viewed as an instance of NAS [33, 51, 53, 54]. Unlike a general search space of network configurations in NAS, however, the search space in our work is limited to combinations of feature layers for visual correspondence.

3. Hyperpixel Flow

Our method presented below, dubbed *hyperpixel flow*, can be divided into three steps: (1) hyperpixel construction, (2) regularized Hough matching, and (3) flow formation. Figure 2 illustrates the overall architecture of our model aligned with the three steps. Each input image is fed

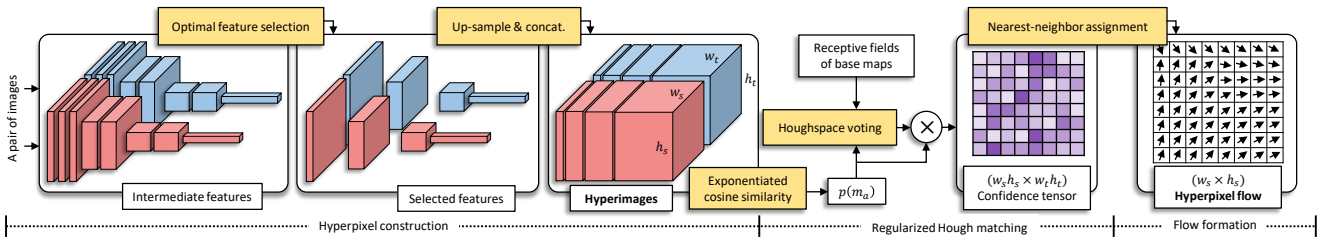

Figure 2: Overall architecture of the proposed method. Hyperpixel flow consists of three main steps: hyperpixel construction, regularized Hough matching, and flow formation. For details, see text.

into a convolutional neural network to create a set of hyperpixels. The hyperpixels are then used as primitives for the regularized Hough matching algorithm to build a tensor of matching confidences for all candidate correspondences. The confidence tensor is transformed into a hyperpixel flow in a post-processing step assigning a match to each hyperpixel. Three steps are detailed in this section.

3.1. Hyperpixel construction

Given an image, a convolutional neural network produces a sequence of L feature maps $(\mathbf{f}^0, \mathbf{f}^1, ..., \mathbf{f}^{L-1})$ as intermediate outputs. We represent the original image by a *hyperimage* by pooling *a small* subset of K feature maps, optimized for semantic correspondence, and concatenating them along channels with upsampling:

$$\mathbf{F} = \left[\mathbf{f}^{l_0}, \zeta(\mathbf{f}^{l_1}), \zeta(\mathbf{f}^{l_2}), ..., \zeta(\mathbf{f}^{l_{K-1}})\right], \tag{1}$$

where ζ denotes a function that upsamples the input feature map to the size of $\mathbf{f}^{l_0}$, the *base* map. We can associate with each spatial position p of the hyperimage the corresponding image coordinates, a hyperpixel feature, and its multi-scale receptive fields. Let us denote by $\mathbf{x}_p$ the image coordinate of position p, and by $\mathbf{f}_p$ the corresponding hyperfeature, *i.e.*, $\mathbf{f}_p = \mathbf{F}(\mathbf{x}_p)$. The hyperpixel at position p on the hyperimage is then defined as

$$\mathbf{h}_p = (\mathbf{x}_p, \mathbf{f}_p). \tag{2}$$

As will be seen in the next subsection, the hyperpixels are used as primitives for the subsequent matching process.

To select the optimal set of feature maps for hyperpixels, we perform a search over all convolutional layers of a given CNN so that a subsequent matching algorithm achieves the best validation performance. In our case, we use regularized Hough matching (Sec. 3.2) for the matching algorithm and the probability of correct keypoints (PCK) (Sec. 5.2) for the performance metric. For the search algorithm, we use a variant of beam search [36], which is a breadth-first search algorithm with a limited memory. Basically, at each iteration, it evaluates the effect of each candidate layer by adding it to current combinations of layers in the memory and then replaces them with a fixed number of top performing combinations. The search process is repeated until the number of selected layers reaches the maximum number of layers allowed. Finally, we use the best combination found along the search. The detailed procedure is summarized in Algorithm 1, where we restrict base layer candidates, $\mathcal{L}_{\text{base}}$ only to layers with a sufficient spatial resolution.

Algorithm 1: Beam search for hyperpixel layers.

Input: $\mathcal{L}_{\text{cand}} = \{0, ..., L-1\}$: all candidate layers
$\mathcal{L}_{\text{base}}$: candidate layers for the base ($\subset \mathcal{L}_{\text{cand}}$)
N_{beam}: the beam size
K_{max}: the maximum number of layers allowed
Output: $\mathcal{L}_{\text{sel}}$: the set of selected layers

function SearchLayers
  // initialize memory buffers
  $\mathcal{M}$.init(); $\mathcal{M}'$.init();
  // base layer search
  forall $l \in \mathcal{L}_{\text{base}}$ **do**
    $v \leftarrow$ evaluateLayerSet($\{l\}$);
    $\mathcal{M}$.insert($(\{l\}, v)$);
  end
  $\mathcal{M}' \leftarrow \mathcal{M}$.findBestN($N_{\text{beam}}$);
  $(\mathcal{L}_{\text{sel}}, v_{\text{sel}}) \leftarrow \mathcal{M}'$.findBest();
  // layer search iterations
  for $k \leftarrow 1$ *to* $K_{\text{max}} - 1$ **do**
    $\mathcal{M}$.init();
    forall $(\mathcal{L}', v') \in \mathcal{M}'$ **do**
      forall $l \in \mathcal{L}_{\text{cand}}$ **do**
        if $l \notin \mathcal{L}' \wedge l > \min(\mathcal{L}')$ **then**
          $v'' \leftarrow$ evaluateLayerSet($\mathcal{L}' \cup \{l\}$);
          $\mathcal{M}$.insert($\mathcal{L}' \cup \{l\}, v''$));
        end
      end
    end
    $\mathcal{M}' \leftarrow \mathcal{M}$.findBestN($N_{\text{beam}}$);
    $(\mathcal{L}^*, v^*) \leftarrow \mathcal{M}'$.findBest();
    if $v^* > v_{\text{sel}}$ **then**
      $(\mathcal{L}_{\text{sel}}, v_{\text{sel}}) \leftarrow (\mathcal{L}^*, v^*)$;
    end
  end
  return $\mathcal{L}_{\text{sel}}$
end

3.2. Regularized Hough matching

In order to establish visual correspondences, we adapt the probabilistic Hough matching (PHM), algorithm of Cho *et al*. [4], to hyperpixels. The key idea of PHM is to re-weight appearance similarity by Hough space voting to enforce geometric consistency. In our context, let $\mathcal{D} = (\mathcal{H}, \mathcal{H}')$ be two sets of hyperpixels, and $m = (\mathbf{h}, \mathbf{h}')$ be a hyperpixel match where $\mathbf{h}$ and $\mathbf{h}'$ are respectively elements of $\mathcal{H}$ and $\mathcal{H}'$. Given a Hough space $\mathcal{X}$ of possible offsets (image transformation) between the two hyperpixels, the confidence for match m, $p(m|\mathcal{D})$, is computed as

$$p(m|\mathcal{D}) \propto p(m_{\mathrm{a}}) \sum_{\mathbf{x} \in \mathcal{X}} p(m_{\mathrm{g}}|\mathbf{x}) \sum_{m \in \mathcal{H} \times \mathcal{H}'} p(m_{\mathrm{a}}) p(m_{\mathrm{g}}|\mathbf{x}), \tag{3}$$

where $p(m_{\mathrm{a}})$ represents the confidence for appearance matching and $p(m_{\mathrm{g}}|\mathbf{x})$ is the confidence for geometric matching with an offset $\mathbf{x}$, measuring how close the offset induced by m is to $\mathbf{x}$. By sharing the Hough space $\mathcal{X}$ for all matches, PHM efficiently computes the match confidence with good empirical performance [4, 14, 16].

In this work we compute appearance matching confidence using hyperpixel features:

$$p(m_{\mathrm{a}}) = \mathrm{ReLU}\Big(\frac{\mathbf{f} \cdot \mathbf{f}'}{\|\mathbf{f}\| \, \|\mathbf{f}'\|}\Big)^d, \tag{4}$$

where the ReLU function clamps negative values to zero and the exponent d is used to emphasize the difference between the hyperpixel features. When combined with Hough voting, this similarity function with $d \geq 2$ improves matching performance by suppressing noisy activations. We set $d = 3$ in our experiments.

To compute $p(m_{\mathrm{g}}|\mathbf{x})$, we construct a two-dimensional offset space, quantize it into a grid of bins, and use a set of center points of the bins for $\mathcal{X}$. For Hough voting, each match m is assigned to the corresponding offset bin to increment the score of the bin by the appearance similarity score, $p(m_{\mathrm{a}})$. Despite their (serial) complexity of $O(|\mathcal{H}| \times |\mathcal{H}'|)$, the operations are mutually independent, and can thus easily be parallelized on a GPU.

Previous versions of PHM all use multi-scale region proposals [35, 40, 48] as matching primitives described with HOG [4, 14] or a single feature map from a CNN [14, 16]. While using irregular and multi-scale region proposals focuses attention on object-like regions, it requires creating a three-dimensional offset space for translation and scale changes with higher memory and computation. In contrast, the use of hyperpixels reduces the Hough space down to two dimensions and makes the voting procedure faster and simpler since all hyperpixels are homogeneous on a predefined regular grid. In addition, unlike region proposals, hyperpixels provide (quasi-)dense image features and their multi-layer features improve performance in practice. In our GPU implementation, our algorithm, *regularized Hough matching* (RHM), runs 100 to 500 times faster than PHM (2∼20 msecs vs. 1∼2 secs), enabling real-time matching.

3.3. Flow formation and keypoint transfer

The raw output of RHM is a tensor of confidences for all candidate matches. It can easily be transformed into a hyperpixel flow in a post-processing step of assigning a match to each hyperpixel, *e.g*., by nearest-neighbor assignment. Since the base map of the hyperimage is selected among early layers, the flow is dense enough for many applications.

Transferring keypoints from an image to the corresponding points in another image is commonly used for evaluating semantic correspondences. We use a simple method for keypoint transfer using hyperpixel flow; given a keypoint $\mathbf{x}_p$ in a source image, its neighbor hyperpixels $\mathcal{N}(\mathbf{x}_p)$ are collected whose base map receptive fields cover the keypoint, and the displacement vectors from the centers of the base map receptive fields to the keypoint, denoted by $\{\mathbf{d}(\mathbf{x}_q)\}_{\mathbf{x}_q \in \mathcal{N}(\mathbf{x}_p)}$, are computed. Given the hyperpixel flow T of $\mathcal{N}(\mathbf{x}_p)$ predicted by our method, we apply the average of the displacements $\{T(\mathbf{x}_q) + \mathbf{d}(\mathbf{x}_q)\}_{\mathbf{x}_q \in \mathcal{N}(\mathbf{x}_p)}$ to localize a corresponding keypoint in the target image.

4. SPair-71k dataset

With growing interest in semantic correspondence, several annotated benchmarks are now available. Some popular ones are summarized in Table 1. Due to the high expense of ground-truth annotations for semantic correspondence, early benchmarks [2, 22] only support indirect evaluation using a surrogate evaluation metric rather than direct matching accuracy. For example, the Caltech-101 dataset in [22] provides binary mask annotations of objects of interest for 1,515 pairs of images and the accuracy of mask transfer is evaluated as a rough approximation to that of matching. Recently, Ham *et al*. [14, 15] and Taniai *et al*. [46] have introduced datasets with ground-truth correspondences. Since then, PF-WILLOW [14] and PF-PASCAL [15] have been used for evaluation in many papers. They contain 900 and 1,300 image pairs, respectively, with keypoint annotations for semantic parts.

All previous datasets, however, have several drawbacks: First, the amount of data is not sufficient to train and test a large model. Second, image pairs do not display much variability in viewpoint, scale, occlusion, and truncation. Third, the annotations are often limited to either keypoints or object segmentation masks, which hinders in-depth analysis. Fourth, the datasets have no clear splits for training, validation, and testing. Due to this, recent evaluations in [16, 42, 43] have been done with different dataset splits of PF-PASCAL. Furthermore, the splits are disjoint in terms of image pairs, but not images: some images are shared between training and testing data.

Dataset name	Size (pairs)	Class	Source datasets	Annotations	Characteristics	Users of the dataset
Caltech-101 [22]	1,515	101	Caltech-101 [11, 29]	object segmentation	tightly cropped images of objects, little background	[14, 16, 20, 24, 28, 41, 42, 45]
PASCAL-PARTS [52]	3,884	20	PASCAL-PARTS [2], PASCAL3D+ [50]	keypoints (0∼12), azimuth, elevation, cyclo-rotation, body part segmentation	tightly cropped images of objects, little background, part and 3D infomation	[5, 16, 24, 25, 39, 47]
Animal-parts [38]	≈7,000	100	ILSVRC 2012 [27]	keypoints (1∼6)	keypoints limited to eyes and feet of animals	[39]
CUB-200-2011 [49]	120k	200	CUB-200-2011 [49]	15 part locations, 312 binary attributes, bbox	tightly cropped images of object, only bird images	[5, 21]
TSS [46]	400	9	FG3DCar [31], JODS [44], PASCAL [17]	object segmentation, flow vectors	cropped images of objects, moderate background	[4, 14, 20, 23, 24, 25, 28, 41, 42, 45]
PF-WILLOW [14]	900	5	PASCAL VOC 2007 [9], Caltech-256 [3, 13]	keypoints (10)	center-aligned images, pairs with the same viewpoint	[14, 16, 23, 24, 25, 39, 41, 45, 47]
PF-PASCAL [15]	1,300	20	PASCAL VOC 2007 [9]	keypoints (4∼17), bbox.	pairs with the same viewpoint	[14, 16, 20, 23, 28, 37, 41, 42, 43, 45]
SPair-71k (ours)	70,958	18	PASCAL3D+ [50], PASCAL VOC 2012 [9]	keypoints (3∼30), azimuth, view-point diff., scale diff., trunc. diff., occl. diff., object seg., bbox.	large-scale data with diverse variations, rich annotations, clear dataset splits	this work

Table 1: Public benchmark datasets for semantic correspondence. The datasets are listed in chronological order. Research papers using the datasets for evaluation are listed in the last column. See text for details.

To resolve these issues, we introduce a new dataset, *SPair-71k*, consisting of total 70,958 pairs of images from PASCAL 3D+ [50] and PASCAL VOC 2012 [9]*. The dataset is significantly larger with rich annotations and clearly organized for learning. In particular, several types of useful annotations are available: keypoints of semantic parts, object segmentation masks, bounding boxes, viewpoint, scale, truncation, and occlusion differences for image pairs, etc. Figure 3 presents the statistics of SPair-71k in pie chart forms and shows a sample image pair with its annotations. For details on our dataset, we refer the readers to the website: http://cvlab.postech.ac.kr/research/SPair-71k/.

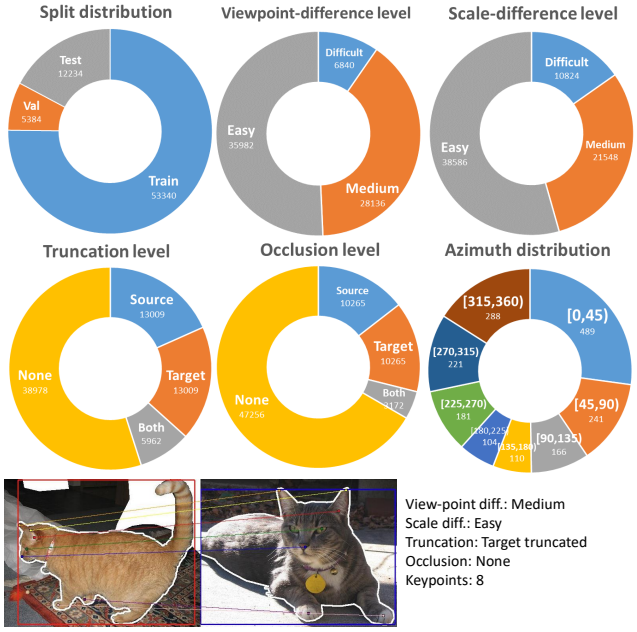

Figure 3: SPair-71k data statistics and an example pair with its annotations. Best viewed in electronic form.

5. Experimental Evaluation

In this section we compare the proposed method with recent state-of-the-art methods and discuss the results.

5.1. Implementation details

We use two CNNs as main backbone networks for hyperpixel features, ResNet-50 and ResNet-101 [19] pre-trained on ImageNet [7]. All convolutional layers of the networks are used as candidate feature layers for hyperpixels. We extract the features at the end of each layer before a ReLU activation. The optimal set of hyperpixel layers, $(l_0, ..., l_{K-1})$, is determined by Algorithm 1 run with a validation split of a target dataset. For this beam search, we set the beam size 4 and the maximum number of layers allowed 8. For the exponent value for hyperpixel similarity, we fix $d = 3$ based on search using PF-PASCAL validation split.

*We do not include 'dining table' and 'sofa' classes because they appear as background in most images and their semantic keypoints are too ambiguous to localize.

5.2. Evaluation metric

For evaluation on PF-WILLOW, PF-PASCAL, and SPair-71k, we use a common evaluation metric of percentage of correct keypoints (PCK), which counts the average number of correctly predicted keypoints given a tolerance threshold. Given predicted keypoint $\mathbf{k}_{\mathrm{pr}}$ and ground-truth keypoint $\mathbf{k}_{\mathrm{gt}}$, the prediction is considered correct if Euclidean distance between them is smaller than a given threshold. The correctness c of each keypoint can be expressed as

$$c = \begin{cases} 1 & \text{if} \quad d(\mathbf{k}_{\mathrm{pr}}, \mathbf{k}_{\mathrm{gt}}) \leq \alpha_\tau \cdot \max(w_\tau, h_\tau) \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

Methods	Supervision	PF-PASCAL (PCK@α_{img})			PF-WILLOW (PCK@α_{bbox})			Caltech-101	
		0.05	0.1	0.15	0.05	0.1	0.15	LT-ACC	IoU
Identity mapping	-	12.7	37.0	60.8	12.2	27.0	41.7	0.77	0.44
PF_{HOG} [14]		31.4	62.5	79.5	28.4	56.8	68.2	0.78	0.50
$\text{CNNGeo}_{\text{res101}}$ [41]	synthetic warp	41.0	69.5	80.4	36.9	69.2	77.8	0.79	0.56
$\text{A2Net}_{\text{res101}}$ [45]	(self-supervised)	42.8	70.8	83.3	36.3	68.8	84.4	0.80	0.57
$\text{DCTM}_{\text{CAT-FCSS}}$ [24]		34.2	69.6	80.2	38.1	61.0	72.1	0.83	0.52
$\text{Weakalign}_{\text{res101}}$ [42]	image labels	49.0	74.8	84.0	37.0	70.2	79.9	0.85	0.63
$\text{NC-Net}_{\text{res101}}$ [43]	(weakly-supervised)	54.3	78.9	86.0	33.8	67.0	83.7	0.85	0.60
$\text{RTNs}_{\text{res101}}$ [23]		55.2	75.9	85.2	41.3	71.9	86.2	-	-
$\text{UCN}_{\text{GoogLeNet}}$ [5]		29.9	55.6	74.0	24.1	54.0	66.5	-	-
$\text{SCNet}_{\text{vgg16}}$ [16]	keypoints	36.2	72.2	82.0	38.6	70.4	85.3	0.79	0.51
$\text{NN-Cyc}_{\text{res101}}$ [28]		55.1	85.7	94.7	40.5	72.5	86.9	0.86	0.62
$\text{HPF}_{\text{res50}}$ (ours)	keypoints	60.5	83.4	92.1	46.5	72.4	84.7	**0.88**	**0.64**
$\text{HPF}_{\text{res101}}$ (ours)	(validation only)	60.1	84.8	92.7	45.9	74.4	85.6	0.87	0.63
$\text{HPF}_{\text{res101-FCN}}$ (ours)		**63.5**	**88.3**	**95.4**	**48.6**	**76.3**	**88.2**	0.87	0.63
$\text{HPF}_{\text{res101}}$ (k=1)	keypoints	$59.4_{\pm 0.89}$	$83.9_{\pm 1.14}$	$92.2_{\pm 0.99}$	$44.5_{\pm 0.90}$	$72.5_{\pm 1.22}$	$84.8_{\pm 0.93}$	0.87	0.63
$\text{HPF}_{\text{res101}}$ (k=2)	(validation only,	$58.3_{\pm 1.33}$	$84.5_{\pm 0.77}$	$92.9_{\pm 0.41}$	$44.7_{\pm 0.92}$	$73.1_{\pm 1.05}$	$85.4_{\pm 0.84}$	0.87	0.63
$\text{HPF}_{\text{res101}}$ (k=3)	small set)	$59.4_{\pm 1.16}$	$84.5_{\pm 0.27}$	$92.7_{\pm 0.35}$	$45.1_{\pm 0.55}$	$73.4_{\pm 0.52}$	$85.4_{\pm 0.48}$	0.87	0.63
$\text{HPF}_{\text{res101}}$ (random)	-	$44.5_{\pm 11.11}$	$74.7_{\pm 6.46}$	$87.3_{\pm 3.13}$	$32.8_{\pm 8.12}$	$62.4_{\pm 6.67}$	$78.2_{\pm 4.20}$	0.85	0.55

Table 2: Results on standard benchmarks of semantic correspondences. Subscripts of the method names indicate backbone networks used. The second column denotes supervisory information used for training or tuning. Numbers in bold indicate the best performance and underlined ones are the second and third best. Results of [14, 16, 24, 41, 42] are borrowed from [23].

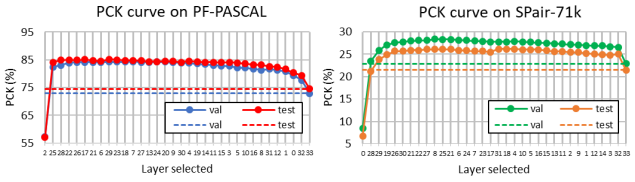

Figure 4: Hyperpixel layer search with ResNet-101 backbone on PF-PASCAL and SPair-71k datasets. Hyperpixel layers are in the order of selection during beam search. Dashed lines indicate PCKs when all layers of a CNN are used for hyperpixels. Best viewed in electronic form.

where w_τ and h_τ are the width and height of either an entire image or object bounding box, $\tau \in \{\text{img}, \text{bbox}\}$, and α_τ is a tolerance factor (in most cases, $\alpha = 0.1$). Note that PCK with α_{bbox} is a more stringent metric than one with α_{img}. The final PCK of a benchmark is evaluated by averaging PCKs of all input image pairs. Following recent papers [16, 41, 42, 43], we evaluate PF-WILLOW with α_{bbox} and PF-PASCAL with α_{img} using the same dataset split as in [43]. For SPair-71k, we use α_{bbox}, which is more stringent.

5.3. Results and analysis

Hyperpixel layers. For PF-PASCAL, the hyperpixel layer results are $(2, 7, 11, 12, 13)$ with ResNet-50 and $(2, 17, 21, 22, 25, 26, 28)$ with ResNet-101. For SPair-71k, the results are $(0, 9, 10, 11, 12, 13)$ with ResNet-50 and $(0, 8, 20, 21, 26, 28, 29, 30)$ with ResNet-101. In order to analyze the effect of each intermediate feature $(\mathbf{f}^{l_0}, ..., \mathbf{f}^{l_{K-1}})$ on hyperpixel, we have measured PCK of our model on both PF-PASCAL and SPair-71k in the order of the layer selection during beam search as shown in Figure 4. The dashed lines represent PCKs using all layers. Interestingly, in both cases, adding the second layer significantly boosts the performance of PCK, and only a few more layers are sufficient to achieve a comparable performance with the best one. After reaching an optimized set of layers, adding more damages the performance. This result demonstrates the effectiveness of hyperpixels compared to conventional hypercolumn features. The result also implies that features resolving local-ambiguity lie in between particular layers, *e.g.*, between layer 20 and 30 in our case.

Benchmark comparisons. Table 2 summarizes comparison to recent methods on three standard benchmarks: PF-PASCAL, PF-WILLOW, and Caltech-101. In this experiment, the hyperpixels tuned using the validation split of PF-PASCAL are evaluated on the test split of PF-PASCAL, and futher evaluated on PF-WILLOW and Caltech-101 for checking transferability as done in [16, 20, 23, 41, 42, 45]. The results clearly show that the proposed method sets new state-of-the-art results on all the three benchmarks, proving the effectiveness of our approach. Note that all recent neural methods for semantic correspondence rely on ImageNet-pretrained features, and thus their performance depends on the backbone networks (indicated by subscripts). As expected, our method using the stronger backbone of ResNet-101 improves the performance compared to using ResNet-50. Furthermore, using the backbone of FCN [30] pretrained with PASCAL VOC 2012 [9], that is a superset of our target dataset [14], significantly boosts performance. This shows that our method is flexible in using backbone networks and can further improve by adopting a better one.

Methods		aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	dog	horse	moto	person	plant	sheep	train	tv	all
Transferred models	CNNGeo$_{res101}$ [41]	21.3	15.1	34.6	12.8	31.2	26.3	24.0	30.6	11.6	24.3	20.4	12.2	19.7	15.6	14.3	9.6	28.5	28.8	18.1
	A2Net$_{res101}$ [45]	20.8	17.1	37.4	13.9	33.6	29.4	26.5	34.9	12.0	26.5	22.5	13.3	21.3	20.0	16.9	11.5	28.9	31.6	20.1
	WeakAlign$_{res101}$ [42]	23.4	17.0	41.6	14.6	37.6	28.1	26.6	32.6	12.6	27.9	23.0	13.6	21.3	22.2	17.9	10.9	31.5	34.8	21.1
	NC-Net$_{res101}$ [43]	24.0	16.0	45.0	13.7	35.7	25.9	19.0	50.4	14.3	32.6	27.4	19.2	21.7	20.3	20.4	13.6	**33.6**	**40.4**	26.4
SPair-71k trained models	CNNGeo$_{res101}$ [41]	23.4	16.7	40.2	14.3	36.4	27.7	26.0	32.7	12.7	27.4	22.8	13.7	20.9	21.0	17.5	10.2	30.8	34.1	20.6
	A2Net$_{res101}$ [45]	22.6	18.5	42.0	**16.4**	37.9	**30.8**	26.5	35.6	13.3	29.6	24.3	16.0	21.6	22.8	20.5	13.5	31.4	36.5	22.3
	WeakAlign$_{res101}$ [42]	22.2	17.6	41.9	15.1	**38.1**	27.4	**27.2**	31.8	12.8	26.8	22.6	14.2	20.0	22.2	17.9	10.4	32.2	35.1	20.9
	NC-Net$_{res101}$ [43]	17.9	12.2	32.1	11.7	29.0	19.9	16.1	39.2	9.9	23.9	18.8	15.7	17.4	15.9	14.8	9.6	24.2	31.1	20.1
HPF$_{res50}$ (ours)		**25.3**	18.5	47.6	14.6	37.0	22.9	18.3	51.1	16.7	31.5	30.8	19.1	23.7	23.8	**23.5**	14.4	30.8	37.2	27.2
HPF$_{res101}$ (ours)		25.2	**18.9**	**52.1**	15.7	38.0	22.8	19.1	**52.9**	**17.9**	**33.0**	**32.8**	**20.6**	**24.4**	**27.9**	21.1	**15.9**	31.5	35.6	**28.2**

Table 3: Per-class PCK ($\alpha_{\text{bbox}} = 0.1$) results on SPair-71k dataset. For transferred model, the original models trained on PASCAL-VOC [41, 45] and PF-PASCAL [42, 43], which are provided by the authors, are used for evaluation. Note that, for SPair-71k trained models, the transferred models are further finetuned on SPair-71k dataset by ourselves with our best efforts. Numbers in bold indicate the best performance and underlined ones are the second and third best.

Approach	Model	PCK	Time (*ms*)
Image alignment	CNNGeo$_{res101}$ [41]	69.5	40
	WeakAlign$_{res101}$ [42]	74.8	41
	A2Net$_{res101}$ [45]	70.8	53
	RTNs$_{res101}$ [23]	75.9	376
Local region matching	SCNet$_{vgg16}$ [16]	72.2	>1000
	PF$_{HOG}$ [14]	62.5	>1000
	NC-Net$_{res101}$ [43]	78.9	261
	HPF$_{res101}$ w/ all layers	74.5	324
	HPF$_{res50}$ w/ all layers	70.1	130
	HPF$_{res101}$	**84.8**	63
	HPF$_{res50}$	83.4	34
	HPF$_{res50}$*	81.1	**19**

Table 4: Inference time comparison on PF-PASCAL benchmark. Hyperpixel layers of HPF$_{res50}$* are (4,7,11,12,13).

Degree of supervision. Different methods in our comparison require different degrees of supervision in training as indicated in the second column of Table 2. The only supervised part of our method is layer selection using a validation set, which can be very small as revealed by small-set experiments, and does not require additional learning: Instead of using all the 308 pairs of the original validation split of PF-PASCAL, the layer search algorithm is performed on k random pairs per class, for a total of $20k$ validation pairs. The average performances over 10 trials are shown along with their standard deviations in the set of rows with $k = 1, 2, 3$ at the bottom of Table 2. Using as little as one sample per class (20 image pairs total) as supervisory signal gives results comparable as using all 308 pairs, outperforming the previous state of the art. Given the cost of data collection and the total amount of user-provided information in weakly-supervised methods, we thus believe that our algorithm with small k values (e.g., $k = 1$) is more cost effective and practical.

Effect of layer search. To check the effect of layer search, we take random combinations of 8 layers (the same number chosen by our layer search) as a baseline. The average results over 10 trials are shown with their standard deviations in the last row of Table 2. Their much worse performance shows that our layer search is crucial.

Matching module	PF-PASCAL $\alpha_{\text{img}} = 0.1$	PF-WILLOW $\alpha_{\text{bbox}} = 0.1$
NN w/ $(d = 1)$	69.0	60.9
RHM w/ $(d = 1)$	81.4	68.6
RHM w/ $(d = 2)$	84.4	73.3
RHM w/ $(d = 3)$*	**84.8**	**74.4**
RHM w/ $(d = 4)$	84.8	74.1
RHM w/ $(d = 5)$	84.5	73.9

Table 5: Ablation studies on RHM with ResNet-101.

Comparison to proposal flow approach [14]. The core differences between hyperpixel flow and proposal flow [14] are the changes in (1) matching primitives, from per-proposal geometric descriptor to hyperpixels, in order to handle problems of local-ambiguity and (2) matching algorithms, from PHM to RHM, in order to leverage hyperpixel geometry for efficiency. In Table 2, significant performance improvements on three different benchmarks demonstrate that our features encoding high-level semantics while being agnostic to instance-specific details are crucial to establish robust correspondences. In addition, as shown in Table 4, the proposed voting method, RHM, with hyperpixels shows an impressive improvement in speed compared to [14].

Inference time comparison. With RHM, predicting dense correspondences for a single pair of images turns out to be much faster compared to other recent models. Table 4 demonstrates the comparison of per-pair inference time on PF-PASCAL. While having more than 5% improvements over current state-of-the-art approach [43], the proposed model runs 4 to 13 times faster. With a slight trade-off on performance, hyperpixels with fewer layers and larger receptive field sizes enables real-time matching.

Ablation studies on matching. To analyze the effects of RHM and its exponent factor d in similarity $p(m_a)$, we experiment with replacing RHM with naïve nearest neighbor matching (NN) and also varying exponent d of similarity. As shown in Table 5, the significant PCK gap between NN and RHM demonstrates the effectiveness of geometry matching. The performance improvement with $d \geq 2$ shows its effect of suppressing noisy votes in RHM.

(a) Source image (b) Target image (c) HPF (ours) (d) CNNGeo [41] (e) A2Net [45] (f) WeakAlign [42] (g) NC-Net [43]

Figure 5: Qualitative results on SPair-71k. The source images are transformed to target images using correspondences.

Methods		View-point			Scale			Truncation				Occlusion				All
		easy	medi	hard	easy	medi	hard	none	src	tgt	both	none	src	tgt	both	
Identity mapping		7.3	3.7	2.6	7.0	4.3	3.3	6.5	4.8	3.5	5.0	6.1	4.0	5.1	4.6	5.6
Transferred models	CNNGeo$_{res101}$ [41]	25.2	10.7	5.9	22.3	16.1	8.5	21.1	12.7	15.6	13.9	20.0	14.9	14.3	12.4	18.1
	A2Net$_{res101}$ [45]	27.5	12.4	6.9	24.1	18.5	10.3	22.9	15.2	17.6	15.7	22.3	16.5	15.2	14.5	20.1
	WeakAlign$_{res101}$ [42]	29.4	12.2	6.9	25.4	19.4	10.3	24.1	16.0	18.5	15.7	23.4	16.7	16.7	14.8	21.1
	NC-Net$_{res101}$ [43]	34.0	18.6	12.8	31.7	23.8	14.2	29.1	22.9	23.4	21.0	29.0	21.1	21.8	19.6	26.4
SPair-71k trained models	CNNGeo$_{res101}$ [41]	28.8	12.0	6.4	24.8	18.7	10.6	23.7	15.5	17.9	15.3	22.9	16.1	16.4	14.4	20.6
	A2Net$_{res101}$ [45]	30.9	13.3	7.4	26.1	21.1	12.4	25.0	17.4	20.5	17.6	24.6	18.6	17.2	16.4	22.3
	WeakAlign$_{res101}$ [42]	29.3	11.9	7.0	25.1	19.1	11.0	24.0	15.8	18.4	15.6	23.3	16.1	16.4	15.7	20.9
	NC-Net$_{res101}$ [43]	26.1	13.5	10.1	24.7	17.5	9.9	22.2	17.1	17.5	16.8	22.0	16.3	16.3	15.2	20.1
HPF$_{res50}$ (ours)		35.0	18.9	13.6	32.0	25.1	15.4	29.7	24.5	23.5	22.9	29.6	22.9	22.1	21.3	27.2
HPF$_{res101}$ (ours)		**35.6**	**20.3**	**15.5**	**33.0**	**26.1**	**15.8**	**31.0**	**24.6**	**24.0**	**23.7**	**30.8**	**23.5**	**22.8**	**21.8**	**28.2**

Table 6: PCK analysis on SPair-71k. Difficulty levels of view points and scales are labeled easy, medium, and hard, while those of truncation and occlusion are indicated by none, source, target, and both.

Model analyses on SPair-71k benchmark. We evaluate several recent methods [41, 42, 43, 45] on our new benchmark dataset. In this experiment, our method tuned using the validation split of SPair-71k is evaluated on the test split of SPair-71k. For each method in comparison, we run two versions of each model: a trained model provided by the authors and the other further finetuned by ourselves on SPair-71k training set. The results are shown in Table 3. We fail to successfully train the method of [42, 43] on SPair-71k so that their performances drop when trained. We guess that their original learning objectives for weakly-supervised learning is fragile in presence of large view-point differences as in SPair-71k. We leave this issue for further investigation and will update the results at our benchmark page.

SPair-71k has several annotation types such as view-point, scale, truncation and occlusion differences. In-depth analyses of each model using these annotations are summarized in Table 6. All models perform better with pairs of small differences, and view-point and scale differences significantly affect the performances. Yet, our method shows more robust results in terms of those variations compared to the others. Figure 5 shows some examples where our method finds reliable correspondences even under a large view-point and scale difference.

6. Conclusion

We have proposed a fast yet effective semantic matching method, hyperpixel flow, which leverages an optimized set of convolutional layer features pre-trained on a classification task. The impressive performance of the proposed method, which is only tuned with a small vadidation split without any end-to-end training, indicates that using relevant levels of multiple neural features is crucial in semantic correspondence. We believe further research in this direction is needed together with feature learning. To this end, we have also introduced a large-scale dataset, SPair-71k, with richer annotations for in-depth analyses, which is intended to resolve drawbacks of existing semantic correspondence datasets and to serve for supervised end-to-end learning of semantic correspondence.

Acknowledgements. This work is supported by Samsung Advanced Institute of Technology (SAIT) and Basic Science Research Program (NRF-2017R1E1A1A01077999), and also in part by the Inria/NYU collaboration and the Louis Vuitton/ENS chair on artificial intelligence.

References

[1] Hilton Bristow, Jack Valmadre, and Simon Lucey. Dense semantic correspondence where every pixel is a classifier. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[2] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 4, 5

[3] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2013. 5

[4] Minsu Cho, Suha Kwak, Cordelia Schmid, and Jean Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2, 4, 5

[5] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *Proc. Neural Information Processing Systems (NeurIPS)*, pages 2414–2422, 2016. 1, 2, 5, 6

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005. 2

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009. 5

[8] Gianluca Donato and Serge Belongie. Approximate thin plate spline mappings. In *Proc. European Conference on Computer Vision (ECCV)*, 2002. 1

[9] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, 111(1):98–136, Jan 2015. 5, 6

[10] Mohammed E Fathy, Quoc-Huy Tran, M Zeeshan Zia, Paul Vernaza, and Manmohan Chandraker. Hierarchical metric learning and matching for 2d and 3d geometric correspondences. In *Proc. European Conference on Computer Vision (ECCV)*, pages 803–819, 2018. 2

[11] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, page 178, 2004. 5

[12] Ross Girshick, Forrest Iandola, Trevor Darrell, and Jitendra Malik. Deformable part models are convolutional neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 437–446, 2015. 2

[13] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. *CalTech Report*, 2007. 5

[14] Bumsub Ham, Minsu Cho, Cordelia Schmid, and Jean Ponce. Proposal flow. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3475–3484, 2016. 2, 4, 5, 6, 7

[15] Bumsub Ham, Minsu Cho, Cordelia Schmid, and Jean Ponce. Proposal flow: Semantic correspondences from object proposals. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(7):1711–1725, 2018. 4, 5

[16] Kai Han, Rafael S Rezende, Bumsub Ham, Kwan-Yee K Wong, Minsu Cho, Cordelia Schmid, and Jean Ponce. Scnet: Learning semantic correspondence. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2, 4, 5, 6, 7

[17] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2011. 5

[18] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 447–456, 2015. 1, 2

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2, 5

[20] Sangryul Jeon, Seungryong Kim, Dongbo Min, and Kwanghoon Sohn. Parn: Pyramidal affine regression networks for dense semantic correspondence. In *Proc. European Conference on Computer Vision (ECCV)*, pages 351–366, 2018. 2, 5, 6

[21] Angjoo Kanazawa, David W Jacobs, and Manmohan Chandraker. Warpnet: Weakly supervised matching for single-view reconstruction. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3253–3261, 2016. 5

[22] Jaechul Kim, Ce Liu, Fei Sha, and Kristen Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2307–2314, 2013. 2, 4, 5

[23] Seungryong Kim, Stephen Lin, Sangryul Jeon, Dongbo Min, and Kwanghoon Sohn. Recurrent transformer networks for semantic correspondence. In *Proc. Neural Information Processing Systems (NeurIPS)*, 2018. 1, 2, 5, 6, 7

[24] Seungryong Kim, Dongbo Min, Bumsub Ham, Sangryul Jeon, Stephen Lin, and Kwanghoon Sohn. Fcss: Fully convolutional self-similarity for dense semantic correspondence. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6560–6569, 2017. 1, 2, 5, 6

[25] Seungryong Kim, Dongbo Min, Stephen Lin, and Kwanghoon Sohn. Dctm: Discrete-continuous transformation matching for semantic flow. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, volume 6, 2017. 2, 5

[26] Tao Kong, Anbang Yao, Yurong Chen, and Fuchun Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 845–853, 2016. 2

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Neural Information Processing Systems (NeurIPS)*, pages 1097–1105, 2012. 5

[28] Zakaria Laskar, Hamed R Tavakoli, and Juho Kannala. Semantic matching by weakly supervised 2d point set registration. *arXiv preprint arXiv:1901.08341*, 2019. 5, 6

[29] Fei-Fei Li, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(4):594–611, 2006. 5

[30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2117–2125, 2017. 2, 6

[31] Yen-Liang Lin, Vlad I Morariu, Winston Hsu, and Larry S Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *Proc. European Conference on Computer Vision (ECCV)*, pages 466–480, 2014. 5

[32] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. In *Proc. European Conference on Computer Vision (ECCV)*, 2008. 2

[33] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *Proc. International Conference on Learning Representations (ICLR)*, 2019. 2

[34] Jonathan L Long, Ning Zhang, and Trevor Darrell. Do convnets learn correspondence? In *Proc. Neural Information Processing Systems (NeurIPS)*, pages 1601–1609, 2014. 2

[35] Santiago Manen, Matthieu Guillaumin, and Luc Van Gool. Prime object proposals with randomized prim's algorithm. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 2536–2543, 2013. 4

[36] M.F. Medress, F.S. Cooper, J.W. Forgie, C.C. Green, D.H. Klatt, M.H. O'Malley, E.P. Neuburg, A. Newell, D.R. Reddy, B. Ritea, J.E. Shoup-Hummel, D.E. Walker, and W.A. Woods. Speech understanding systems: Report of a steering committee. *Artificial Intelligence*, 9(3):307 – 316, 1977. 3

[37] David Novotny, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Self-supervised learning of geometrically stable features through probabilistic introspection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3637–3645, 2018. 2, 5

[38] David Novotny, Diane Larlus, and Andrea Vedaldi. I have seen enough: Transferring parts across categories. In *Proc. British Machine Vision Conference (BMVC)*, 2016. 5

[39] David Novotny, Diane Larlus, and Andrea Vedaldi. Anchornet: A weakly supervised network to learn geometry-sensitive features for semantic matching. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2867–2876, 2017. 2, 5

[40] Jordi Pont-Tuset, Pablo Arbelaez, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(1):128–140, 2017. 4

[41] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. Convolutional neural network architecture for geometric matching. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 5, 6, 7, 8

[42] Ignacio Rocco, Relja Arandjelovi, and Josef Sivic. End-to-end weakly-supervised semantic alignment. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 4, 5, 6, 7, 8

[43] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood consensus networks. In *Proc. Neural Information Processing Systems (NeurIPS)*, pages 1656–1667, 2018. 2, 4, 5, 6, 7, 8

[44] Michael Rubinstein, Armand Joulin, Johannes Kopf, and Ce Liu. Unsupervised joint object discovery and segmentation in internet images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1939–1946, 2013. 5

[45] Paul Hongsuck Seo, Jongmin Lee, Deunsol Jung, Bohyung Han, and Minsu Cho. Attentive semantic alignment with offset-aware correlation kernels. In *Proc. European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 5, 6, 7, 8

[46] Tatsunori Taniai, Sudipta N Sinha, and Yoichi Sato. Joint recovery of dense correspondence and cosegmentation in two images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4246–4255, 2016. 2, 4, 5

[47] Nikolai Ufer and Björn Ommer. Deep semantic feature matching. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5929–5938, 2017. 2, 5

[48] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International Journal of Computer Vision (IJCV)*, 104(2):154–171, 2013. 4

[49] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 5

[50] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. *Proc. Winter Conference on Applications of Computer Vision (WACV)*, pages 75–82, 2014. 5

[51] Saining Xie, Alexander Kirillov, Ross B. Girshick, and Kaiming He. Exploring randomly wired neural networks for image recognition. *arXiv preprint arXiv:1904.01569*, 2019. 2

[52] Tinghui Zhou, Yong Jae Lee, Stella X Yu, and Alyosha A Efros. FlowWeb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1191–1200, 2015. 5

[53] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *Proc. International Conference on Learning Representations (ICLR)*, 2017. 2

[54] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2