SegSort: Segmentation by Discriminative Sorting of Segments

Jyh-Jing Hwang[1,2] Stella X. Yu[1] Jianbo Shi[2]
Maxwell D. Collins[3] Tien-Ju Yang[4] Xiao Zhang[3] Liang-Chieh Chen[3]
[1]UC Berkeley / ICSI [2]University of Pennsylvania [3]Google Research [4]MIT
{jyh,stellayu}@berkeley.edu {jyh,jshi}@seas.upenn.edu
{maxwellcollins,tjy,andypassion,lcchen}@google.com

Abstract

Almost all existing deep learning approaches for semantic segmentation tackle this task as a pixel-wise classification problem. Yet humans understand a scene not in terms of pixels, but by decomposing it into perceptual groups and structures that are the basic building blocks of recognition. This motivates us to propose an end-to-end pixel-wise metric learning approach that mimics this process. In our approach, the optimal visual representation determines the right segmentation within individual images and associates segments with the same semantic classes across images. The core visual learning problem is therefore to maximize the similarity within segments and minimize the similarity between segments. Given a model trained this way, inference is performed consistently by extracting pixel-wise embeddings and clustering, with the semantic label determined by the majority vote of its nearest neighbors from an annotated set. As a result, we present the SegSort, as a first attempt using deep learning for unsupervised semantic segmentation, achieving 76% performance of its supervised counterpart. When supervision is available, SegSort shows consistent improvements over conventional approaches based on pixel-wise softmax training. Additionally, our approach produces more precise boundaries and consistent region predictions. The proposed SegSort further produces an interpretable result, as each choice of label can be easily understood from the retrieved nearest segments.

1. Introduction

Semantic segmentation is usually approached by extending image-wise classification [41, 38] to pixel-wise classification, deployed in a fully convolutional fashion [47]. In contrast, we study the semantic segmentation task in terms of perceiving an image in groups of pixels and associating objects from a large set of images. Particularly, we take the perceptual organization view [66, 6] that pixels group by visual similarity and objects form by visual familiarity; consequently a representation is developed to best relate pixels and segments to each other in the visual world. Our method, such motivated, not only achieves better supervised semantic segmentation but also presents the first attempt using deep learning for *unsupervised* semantic segmentation.

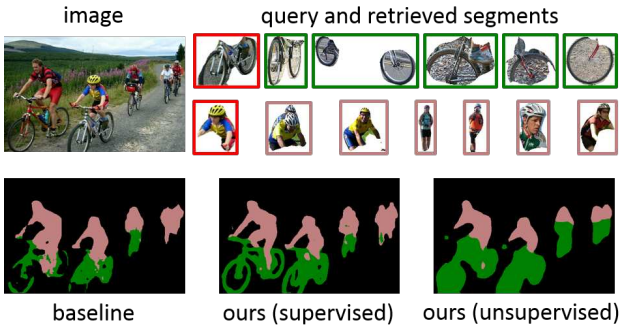

Figure 1. Top: Our proposed approach partitions an image in the embedding space into aligned segments (framed in red) and assign the majority labels from retrieved segments (framed in green or pink). Bottom: Our approach presents the first deep learning based unsupervised semantic segmentation (right). If supervised, our approach produces more consistent region predictions and precise boundaries in the supervised setting (middle) compared to its parametric counterpart (left).

We formulate this intuition as an end-to-end metric learning problem. Each pixel in an image is mapped via a CNN to a point in some visual embedding space, and nearby points in that space indicate pixels belonging to the same segments. From all the segments collected across images, clusters in the embedding space form semantic concepts. In other words, we *sort segments* with respect to their visual and semantic attributes. The optimal visual representation delivers the right segmentation within individual images and associates segments with the same semantic classes across images, yielding a non-parametric model as its complexity scales with number of segments (exemplars).

We derive our method based on maximum likelihood estimation of a single equation, resulting in a two-stage

Expectation-Maximization (EM) framework. The first stage performs a spherical (von Mises-Fisher) K-Means clustering [4] for image segmentation. The second stage adapts the E-step for a pixel-to-segment loss to optimize the metric learning CNN.

As a result, we present the SegSort (Segment Sorting) as a first attempt to apply deep learning for semantic segmentation from the *unsupervised* perspective. Specifically, we create pseudo segmentation masks aligned with visual cues using a contour detector [2, 32, 73] and train the pixel-wise embedding network to separate all the segments. The unsupervised SegSort achieves 76% performance of its supervised counterpart. We further show that various visual groups are automatically discovered in our framework.

When supervision is available (*i.e.*, supervised semantic segmentation), we segment each image with the spherical K-Means clustering and train the network following the same optimization, but incorporated with Neighborhood Components Analysis criterion [22, 71] for semantic labels.

To summarize our major contributions:

1. We present the first end-to-end trained non-parametric approach for supervised semantic segmentation, with performance exceeding its parametric counterparts that are trained with pixel-wise softmax loss.
2. We propose the first unsupervised deep learning approach for semantic segmentation, which achieves 76% performance of its supervised counterpart.
3. Our segmentation results can be easily understood from retrieved nearest segments and readily interpretable.
4. Our approach produces more precise boundaries and more consistent region segmentations compared with parametric pixel-wise prediction approaches.
5. We demonstrate the effectiveness of our method on two challenging datasets, PASCAL VOC 2012 [16] and Cityscapes [14].

2. Related Works

Segmentation and Clustering. Segmentation involves extracting representations from local patches and clustering them based on different criteria, *e.g.*, fitting mixture models [74, 5], mode-finding [13, 4], or graph partitioning [18, 62, 49, 64, 78]. The mode-finding algorithms, *e.g.*, mean shift [13] or K-Means [26, 4], are mostly related. Traditionally, pixels are encoded in a joint spatial-range domain by a single vector with their spatial coordinates and visual features concatenated. Applying mean shift or K-Means filtering can thus converge for each pixel. Spectral graph theory [12], and in particular the Normalized Cut [62] criterion provides a way to further integrate global image information for better segmentation. More recently, superpixel approaches [1] emerge to be a popular pre-processing step that helps reduce the computation, or can be used to refine the semantic segmentation predictions [20]. However, the challenge of perceptual organization is to process information from different levels together to form consensus segmentation. Hence, our proposed approach aims to integrate image segmentation and clustering into end-to-end embedding learning for semantic segmentation.

Semantic Segmentation. Current state-of-the-art semantic segmentation models are based on Fully Convolutional Networks [41, 61, 47], tackling the problem via pixel-wise classification. Given limited local context, it may be ambiguous to correctly classify a single pixel, and thus it is common to resort to multi-scale context information [28, 63, 36, 39, 23, 76, 51, 34, 31]. Typical approaches include image pyramids [17, 55, 15, 43, 10, 8] and encoder-decoder structures [3, 56, 42, 19, 54, 77, 79, 11]. Notably, to better capture multi-scale context, PSPNet [80] performs spatial pyramid pooling [24, 40, 46] at several grid scales, while DeepLab [8, 9, 75] applies the ASPP module (Atrous Spatial Pyramid Pooling) consisting of several parallel atrous convolution [30, 21, 61, 53] with different rates. In this work, we experiment with applying our proposed training algorithm to PSPNet and DeepLabv3+, and show consistent improvements.

Before deep learning takes a leap, non-parametric methods for semantic segmentation are explored. In the unsupervised setting, [57] proposes data-driven boundary and image grouping, formulated with MRF to enhance semantic boundaries; [67] extracts superpixels before nearest neighbor search; [45] performs dense SIFT to find dense deformation fields between images to segment and recognize a query image. With supervision, [50] learns semantic object exemplars for detection and segmentation.

It is worth noting Kong and Fowlkes [37] also integrate vMF mean-shift clustering into the semantic segmentation pipeline. However, the clustering with contrastive loss is used for regularizing features and the whole system still relies on softmax loss to produce the final segmentation.

Our work also bears a similarity to the work Scene Collaging [33], which presents a nonparametric scene grammar for parsing the images into segments for which object labels are retrieved from a large dataset of example images.

Metric Learning. Metric learning approaches [35, 22] have achieved remarkable performance on different vision tasks, such as image retrieval [70, 72, 71] and face recognition [65, 69, 60]. Such tasks usually involve open world recognition, since classes during testing might be disjoint from the ones in the training set. Metric learning minimizes intra-class variations and maximizes inter-class variations with pairwise losses *e.g.*, contrastive loss [7] and triplet loss [29]. Recently, Wu *et al*. [72] propose a non-parametric softmax formulation for training feature embeddings to separate every image for unsupervised image recognition and retrieval.

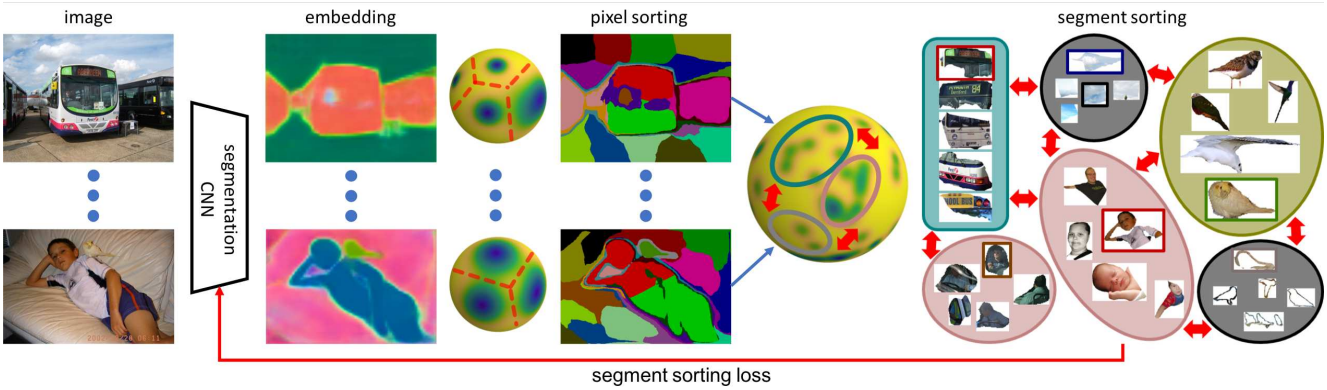

Figure 2. The overall training diagram for our proposed framework, Segment Sorting (SegSort), with the vMF clustering [4]. Given a batch of images (leftmost), we compute pixel-wise embeddings (middle left) from a metric learning segmentation network. Then we segment each image with the vMF clustering (middle right), dubbed pixel sorting. We train the network via the maximum likelihood estimation derived from a mixture of vMF distributions, dubbed segment sorting. In between, we also illustrate how to process pixel-wise features on a hyper-sphere for pixel and segment sorting. A segment (rightmost) is color-framed with its corresponding vMF clustering color if in the displayed images. Unframed segments from different images are associated in the embedding space. The inference is done with the same procedure but using the k-nearest neighbor search to associate segments in the training set.

The non-parametric softmax is further incorporated with Neighborhood Components Analysis [22] to improve generalization for supervised image recognition [71]. An important technical point on metric learning is normalization [68, 60] so that features lie on a hypersphere, which is why the vMF distribution is of particular interest.

3. Method

Our end-to-end learning framework consists of three sequential components: 1) A CNN, *e.g.*, DeepLab [11], FCN [47], or PSPNet [80], that generates pixel-wise embeddings from an image. 2) A clustering method that partitions the pixel-wise embeddings into a fine segmentation, dubbed pixel sorting. 3) A metric learning formulation for separating and grouping the segments into semantic clusters, dubbed segment sorting.

We start with an assumption that the pixel-wise normalized embeddings from the CNN within a segment follow a von Mises-Fisher (vMF) distribution. We thus formulate the pixel sorting with spherical K-Means clustering and the segment sorting with corresponding maximum likelihood estimation. During inference, the segment sorting is replaced with k-nearest neighbor search. We then apply to each query segment the majority label of retrieved segments.

We now give a high level mathematical explanation of the entire optimization process. Let $\mathcal{V} = \{\boldsymbol{v}_i\} = \{\phi(x_i)\}$ be the set of pixel embeddings where $\boldsymbol{v}_i$ is produced by a CNN ϕ centered at pixel x_i. Let $\mathcal{Z} = \{z_i\}$ be the image segmentation with k segments, or $z_i = s$ indicates if a pixel i belongs to a segment s. Let $\Theta = \{\theta_{z_i}\}$ be the set of parameters that capture the representative feature of a segment through a predefined distribution f (mixture of vMF here). Our main optimization objective can be concluded as:

$$\min_{\phi,\mathcal{Z},\Theta} -\log P(\mathcal{V}, \mathcal{Z} \mid \Theta) = \min_{\phi,\mathcal{Z},\Theta} -\sum_i \log \frac{1}{k} f_{z_i}(\boldsymbol{v}_i \mid \theta_{z_i}). \tag{1}$$

In pixel sorting, we use a standard EM framework to find the optimal $\mathcal{Z}$ and Θ, with ϕ fixed. In segment sorting, we adapt the previous E step for loss calculation through a set of images to optimize ϕ, with $\mathcal{Z}$ and Θ fixed. Performing pixel sorting and segment sorting can thus be viewed as a two-stage EM framework.

This section is organized as follows. We first describe the pixel sorting in Sec. 3.1, which includes a brief review of spherical K-Means clustering and creation of aligned segments. We then derive two forms of the segment sorting loss for segment sorting in Sec. 3.2. Finally, we describe the inference procedure in Sec. 3.3. The overall training diagram is illustrated in Fig. 2 and the summarized algorithm can be found in the supplementary.

3.1. Pixel Sorting

We briefly review the vMF distribution and its corresponding spherical K-Means clustering algorithm [4], which is used to segment an image as pixel sorting.

We assume the pixel-wise d-dimensional embeddings $\boldsymbol{v} \in \mathbb{S}^{d-1}$ (CNN's last layer features after normalization) within a segment follow a vMF distribution. vMF distributions are of particular interest as it is one of the simplest distributions with properties analogous to those of the multivariate Gaussian for directional data. Its probability density function is given by

$$f(\boldsymbol{v} \mid \boldsymbol{\mu}, \kappa) = C_d(\kappa) \exp(\kappa \boldsymbol{\mu}^\top \boldsymbol{v}), \tag{2}$$

where $C_d(\kappa) = \frac{\kappa^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\kappa)}$ is the normalizing constant where $I_r(\cdot)$ represents the modified Bessel function of the first kind and order r. $\boldsymbol{\mu} = \sum_i \boldsymbol{v}_i / ||\sum_i \boldsymbol{v}_i||$ is the mean direction and $\kappa \geq 0$ is the concentration parameter. Larger κ indicates stronger concentration about μ. In our particular case, we assume a constant κ for all vMF distributions to circumvent the expensive calculation of $C_d(\kappa)$.

The embeddings of an image with k segments can thus be considered as a mixture of k vMF distributions with a uniform prior, or

$$f(\boldsymbol{v} \mid \Theta) = \sum_{s=1}^{k} \frac{1}{k} f_s(\boldsymbol{v} \mid \boldsymbol{\mu}_s, \kappa), \tag{3}$$

where $\Theta = \{\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_k, \kappa\}$. Let z_i be the hidden variable that indicates a pixel embedding $\boldsymbol{v}_i$ belongs to a particular segment s, or $z_i = s$. Let $\mathcal{V} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k\}$ be the set of pixel embeddings and $\mathcal{Z} = \{z_1, \cdots, z_k\}$ be the set of corresponding hidden variables. The log-likelihood of the observed data is thus given by

$$\log P(\mathcal{V}, \mathcal{Z} \mid \Theta) = \sum_i \log \frac{1}{k} f_{z_i}(\boldsymbol{v}_i \mid \boldsymbol{\mu}_{z_i}, \kappa). \tag{4}$$

Since $\mathcal{Z}$ is unknown, the EM framework is used to estimate this otherwise intractable maximum likelihood, resulting in the spherical K-Means algorithm with an assumption of $\kappa \mapsto \infty$. This assumption holds if all the embeddings within a region are the same (homogeneous), which will be our training objective described in Sec. 3.2.

The E-step that maximizes the likelihood of Eqn. 4 is to assign $z_i = s$ with a posterior probability [52]:

$$p(z_i = s | \boldsymbol{v}_i, \Theta) = \frac{f_s(\boldsymbol{v}_i \mid \Theta)}{\sum_{l=1}^{k} f_l(\boldsymbol{v}_l | \Theta)}. \tag{5}$$

In the setting of K-Means, we use hard assignments to update z_i, or $z_i = \text{argmax}_s\, p(z_i = s \mid \boldsymbol{v}_i, \Theta) = \text{argmax}_s\, \boldsymbol{\mu}_s^\top \boldsymbol{v}_i$. We further denote the set of pixels within a segment c as $\mathcal{R}_c$; hence $p(z_i = c \mid \boldsymbol{v}_i, \Theta) = 1$ if $i \in \mathcal{R}_c$ or 0 otherwise after hard assignments.

The M-step that maximizes the expectation of Eqn. 4 can be derived [4] as

$$\hat{\boldsymbol{\mu}}_c = \frac{\sum_i \boldsymbol{v}_i p(z_i = c \mid \boldsymbol{v}_i, \Theta)}{||\sum_i \boldsymbol{v}_i p(z_i = c \mid \boldsymbol{v}_i, \Theta)||} = \frac{\sum_{i \in \mathcal{R}_c} \boldsymbol{v}_i}{||\sum_{i \in \mathcal{R}_c} \boldsymbol{v}_i||}, \tag{6}$$

which is the mean direction of pixel embeddings within segment c. The spherical K-Means clustering is thus done through alternating updates of $\mathcal{Z}$ (E-step) and Θ (M-step).

One problem of K-Means clustering is the dynamic number of EM steps, which would cause uncertain memory consumption during training. However, we find in practice a small fixed number of EM steps, *i.e.*, 10 iterations, can already produce good segmentations.

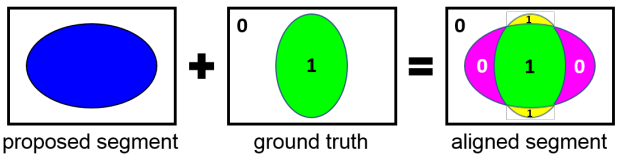

Figure 3. During supervised training, we partition the proposed segments (left) given the ground truth mask (middle). The yielded segments (right) are thus aligned with ground truth mask. Each aligned segment is labeled (0 or 1) according to ground truth mask. Note that the purple and yellow segments become, respectively, false positive and false negative that help regularize predicted boundaries.

If we only use embedding features for K-Means clustering, each resulted cluster is often disconnected and scattered. As our goal is to spatially segment an image, we concatenate pixel coordinates with the embeddings so that the K-Means clustering is guided by spatiality.

Creating Aligned Segments. Segments that are aligned with different visual cues are critical for producing coherent boundaries. However, segments produced by K-Means clustering do not always conform to the ground truth boundaries. If one segment contains different semantic labels, it clearly contradicts our assumption of homogeneous embeddings within a segment. Therefore, we partition a segment given the ground truth mask in the supervised setting so that each segment contains exactly a single semantic label as illustrated in Fig. 3.

It is easy to see that the segments after partition are always aligned with semantic boundaries. Furthermore, this partition creates small segments of false positives and false negatives which can naturally serve as hard negative examples during loss calculation.

3.2. Segment Sorting

Following our assumption of homogeneous embeddings per segment, the training is therefore to enforce this criterion, which is done by optimizing the CNN parameters for better feature extraction.

We first define a *prototype* as the most representative embedding feature of a segment. Since the embeddings in a segment follow a vMF distribution, the mean direction vector $\boldsymbol{\mu}_c$ in Eqn. 6 can naturally be used as the prototype.

In Sec. 3.1, we consider the posterior probability of a pixel embedding $\boldsymbol{v}_i$ belonging to a segment s with fixed CNN parameters in Eqn. 5. Now we revisit it with free CNN parameters ϕ and a constant hyperparameter κ:

$$p_\phi(z_i = s \mid \boldsymbol{v}_i, \Theta) = \frac{f_s(\boldsymbol{v}_i | \Theta)}{\sum_{l=1}^{k} f_l(\boldsymbol{v}_l | \Theta)} = \frac{\exp(\kappa \boldsymbol{\mu}_s^\top \boldsymbol{v}_i)}{\sum_{l=1}^{k} \exp(\kappa \boldsymbol{\mu}_l^\top \boldsymbol{v}_i)}. \tag{7}$$

As both the embedding $\boldsymbol{v}$ and prototype $\boldsymbol{\mu}$ are of unit length,

the dot product $\boldsymbol{v}^\top \boldsymbol{\mu} = \frac{\boldsymbol{v}^\top \boldsymbol{\mu}}{||\boldsymbol{v}|| \, ||\boldsymbol{\mu}||}$ becomes the cosine similarity. The numerator indicates the exponential cosine similarity between a pixel embedding $\boldsymbol{v}_i$ and a particular segment prototype $\boldsymbol{\mu}_s$. The denominator includes the exponential cosine similarities w.r.t. all the segment prototypes. The value of p_ϕ indicates the ratio of pixel embedding $\boldsymbol{v}_i$ close to segment s compared to all the other segments.

The training objective is thus to maximize the posterior probability of a pixel embedding belonging to its corresponding segment c obtained from the K-Means clustering. In other words, we want to minimize the following negative log-likelihood, or the vMF loss:

$$L^i_{\text{vMF}} = -\log p_\phi(c \mid \boldsymbol{v}_i, \Theta) = -\log \frac{\exp(\kappa \boldsymbol{\mu}_c^\top \boldsymbol{v}_i)}{\sum_{l=1}^{k} \exp(\kappa \boldsymbol{\mu}_l^\top \boldsymbol{v}_i)}. \tag{8}$$

The total loss is the average over all pixels. As a result, minimizing L_{vMF} has two effects: One is expressed by the numerator, where it encourages each pixel embedding to be close to its own segment prototype. The other is from the denominator, where it encourages each embedding feature to be far away from all other segment prototypes.

Note that this vMF loss does not require any ground truth semantic labels. We can therefore use this loss to train the CNN in an unsupervised setting. As the loss pushes every segment as far away as possible, visually similar segments are forced to stay closer on the hypersphere.

To make use of ground truth semantic information, we consider soft neighborhood assignments in the Neighborhood Components Analysis [22]. The idea of soft neighborhood assignments is to encourage the probability of one example selecting its neighbors (excluding itself) of the same category. In our case, we want to encourage the probability of a pixel embedding $\boldsymbol{v}_i$ selecting any other segment in the same category, denoted as c^+, as its neighbors. We can define such probability as follows, adapted from Eqn. 7:

$$p'_\phi(z_i = c^+ \mid \boldsymbol{v}_i, \Theta) = \frac{f_{c^+}(\boldsymbol{v}_i|\Theta)}{\sum_{l \neq c} f_l(\boldsymbol{v}_l|\Theta)} = \frac{\exp(\kappa \boldsymbol{\mu}_{c^+}^\top \boldsymbol{v}_i)}{\sum_{l \neq c} \exp(\kappa \boldsymbol{\mu}_l^\top \boldsymbol{v}_i)},$$
$$p'_\phi(z_i = c \mid \boldsymbol{v}_i, \Theta) = 0. \tag{9}$$

We denote the set of segments $\{c^+\}$ w.r.t. pixel i as C_i^+.

Our final loss function is therefore the negative log total probability of pixel i selecting a neighbor prototype in the same category:

$$\begin{aligned} L^i_{\text{vMF-N}} &= -\log \sum_{s \in C_i^+} p'_\phi(z_i = s \mid \boldsymbol{v}_i, \Theta) \\ &= -\log \frac{\sum_{s \in C_i^+} \exp(\kappa \boldsymbol{\mu}_s^\top \boldsymbol{v}_i)}{\sum_{l \neq c} \exp(\kappa \boldsymbol{\mu}_l^\top \boldsymbol{v}_i)}. \end{aligned} \tag{10}$$

The total loss is the average over all pixels. Minimizing this loss is to maximize the expected number of pixels correctly classified by associating the right neighbor prototypes. The ground truth labels are thus used for finding the set of same-class segments C_i^+ w.r.t. pixel i within a mini-batch (and memory banks). If there is no other segment in the same category, we fall back to the previous vMF loss. Since both vMF and vMF-N losses serve the same purpose for grouping and separating segments by optimizing the CNN feature extraction, we dub them segment sorting losses.

Understandably, an essential component of the segment sorting loss is the existence of semantic neighbor segments (in the numerator) and the abundance of alien segments (in the denominator). That is, the more examples presented at once, the better the optimization. We thus leverage two strategies: 1) We calculate the loss w.r.t. all the segments in the batch as opposed to traditionally image-wise loss function. 2) We use additional memory banks that cache the segment prototypes from previous batches. In our experiments, we cache up to 2 batches. These two strategies help the fragmented segments (produced by segment partition in Fig. 3) connect to other similar segments between different images, or even between different batches.

3.3. Inference via K-Nearest Neighbor Retrieval

After training, we calculate and save all the segment prototypes in the training set. We calculate the prototypes using pixels with majority labels within the segments, ignoring other unresolved noisy pixels.

During inference, we again conduct the K-Means clustering and then perform k-nearest neighbor search for each segment to retrieve the labels from segments in the training set. The ablation study on inference runtime and memory can be found in the supplementary.

Our overall framework is non-parametric. We use vMF clustering to organize pixel embeddings into segment exemplars, whose number is proportional to number of images in the training set. The embeddings of exemplars are trained with a nearest neighbor criterion such that the inference can be done consistently, resulting in a non-parametric model.

Base / Backbone / Method	mIoU	f-measure
DeepLabv3+ / MNV2 / Softmax	72.51	50.90
DeepLabv3+ / MNV2 / SegSort	74.94	58.83
PSPNet / ResNet-101 / Softmax	80.12	59.64
PSPNet / ResNet-101 / ASM [31]	81.43	62.35
PSPNet / ResNet-101 / SegSort	81.77	63.71
DeepLabv3+ / MNV2 / Softmax	73.25	-
DeepLabv3+ / MNV2 / SegSort	74.88	-
PSPNet / ResNet-101 / Softmax	80.63	-
PSPNet / ResNet-101 / SegSort	82.41	-

Table 1. Quantitative results on Pascal VOC 2012. The first 4 rows with gray colored background are on validation set while the last 4 rows are on testing set. Networks trained with SegSort consistently outperform their parametric counterpart (Softmax) by 1.63 to 2.43% in mIoU and by 4.07 to 7.97% in boundary f-measure.

4. Experiments

In this section, we demonstrate the efficacy of our Segment Sorting (SegSort) through experiments and visual analyses. We first describe the experimental setup in Section 4.1. Then we summarize all the quantitative and qualitative results of fully supervised semantic segmentation in Section 4.2. Lastly, we present results of the proposed approach for unsupervised semantic segmentation in Section 4.3. Additional experiments including ablation studies, t-SNE embedding visualization, and qualitative results on Cityscapes can be found in the supplementary.

4.1. Experimental Setup

Datasets. We mainly use two datasets in the experiments, i.e., PASCAL VOC 2012 [16] and Cityscapes [14].

PASCAL VOC 2012 [16] segmentation dataset contains 20 object categories and one background class. The original dataset contains $1,464$ (*train*) / $1,449$ (*val*) / $1,456$ (*test*) images. Following the procedure of [47, 8, 80], we augment the training data with the annotations of [25], resulting in $10,582$ (*train_aug*) images.

Cityscapes [14] is a dataset for semantic urban street scene understanding. $5,000$ high quality pixel-level finely annotated images are divided into training, validation, and testing sets with $2,975$ / 500 / $1,525$ images, respectively. It defines 19 categories containing flat, human, vehicle, construction, object, nature, etc.

Segmentation Architectures. We use DeepLabv3+ [11] and PSPNet [80] as the segmentation architectures, powered by MobileNetV2 [58] and ResNet101 [27], respectively, both of which are pre-trained on ImageNet [38].

We follow closely the training procedures of the base architectures when training the baseline model with the standard pixel-wise softmax cross-entropy loss. The performance of the final model might be slightly worse from what is reported in the original papers mainly due to two reasons: 1) We do not pre-train on any other segmentation dataset, such as MS COCO [44] dataset. 2) We do not adopt any additional training tricks, such as balance sampling or fine-tuning specific categories.

Hyper-parameters of SegSort. For all the experiments, we use the following hyper-parameters for training SegSort: The dimension of embeddings is 32. The number of clustering in K-Means are set to 25 and 64 for VOC and Cityscapes, respectively. The EM steps in K-Means are set to 10 and 15 for VOC and Cityscapes, respectively. The concentration constant is set to 10. During inference, we use the same hyper-parameters for K-Means segmentation and 21 nearest neighbors for predicting categories.

We use different learning rates and iterations for supervised training with SegSort. For VOC 2012, we train the network with initial learning rate 0.002 for 100k iterations on *train_aug* set and with initial learning rate 0.0002 for 30k iterations on *train* set. For Cityscapes, we train the network with initial learning rate 0.005 for the same 90k iterations as the softmax baseline.

Training on VOC 2012 requires more iterations than the baseline procedure mainly because most images only contain very few categories while the network can only compare segments in 3 batches (2 batches were cached). We find that enlarging the batch size or increasing memory banks might reduce the training iterations. As a comparison, images from Cityscapes contain ample categories, so the training iterations remain the same.

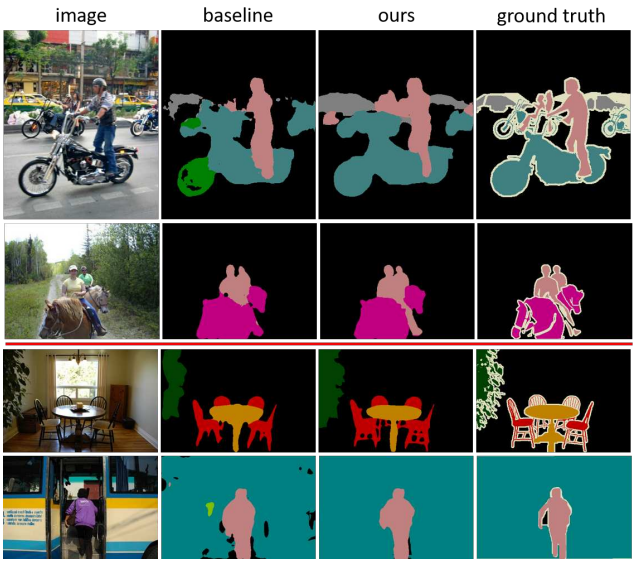

Figure 4. Visual comparison on PASCAL VOC 2012 validation set. We show the visual examples with DeepLabv3+ (upper 2 rows) and PSPNet (lower 2 rows). We observe prominent improvements on thin structures, such as human leg and chair legs. Also, more consistent region predictions can be observed when context is critical, such as wheels in motorcycles and big trunk of buses.

4.2. Fully Supervised Semantic Segmentation

VOC 2012: We summarize the quantitative results of fully supervised semantic segmentation on Pascal VOC 2012 [16] in Table 1, evaluated using mIoU and boundary evaluation following [2, 34] on both validation and testing set.

We conclude that networks trained with SegSort consistently outperform their parametric counterpart (Softmax) by 1.63 to 2.43% in mIoU and by 4.07 to 7.97% in mean boundary f-measure. (Per-class results can be found in the supplementary.) We notice that SegSort with DeepLabv3+ / MNV2 captures better fine structures, such as in 'bike' and 'mbike' while with PSPNet / ResNet-101 enhances more towards detecting small objects, such as in 'boat' and 'plant'.

We present the visual comparison in Fig. 4. We observe

Method	road	swalk	build.	wall	fence	pole	tlight	tsign	veg.	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
Softmax	97.96	83.89	92.22	57.24	59.31	58.89	68.39	77.07	92.18	63.71	94.42	81.80	63.11	94.85	73.54	84.82	67.42	69.34	77.42	76.72
SegSort	98.18	84.86	92.75	55.63	61.57	63.72	71.66	80.01	92.62	64.64	94.65	82.32	62.75	95.08	77.27	87.07	78.89	63.63	77.51	78.15

Table 2. Per-class results on Cityscapes validation set. We conclude that network trained with SegSort outperforms Softmax consistently.

Figure 5. Two examples, correct and incorrect predictions, for segment retrieval for supervised semantic segmentation on VOC 2012 validation set. Query segments (leftmost) are framed by the same color in clustering. (Top) The query segments of rider, horse, and horse outlines can retrieve corresponding semantically relevant segments in the training set. (Bottom) For the failure case, it can be inferred from the retrieved segments that the number tag on the front of bikes is confused by the other number tags or front lights on motorbikes, resulting in false predictions.

prominent improvements on thin structures, such as human legs and chair legs. Also, more consistent region predictions can be found when context is critical, such as wheels in motorcycles and big trunk of buses.

One of the most important features of SegSort is the self-explanatory predictions via nearest neighbor segment retrieval. We therefore demonstrate two examples, correct and incorrect predictions, in Fig. 5. As can be seen, the query segments (on the leftmost) of rider, horse, and horse outlines can retrieve corresponding semantically relevant segments in the training set. For the incorrect example, it can be inferred from the retrieved segments that the number tag on the front of bikes was confused by the other number tags on motorbikes, resulting in false predictions.

Cityscapes: We summarize the quantitative results of fully supervised semantic segmentation on Cityscapes [14] in Table 2, evaluated on the validation set. Due to limited space, visual results are included in the supplementary.

The network trained with SegSort outperforms Softmax consistently. Large objects, *e.g.*, 'bus' and 'truck', are improved thanks to more consistent region predictions while small objects, *e.g.*, 'pole' and 'tlight', are better captured.

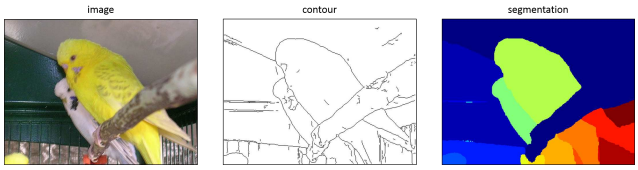

Figure 6. Training data for unsupervised semantic segmentation. We produce fine segmentations (right), HED-owt-ucm, from the contours (middle) detected by HED [73], followed by the procedure in gPb-owt-ucm [49].

unsup. on *train_aug*	sup. on *train_aug*	sup. on *train*	mIoU	f-measure
			49.50	40.86
✓			55.86	44.78
		✓	71.86	55.70
✓		✓	72.47	56.52
	✓	✓	73.35	55.69

Table 3. Quantitative results related to unsupervised semantic segmentation on Pascal VOC 2012 validation set. Our unsupervised trained network (2nd row) outperforms the baseline (1st row) of directly clustering pretrained features using HED-owt-ucm [73] and achieves 76% performance of its supervised counterpart (5th row). Also, the network fine-tuned from unsupervised pre-trained embeddings (4th row) outperforms the one without (3rd row) in both mIoU and boundary f-measure.

4.3. Unsupervised Semantic Segmentation

We train the model using our framework **without** any ground truth labels at any level, pixel-wise or image-wise.

To adapt our approach for unsupervised semantic segmentation, what we need is a good criterion for segmenting an image along visual boundaries, which serves as a pseudo ground truth mask. There is an array of methods that meet the requirement, *e.g.*, SLIC [1] for super-pixels or gPb-owt-ucm [2] for hierarchical segmentation. We choose the HED contour detector [73] pretrained on BSDS500 dataset [2], and follow the procedure in gPb-owt-ucm [2] to produce the hierarchical segmentation, or HED-owt-ucm (Fig. 6).

We train the PSPNet / ResNet-101 network on the same augmented training set on VOC 2012 as in the supervised setting with the same initial learning rate yet for only 10k iterations. The hyper-parameters remain unchanged.

Note that the contour detector only provides visual boundaries without any concept of semantic segments, yet through our feature learning with segment sorting, our method discovers segments of common features – *semantic segmentation without names*.

For the sake of performance evaluation, we assume there is a separate annotated image set available during inference. For each segment under query, we assign a label by the ma-

Figure 7. Segment retrieval results for unsupervised semantic segmentation on VOC 2012 vadlidation set. Query segments (leftmost) are framed by the same color in clustering. As is observed, the embeddings learned by unsupervised SegSort attend to more visual than semantic similarities compared to the supervised setting. Hairs, faces, blue shirts, and wheels are retrieved successfully. The last query segment fails because the texture around knee is more similar to animal skins.

jority vote of its nearest neighbors from that annotated set.

Table 3 shows that our unsupervised trained network outperforms the baseline of directly clustering pretrained features using HED-owt-ucm [73] segmentation and further achieves 76% performance of its supervised counterpart. Together, We also showcase one possible way to make use of the unsupervised learned embedding. The network fine-tuned from unsupervised pre-trained embeddings outperforms the one without. Fig. 7 shows the embeddings learned by unsupervised SegSort attend to more visual than semantic similarities compared to the supervised setting because the fine segmentation formed by contour detectors partitions the image into visually consistent segments. Hairs, faces, blue shirts, and wheels are retrieved successfully. The last query segment fails because the texture around the knee is more similar to animal skins.

Automatic Discovery of Visual Groups. We noticed in the retrieval results that CNNs trained this way can discover visual groups. We wonder if such visual structures actually form different clusters (or fine categories).

We extract all foreground segments in the training set and perform a nearest neighbor based hierarchical agglomerative clustering algorithm FINCH [59]. FINCH merges two points if one is the nearest neighbor of the other (with undirectional link). This procedure can be performed recursively. We start with 1501 segment prototypes and performs FINCH to produce 252, 57, 16, 3, and 1 clusters after each iteration. We visualize some segment groups at the finest level and a coarser level of 16 clusters in Fig. 8.

Figure 8. We perform a nearest neighbor based hierarchical agglomerative clustering, FINCH [59] on foreground segment prototypes to discover visual groups. Top two rows show random samples from two clusters at the finest level. Bottom table displays clusters at a coarser level of 16 clusters. We show four representative segments per cluster.

A bigger picture of how the segments relate to each other from t-SNE [48] can be found in the supplementary.

5. Conclusion

We proposed an end-to-end pixel-wise metric learning approach that is motivated by perceptual organization. We integrated the two essential components, pixel-level and segment-level sorting, in a unified framework, derived from von Mises-Fisher clustering. We demonstrated the proposed approach consistently improves over the conventional pixel-wise prediction approaches for supervised semantic segmentation. We also presented the first attempt for unsupervised semantic segmentation. Intriguingly, the predictions produced by our approach, correct or not, can be inherently explained by the retrieved nearest segments.

Acknowledgements. This research was supported, in part, by Berkeley Deep Drive, NSF (IIS-1651389), DARPA.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, Sabine Süsstrunk, et al. Slic superpixels compared to state-of-the-art superpixel methods. *PAMI*, 2012.

[2] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *PAMI*, 2011.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561*, 2015.

[4] Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 2005.

[5] Serge Belongie, Chad Carson, Hayit Greenspan, and Jitendra Malik. Color-and texture-based image segmentation using em and its application to content-based image retrieval. In *ICCV*, 1998.

[6] Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 94(2):115, 1987.

[7] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *NIPS*, 1994.

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

[9] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.

[10] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, 2016.

[11] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[12] Fan RK Chung and Fan Chung Graham. *Spectral graph theory*. American Mathematical Soc., 1997.

[13] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 2002.

[14] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[15] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015.

[16] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[17] Clément Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *PAMI*, 2013.

[18] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 2004.

[19] Jun Fu, Jing Liu, Yuhang Wang, and Hanqing Lu. Stacked deconvolutional network for semantic segmentation. *arXiv:1708.04943*, 2017.

[20] Raghudeep Gadde, Varun Jampani, Martin Kiefel, Daniel Kappler, and Peter V Gehler. Superpixel convolutional networks using bilateral inceptions. In *ECCV*, 2016.

[21] A. Giusti, D. Ciresan, J. Masci, L.M. Gambardella, and J. Schmidhuber. Fast image scanning with deep max-pooling convolutional neural networks. In *ICIP*, 2013.

[22] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *NIPS*, 2005.

[23] Stephen Gould, Richard Fulton, and Daphne Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[24] Kristen Grauman and Trevor Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, 2005.

[25] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *ICCV*, 2011.

[26] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 1979.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[28] Xuming He, Richard S Zemel, and MA Carreira-Perpindn. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

[29] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, 2015.

[30] Matthias Holschneider, Richard Kronland-Martinet, Jean Morlet, and Ph Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets: Time-Frequency Methods and Phase Space*, 1989.

[31] Jyh-Jing Hwang, Tsung-Wei Ke, Jianbo Shi, and Stella X Yu. Adversarial structure matching for structured prediction tasks. 2019.

[32] Jyh-Jing Hwang and Tyng-Luh Liu. Pixel-wise deep learning for contour detection. *arXiv preprint arXiv:1504.01989*, 2015.

[33] Phillip Isola and Ce Liu. Scene collaging: Analysis and synthesis of natural images with semantic layers. In *ICCV*, 2013.

[34] Tsung-Wei Ke, Jyh-Jing Hwang, Ziwei Liu, and Stella X. Yu. Adaptive affinity fields for semantic segmentation. In *ECCV*, 2018.

[35] Martin Koestinger, Martin Hirzer, Paul Wohlhart, Peter M Roth, and Horst Bischof. Large scale metric learning from equivalence constraints. In *CVPR*, 2012.

[36] Pushmeet Kohli, Philip HS Torr, et al. Robust higher order potentials for enforcing label consistency. *IJCV*, 82(3):302–324, 2009.

[37] Shu Kong and Charless Fowlkes. Recurrent pixel embedding for instance grouping. In *CVPR*, 2018.

[38] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[39] Lubor Ladicky, Christopher Russell, Pushmeet Kohli, and Philip HS Torr. Associative hierarchical crfs for object class image segmentation. In *ICCV*, 2009.

[40] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[41] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1989.

[42] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks with identity mappings for high-resolution semantic segmentation. In *CVPR*, 2017.

[43] Guosheng Lin, Chunhua Shen, Anton van den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *CVPR*, 2016.

[44] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[45] Ce Liu, Jenny Yuen, and Antonio Torralba. Nonparametric scene parsing via label transfer. *PAMI*, 2011.

[46] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv:1506.04579*, 2015.

[47] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[48] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 2008.

[49] Jitendra Malik, Serge Belongie, Thomas Leung, and Jianbo Shi. Contour and texture analysis for image segmentation. *IJCV*, 2001.

[50] Tomasz Malisiewicz and Alexei A Efros. Recognition by association via learning per-exemplar distances. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[51] Mohammadreza Mostajabi, Payman Yadollahpour, and Gregory Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *CVPR*, 2015.

[52] Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*. Springer, 1998.

[53] George Papandreou, Iasonas Kokkinos, and Pierre-Andre Savalle. Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In *CVPR*, 2015.

[54] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *CVPR*, 2017.

[55] Pedro Pinheiro and Ronan Collobert. Recurrent convolutional neural networks for scene labeling. In *ICML*, 2014.

[56] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[57] Bryan Russell, Alyosha Efros, Josef Sivic, Bill Freeman, and Andrew Zisserman. Segmenting scenes by matching image composites. In *NIPS*, 2009.

[58] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[59] M Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In *CVPR*, 2019.

[60] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[61] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.

[62] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *PAMI*, 2000.

[63] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 2009.

[64] X Yu Stella and Jianbo Shi. Multiclass spectral clustering. In *ICCV*, 2003.

[65] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014.

[66] Jay M Tenenbaum and AP Witkin. On the role of structure in vision. *Human and machine vision*, pages 481–543, 1983.

[67] Joseph Tighe and Svetlana Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *ECCV*, 2010.

[68] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: l 2 hypersphere embedding for face verification. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 2017.

[69] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018.

[70] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017.

[71] Zhirong Wu, Alexei A Efros, and Stella X Yu. Improving generalization via scalable neighborhood component analysis. In *ECCV*, 2018.

[72] Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination. In *CVPR*, 2018.

[73] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, 2015.

[74] Allen Y Yang, John Wright, Yi Ma, and S Shankar Sastry. Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 2008.

[75] Tien-Ju Yang, Maxwell D Collins, Yukun Zhu, Jyh-Jing Hwang, Ting Liu, Xiao Zhang, Vivienne Sze, George Papandreou, and Liang-Chieh Chen. Deeperlab: Single-shot image parser. *arXiv preprint arXiv:1902.05093*, 2019.

[76] Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *CVPR*, 2012.

[77] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Learning a discriminative feature network for semantic segmentation. In *CVPR*, 2018.

[78] Stella X Yu and Jianbo Shi. Segmentation given partial grouping constraints. *PAMI*, 2004.

[79] Zhenli Zhang, Xiangyu Zhang, Chao Peng, Dazhi Cheng, and Jian Sun. Exfuse: Enhancing feature fusion for semantic segmentation. In *ECCV*, 2018.

[80] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.