Unsupervised Learning of Supervoxel Embeddings for Video Segmentation

Mehran Khodabandeh[1,2], Srikanth Muralidharan[1], Arash Vahdat[1], Nazanin Mehrasa[1], Eduardo M. Pereira[3], Shin'ichi Satoh[2] and Greg Mori[1]

[1]Simon Fraser University
[2]National Institute of Informatics
[3]INESC TEC
{*mkhodaba, smuralid, avahdat, nmehrasa, mori*}*@sfu.ca, ejmp@inesctec.pt, satoh@nii.ac.jp*

Abstract

We present an algorithm for learning a feature representation for video segmentation. Standard video segmentation algorithms utilize similarity measurements in order to group related pixels. The contribution of our paper is an unsupervised method for learning the feature representation used for this similarity. The feature representation is defined over video supervoxels. An embedding framework learns a feature mapping for supervoxels in an unsupervised fashion such that supervoxels with similar context have similar embeddings. Based on the learned representation, we can merge similar supervoxels into spatio-temporal segments. Experimental results demonstrate the effectiveness of this learned supervoxel embedding on standard benchmark data.

1 Introduction

Effective video segmentation can be achieved if a robust measurement of pixel similarity can be devised. However, this task is extremely challenging due to various complexities that arise from the variability in object appearance, occlusions in the scene, and camera motion, among other challenges. In this paper we present a method for unsupervised learning of feature representations for video segmentation.

There is a rich history of algorithms for video segmentation within the computer vision literature. Seminal work includes algorithms for analyzing layers of moving objects, probabilistic models, and graph-based segmentation algorithms [18, 15, 7]. A key facet of methods for video segmentation is deciding upon a representation for pixels such that pixels from similar objects are similar in feature representation.

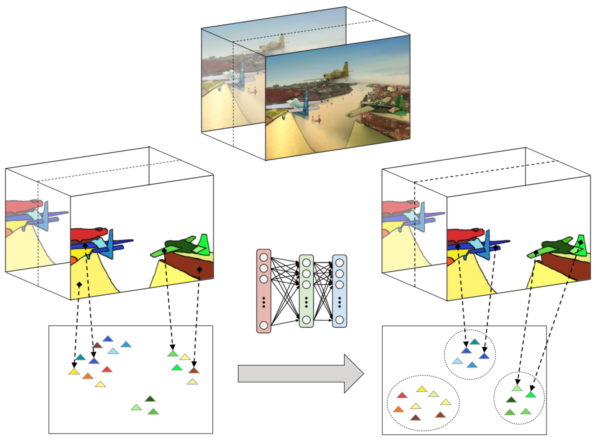


Figure 1: In the embedding space supervoxels of the same object are brought together while being separated from supervoxels of the other objects.

Broadly speaking, two veins of work exist for determining a feature representation for video segmentation. One involves supervised learning: a training set of segmentations with semantic object category labels is used. From this, a feature representation is learned that can distinguish between objects, and is then applied to segmentation (e.g. [10]).

The second vein uses unsupervised learning / hand-crafting feature representations. For example, principles from Gestalt psychology such as *common fate* have been used to motivate the design of feature representations based on motion similarity in classic segmentation work [15]. More recent segmentation systems combine a set of such hand-crafted feature representations (e.g. [12]).

The main idea of our paper is to leverage these unsupervised techniques to learn better feature representations for segmentation. We operate in a graph-based segmentation framework on supervoxels. For example, two supervoxels that have similar motion are likely to belong to the same object. This fact can be used to refine an appearance-based feature representation, placing these supervoxels closer together. Likewise, two supervoxels that have similar appearance are likely to belong to the same object. This could be used to refine a motion-based feature presentation for a supervoxel.

We operationalize this by learning an embedding representation for supervoxels, in which supervoxels that belong to the same segment have similar representations. For example, consider the video shown in Fig. 1. Supervoxels that belong to the airplane should have a similar feature representation. However, their initial appearance features are different. Our unsupervised learning algorithm uses the fact that nearby, similarly moving supervoxels are likely to be from similar objects to learn an embedding that moves the airplane supervoxels closer together.

This idea is similar to the *word2vec* algorithm [13], which learns an embedding that is able to predict a word from the context of the neighboring words. In our problem the context of each supervoxel is determined by the spatio-temporally nearby supervoxels. Similarly, in this work, we aim to learn an embedding of supervoxels from videos, such that contextually similar segments would have similar representations in the embedded space. After learning the embeddings, we use a graph-based partitioning algorithm using the embedded features to produce the final segmentation.

We conduct our experiments on the video segmentation benchmark dataset [6]. Through our experiments, we show that the learned embeddings can improve the performance of video segmentation using appearance-based [11] or motion-based [3] features.

This paper is organized as follows. Section 2 gives an overview of the related work, Section 3 describes our method, Section 4 contains details of our experiments, and Section 5 concludes our work.

2 Related Work

Video Segmentation: Video segmentation is a challenging area of research in computer vision, and there exists an abundant literature pertaining to it [4, 5, 17, 20, 10].

Khoreva et al. use a graph based supervised video segmentation algorithm [10], where different classifiers are used region-wise to learn a graph representation for the video frames, before eventually performing the segmentation using standard graph partitioning methods. Badrinarayanan et al. [1] develop a semi-supervised approach involving a mixture of trees based probabilistic graphical model, where superpixel labels are obtained by variational inference. Chang et al. build an unsupervised approach to video segmentation [2]. A conditional random field is employed to assign labels to all the pixels, using higher order potentials with softer label consistency constraints in order to identify complex spatio-temporal tubes that actually correspond to the same label. Zhang et al. [20] first identify a primary object using a layered directed acyclic graph based approach that assumes that objects are spatially cohesive and follow smooth trajectory patterns. Once the primary object models are extracted, segmentation is performed by invoking a scoring function that segments regions with high optical flow gradients from the background.

Deep Embedding Methods: Deep learning based embedding approaches have gained a lot of attention recently, and were shown to be useful under various scenarios. Karpathy et al. [8] work on dense image descriptions. A weakly supervised bimodal text-image embedding is performed to generate text phrases for corresponding regions of the image, which are then used to generate dense image descriptions. Ramanathan et al. [14], learn temporal embeddings using a large unlabeled video database, and show that these embeddings improve the performance of several video related tasks. Srivatsava et al. [16] make use of a recurrent neural network architecture and a large video classification database [9] to learn unsupervised representations for contiguous video frames. They eventually show that these embeddings significantly help in activity recognition.

3 Our Method

Our goal in this paper is to develop a principled framework for learning contextually-aware embeddings for video segments in an unsupervised setting. Video segments with the same "context" are those which are in the same vicinity. Therefore, contextually-aware embeddings proposed in this paper can be used to group small video segments into larger groups which represent either semantically consistent objects or actions that intuitively share a common context.

The main focus of this paper is learning a similarity measure for supervoxels, which is used for video segmentation. We build our segmentation approach on top of two recently proposed algorithms. Xu et al.'s supervoxel extraction algorithm [19] is employed to generate a large set of small supervoxels for a video. The graph partitioning algorithm proposed in [5] is then used to group supervoxels based on embeddings trained by our proposed model.

This section presents our method as follows. Our embedding approach is presented in Sec. 3.1. Sec. 3.2 explains the neighbor/negative selection strategy, followed by Sec. 3.3 that contains details about supervoxel extraction and the features used. Finally, we conclude the section with the video segmentation algorithm that leverages the similarity measure to construct and partition the graph representation in Sec. 3.4.

3.1 Learning an Embedding Function

As illustrated in Figure 1, we aim at learning an embedding function ϕ that maps the supervoxel feature space to another m-dimensional space. In this space, contextually similar supervoxels should have similar vector representations when compared via the dot product similarity measurement.

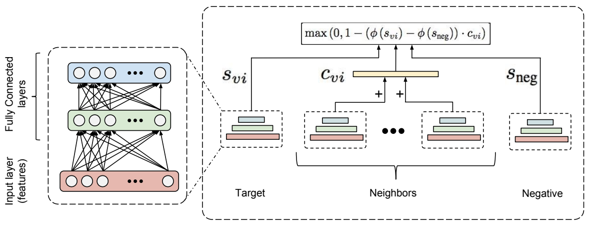
We implement the embedding function ϕ using a neural network. The input to the network is a feature vector describing a supervoxel (e.g. motion features or appearance features from a deep network). The embedding function network is constructed on top of this input. The network we use has two fully connected layers, each followed by a nonlinear activation function. In this work we used rectified linear units (ReLUs) as the activation functions. Figure 2 illustrates the structure of our network.

Inspired by the word2vec idea, we wish to perform a warping of the original supervoxel space to a space in which a given target supervoxel is separated from "other" supervoxels using the neighboring supervoxels as much as possible. Towards this goal, we aim to formulate this problem as an embedding learning problem that is trained to separate the neighbouring supervoxels from negatives by a large margin, while bringing the target closer to the neighbors.

Assume $S_V = \{s_1, s_2, \ldots, s_n\}$ is the set of all the supervoxels extracted from video V. For each supervoxel $s_i \in S_V$, we define sets of positive and negative supervoxels. N_i^+ denotes the set of neighbouring supervoxels that are close to s_i in space and time (and are likely to be from the same object), and N_i^- is the set of negative supervoxels that are distant from s_i (in other words, a limited set of supervoxels that are *not* in the same object that s_i belongs to). The details of the neighbour/negative selection strategy are explained in Section 3.2.

We define the representation of the object that s_i belongs to:

$$c_i = \frac{1}{|N_i^+|} \sum_{s_{\text{nei}} \in N_i^+} \phi(s_{\text{nei}}) \tag{1}$$



$\max\left(0, 1 - \left(\phi\left(s_{vi}\right) - \phi\left(s_{\text{neg}}\right)\right) \cdot c_{vi}\right)$
Fully Connected layers
Input layer (features)
s_{vi}
c_{vi}
s_{neg}
+
+
Target
Neighbors
Negative

Figure 2: Architecture of our Embedding Network. Weights between layers with similar colors are shared. I.e. identical embedding applies to target, negative, and neighbors.

Where s_{nei} is a neighbour of s_i and a member of its neighbour set, N_i^+.

Here, we use the hinge loss as an objective function to learn the embedding function ϕ:

$$E(\phi) = \sum_{s_i \in S_V} \sum_{s_{\text{neg}} \in N_i^-} \max(0, 1 - (\phi(s_i) - \phi(s_{\text{neg}})) \cdot c_i) \tag{2}$$

Here, S_V is the set of supervoxels in video V, s_i is the i^{th} supervoxel, and s_{neg} is an instance of a negative of s_i, which is a member of N_i^-.

For a given video V, the goal is to find a $\phi(\cdot)$ that minimizes this objective function defined in Equation 2. For each video we train an individual neural network with back-propagation with the standard stochastic gradient descent (SGD) method to perform this minimization.

The intuition behind Equation 2 is that by minimizing $E(\phi)$ and therefore maximizing $\phi(s_i) \cdot c_i - \phi(s_{\text{neg}}) \cdot c_i$, we force the embedding to separate a negative supervoxel from supervoxels of the object that s_i belongs to, while bringing s_i closer to them. Note that here we use dot product similarity, so "separating" and "bringing close" mean "more difference" and "less difference" in the angle between two vectors, respectively.

3.2 Negative and Neighbor Selection Strategy

In our algorithm, the selection of neighbors and negatives are critical to the performance of our algorithm. For a given supervoxel s_i, we assume that the embedding of s_i is similar to the embedding of the set of supervoxels that are close to s_i in space and time. In addition, based on our assumption, the embedding of s_i should be different from the embedding of a negative supervoxel, s_{neg}, which is not from the same object as s_i. Essentially, any supervoxel that is far from s_i and is not from the same object is a candidate negative. However, in our experiments we found that not all the far supervoxels are proper negative candidates; and not all the neighboring supervoxels represent the object that s_i is extracted from. As for negatives, based on our observations we decided to choose "hard negatives". Hard negatives of s_i are the closest supervoxels to s_i which are not in the set of neighbours (N_i^+).

Therefore, for choosing the hard negatives and neighbours of supervoxel s_i, we first create a pool of candidate supervoxels by extracting a set of supervoxels that are close to s_i **spatio-temporally**. From this set we pick both "neighbors" and "negatives." We use center of mass to represent the spatial position of a supervoxel, and we use Euclidean distance to measure the distance between suprevoxels. Then we sort the set of candidates based on their distance to s_i in the **feature space**. Now we select the K closest ones as "neighbors" and the F farthest ones as "negatives" (Figure 4). In our experiments we tuned these parameters to obtain the best performance.

3.3 Supervoxels and Feature Extraction

Given an input video V we use Xu's [19] algorithm to extract the initial supervoxel set S_V. Xu et al. [19] produces supervoxels in a hierarchical manner, where at each level of the hierarchy supervoxels are constructed by merging the previous level in the hierarchy. Therefore the lower the level in the hierarchy the more supervoxels are available. We use the lowest level that is computationally affordable, level 8 with about 2000 supervoxels constructed from a 120 frame video. Other parameters are set as the default of the code. We use a simple averaging function as the aggregation method for computing supervoxel feature:

$$s_i = \sum_{j=1}^{P_i} \frac{p_j}{P_i} \tag{3}$$

where i = 1, 2, ..., n. Here n is equal to the number of supervoxels, P_i is equal to the number of pixels belonging to supervoxel i, and p_j is the value of the feature map of the j^{th} pixel of the supervoxel.

We extract two different pixel-level feature types. The first representation is HOF [3], representing pixel motion from optical flow. The second descriptor is the FCN [11] segmentation mask, which represents each pixel with a 21 class score map obtained by passing a frame as input to the model [1] that was trained on the PASCAL VOC dataset. We used default parameters provided by the code for both HOF and FCN.

[1]`http://dl.caffe.berkeleyvision.org/fcn-8s-pascal.caffemodel`

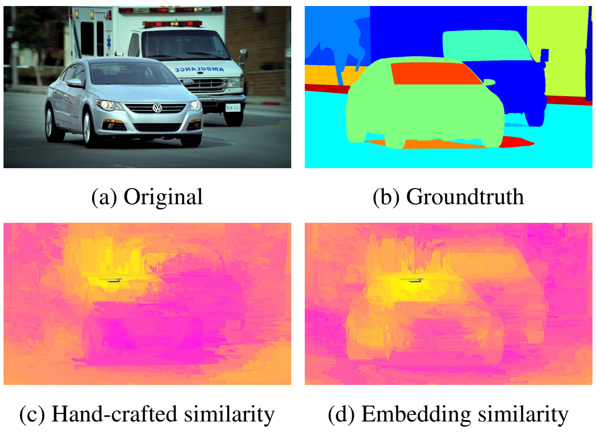

(a) Original (b) Groundtruth
(c) Hand-crafted similarity (d) Embedding similarity

Figure 3: Given a target supervoxel (the blue one on the front car in (c)), we color the other supervoxels based on their cosine similarity to the target. Intensity of pink and yellow indicates magnitude of dissimilarity and similarity, respectively. After the embedding, supervoxels on the front car are more similar to the target (blue) supervoxel.

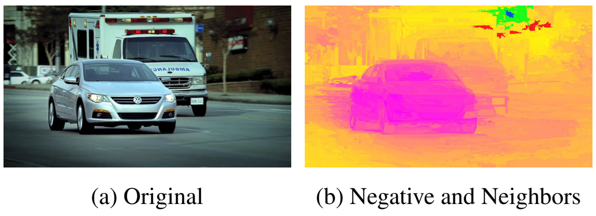
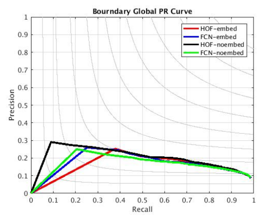

(a) Original (b) Negative and Neighbors

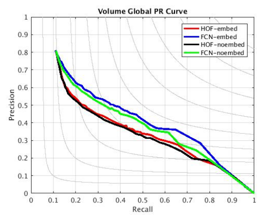
Figure 4: A sample of selected negative supervoxels (red region) and neighboring supervoxels (green region) for a given supervoxel (blue).

3.4 Video Segmentation

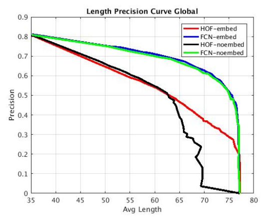
In our work, for each video we train a separate network. Given a trained network and a supervoxel s_i, we obtain the learned representation of s_i, which is $\phi(s_i)$ by feeding its feature vector to the network and collecting the values of the network output. We define the similarity matrix $[W]_{n\times n}$ that contains pairwise affinities as: $W = \Phi\Phi^T$. Here $[\Phi]_{n\times m}$ is the embedding of all the supervoxels. Then we use Galasso's algorithm [4] to re-weight the matrix W based on the supervoxels in the lower level of hierarchy. The i-th row of W contains the similarity between supervoxel s_i and all other supervoxels. Figure 3 shows how the similarity between a particular supervoxel and other supervoxels change after the embedding. The resulting similarity matrix is then used in spectral clustering to obtain a final segmentation.



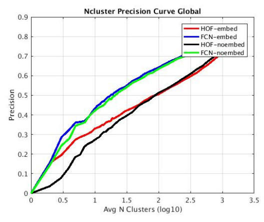
(a) **BPR**



(b) **VPR**



(c) **LPC**



(d) **NCluster**

Figure 5: Boundary Precision Recall(BPR), Volume Precision Recall(VPR), Length Precision Curve(LPC), Ncluster video segmentation statistics obtained using FCN and HOF embeddings

4 Experiments

We evaluate our method on the VSB100 database [6]. It consists of a diverse set of video instances. The database specifies training and testing sets of videos. Our method is an unsupervised learning method in which we learn an embedding for supervoxels. In our experiments we perform this learning one video at a time, and hence we use only the test set for evaluating our method. The test set consists of 60 videos. We used HOF and FCN as features for testing the efficacy of neural network based embedding . As a baseline, for each feature we evaluate the segmentation performance using the corresponding feature without embedding. We follow the evaluation protocol in VSB100: Volume Precision Recall (VPR), Boundary Precision Recall (BPR), Length Precision Recall (LPR) and number of clusters (NCluster) statistics are reported; the Volume Precision Recall curve's best F-Measure is used as an aggregate measure of performance.

We use Caffe to implement our learning framework. We use a separate network for each video. We tuned the parameters for each network by trying a range of parameters and choosing the ones with highest F-measure (VPR). The values that we chose for the parameters are as follows: number of neighbors $\in \{4, 8\}$, number of negatives $\in \{6, 12\}$, batch size $\in \{128, 256\}$. In all our experiments, we use an initial learning rate of 0.001. The learning rate is reduced by a factor of 0.1 after each 400 iterations. We perform 20 epochs of stochastic gradient descent, which is sufficient for convergence for this dataset.

Method	VPR
HOF without embedding	44.30
HOF with embedding	46.50
FCN without embedding	51.39
FCN with embedding	**53.68**
Concat HOF, FCN without embedding	49.70
Concat HOF, FCN and embed	52.52

Table 1: Comparison of F-Measure score of Volume Precision Rate (VPR) obtained using feature after embedding versus features without embedding on VSB100 test set.

From the numbers tabulated in Table 1, it is evident that embedding features improves the segmentation performance compared to its non-embedded counterpart by 2.2% on HOF, 2.3% on FCN, and 2.8% on combination of HOF and FCN. Fig. 5 shows different segmentation statistics obtained in these experiments.

Note that state-of-the-art supervised learning methods [10] do obtain higher performance on this task. When using pixel-level labels in training data, 70% VPR can be achieved. However, acquiring pixel-level labels can be expensive. Our experiments demonstrate the effectiveness of learning embeddings without need for additional supervision.

4.1 Discussion

We found that the performance of our model is high in the videos where there are heterogeneous object(s) with different spatial and temporal features. This is because, firstly, our model is constructed on the hypothesis that the supervoxels that are close enough in the spatial dimension, and also in the feature space used to decide on the neighbors/negatives (e.g. HOF in case of FCN embeddings, and vice versa), are from the same object.

Conversely, it is difficult for our model to produce good segmentation in videos that have numerous smaller and visually similar objects. Firstly this violates our neighbor/negative hypothesis construction. Secondly, the evaluation is originally intended for testing semantic segmentation algorithms and might require higher-level semantic information to make these distinctions. These features of our model are evident from the sample visualizations shown in Fig. 3 and 6.

5 Conclusion

In this work, we tackled the problem of video segmentation using an unsupervised deep embedding ap-

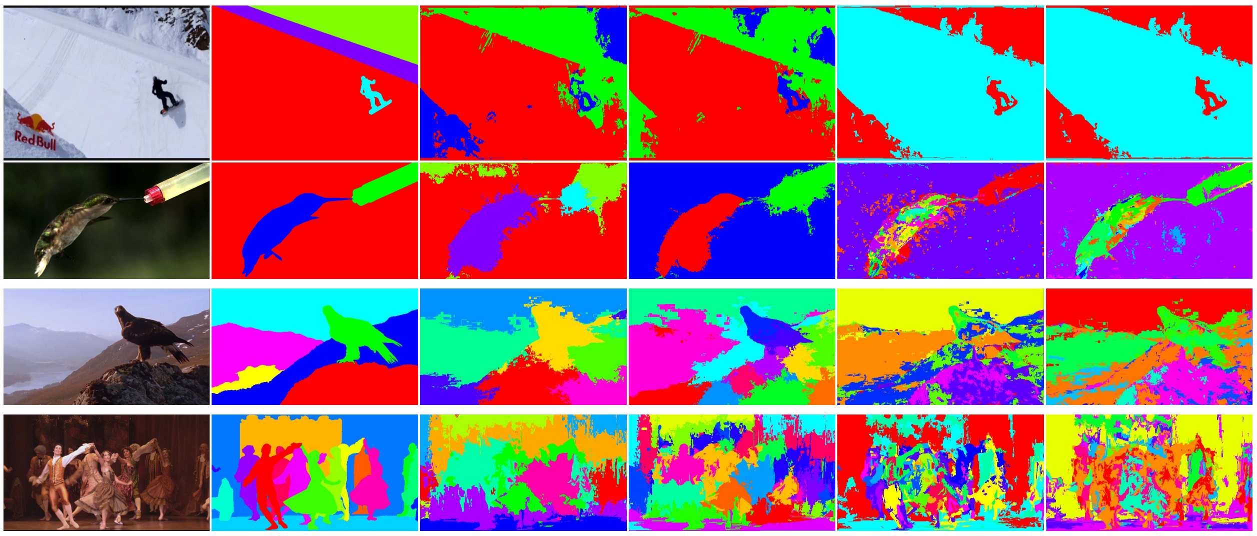


Figure 6: Visualization of segmentations obtained by applying our method using different features. Each colour in a segmentation represents a distinct segment obtained by applying that method.

proach. This embedding provides a discriminative mapping of supervoxel feature representations for a given video. We found that this learning has the potential to help in separating a supervoxel's neighbor feature representation from the feature representation of a distant supervoxel. We demonstrated this embedding on different spatial/temporal features.

References

[1] V. Badrinarayanan, I. Budvytis, and R. Cipolla. Mixture of trees probabilistic graphical model for video segmentation. *IJCV*, 2014.

[2] J. Chang, D. Wei, and J. Fisher. A video representation using temporal superpixels. *CVPR*, 2013.

[3] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. *ECCV*, 2006.

[4] F. Galasso, R. Cipolla, and B. Schiele. Video segmentation with superpixels. *ACCV*, 2012.

[5] F. Galasso, M. Keuper, T. Brox, and B. Schiele. Spectral graph reduction for efficient image and streaming video segmentation. *CVPR*, 2014.

[6] F. Galasso, N. Nagaraja, T. Cardenas, T. Brox, and B. Schiele. A unified video segmentation benchmark: Annotation, metrics and analysis. *ICCV*, 2013.

[7] M. Irani and P. Anandan. A unified approach to moving object detection in 2d and 3d scenes. *PAMI*, 1998.

[8] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *CVPR*, 2015.

[9] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. *CVPR*, 2014.

[10] A. Khoreva, F. Galasso, M. Hein, and B. Schiele. Classifier based graph construction for video segmentation. In *CVPR*, 2015.

[11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, 2015.

[12] J. Lu, R. Xu, and J. J. Corso. Human action segmentation with hierarchical supervoxel consistency. *CVPR*, 2015.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[14] V. Ramanathan, K. Tang, G. Mori, and L. Fei-Fei. Learning temporal embeddings for complex video analysis. *ICCV*, 2015.

[15] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. *ICCV*, 1998.

[16] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. *arXiv preprint arXiv:1502.04681*, 2015.

[17] N. Sundaram and K. Keutzer. Long term video segmentation through pixel level spectral clustering on gpus. *ICCV*, 2011.

[18] Y. Weiss and E. H. Adelson. A unified mixture framework for motion segmentation: incorporating spatial coherence and estimating the number of models. *CVPR*, 1996.

[19] C. Xu, C. Xiong, and J. J. Corso. Streaming hierarchical video segmentation. *ECCV*, 2012.

[20] D. Zhang, O. Javed, and M. Shah. Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. *CVPR*, 2013.