Temporal Tessellation: A Unified Approach for Video Analysis

Dotan Kaufman[1], Gil Levi[1], Tal Hassner[2,3], and Lior Wolf[1,4]

[1]The Blavatnik School of Computer Science , Tel Aviv University, Israel
[2]Information Sciences Institute , USC , CA, USA
[3]The Open University of Israel, Israel
[4]Facebook AI Research

Abstract

We present a general approach to video understanding, inspired by semantic transfer techniques that have been successfully used for 2D image analysis. Our method considers a video to be a 1D sequence of clips, each one associated with its own semantics. The nature of these semantics – natural language captions or other labels – depends on the task at hand. A test video is processed by forming correspondences between its clips and the clips of reference videos with known semantics, following which, reference semantics can be transferred to the test video. We describe two matching methods, both designed to ensure that (a) reference clips appear similar to test clips and (b), taken together, the semantics of the selected reference clips is consistent and maintains temporal coherence. We use our method for video captioning on the LSMDC'16 benchmark, video summarization on the SumMe and TV-Sum benchmarks, Temporal Action Detection on the Thumos2014 benchmark, and sound prediction on the Greatest Hits benchmark. Our method not only surpasses the state of the art, in four out of five benchmarks, but importantly, it is the only single method we know of that was successfully applied to such a diverse range of tasks.

1. Introduction

Despite decades of research, video understanding still challenges computer vision. The reasons for this are many, and include the hurdles of collecting, labeling and processing video data, which is typically much larger yet less abundant than images. Another reason is the inherent ambiguity of actions in videos which often defy attempts to attach dichotomic labels to video sequences [26]

Rather than attempting to assign videos with single *action labels* (in the same way that 2D images are assigned object classes in, say, the ImageNet collection [47]) an increasing number of efforts focus on other representations

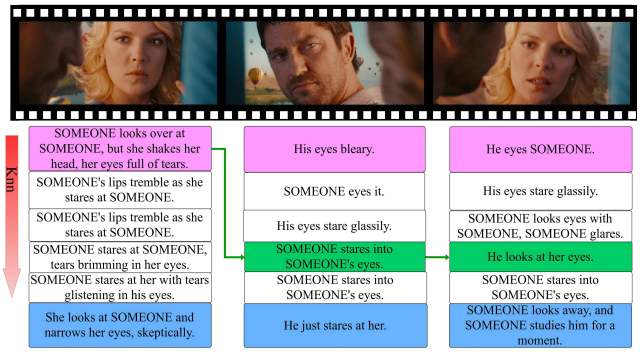

Figure 1. **Tessellation for temporal coherence**. For video captioning, given a query video (top), we seek reference video clips with similar semantics. Our tessellation ensures that the semantics assigned to the test clip are not only the most relevant (the five options for each clip) but also preserve temporal coherence (green path). Ground truth captions are provided in blue.

for the semantics of videos. One popular approach assigns videos with natural language text annotations which describe the events taking place in the video [4, 44]. Systems are then designed to automatically predict these annotations. Others attach video sequences with numeric values indicating what parts of the video are more interesting or important [13]. Machine vision is then expected to determine the importance of each part of the video and summarize videos by keeping only their most important parts.

Although impressive progress was made on these and other video understanding problems, this progress was often made disjointedly: separate specialized systems were utilized that were tailored to obtain state of the art performance on different video understanding problems. Still lacking is a *unified* general approach to solving these different tasks.

Our approach is inspired by recent 2D dense correspondence estimation methods (e.g., [16, 34]). These methods were successfully shown to solve a variety of image understanding problems by transferring per-pixel semantics from reference images to query images. This general ap-

proach was effectively applied to a variety of tasks, including single view depth estimation, semantic segmentation and more. We take an analogous approach, applying similar techniques to 1D video sequences rather than 2D images.

Specifically, image based methods combine local, per-pixel appearance similarity with global, spatial smoothness. We instead combine local, per-region appearance similarity with global semantics smoothness, or *temporal coherence*. Fig. 1 offers an example of this, showing how temporal coherence improves the text captions assigned to a video.

Our contributions are as follows: **(a)** We describe a novel method for matching test video clips to reference clips. References are assumed to be associated with semantics representing the task at hand. Therefore, by this matching we transfer semantics from reference to test videos. This process seeks to match clips which share similar appearances while maintaining semantic coherency between the assigned reference clips. **(b)** We discuss two techniques for maintaining temporal coherency: the first uses unsupervised learning for this purpose whereas the second is supervised.

Finally, **(c)**, we show that our method is general by presenting state of the art results on three recent and challenging video understanding tasks, previously addressed separately: Video caption generation on the LSMDC'16 benchmark [46], video summarization on the SumMe [13] and TVSum [53] benchmarks, and action detection on the THUMOS'14 benchmark [20]. In addition, we report results comparable to the state of the art on the Greatest Hits benchmark [38] for sound prediction from video. Importantly, we will *publicly release our code and models*.[1]

2. Related work

Video annotation. Significant progress was made in the relatively short time since work on video annotation / caption generation began. Early methods such as [1, 18, 37, 68] attempted to cluster captions and videos and applied this for video retrieval. Others [12, 27, 58] generated sentence representations by first identifying semantic video content (e.g., verb, noun, etc.) using classifiers tailored for particular objects and events. They then produce template based sentences. This approach, however, does not scale well, since it requires substantial efforts to provide suitable training data for the classifiers, as well as limits the possible sentences that the model can produce.

More recently, and following the success of image annotation systems based on deep networks such as [8, 64], similar techniques were applied to videos [8, 55, 62, 69]. Whereas image based methods used convolutional neural networks (CNN) for this purpose, application to video involve temporal data, which led to the use of recurrent neural networks (RNN), particularly long short-term memory networks (LSTM) [17]. We also use CNN and LSTM models but in fundamentally different ways, as we later explain in Sec. 4.

Video summarization. This task involves selecting the subset of a query video's frames which represents its most important content. Early methods developed for this purpose relied on manually specified cues for determining which parts of a video are important and should be retained. A few such examples include [5, 41, 53, 73].

More recently, the focus shifted towards supervised learning methods [11, 13, 14, 74], which assume that training videos also provide manually specified labels indicating the importance of different video scenes. These methods sometimes use multiple individual-tailored decisions to choose video portions for the summary [13, 14] and often rely on the determinantal point process (DPP) in order to increase the diversity of selected video subsets [3, 11, 74]. Unlike video description, LSTM based methods were only considered for summarization very recently [75]. Their use of LSTM is also very different from ours.

Temporal action detection. Early work on video action recognition relied on hand crafted space-time features [24, 25, 30, 65]. More recently, deep methods have been proposed [19, 21, 57], many of which learn deep visual and motion features [32, 51, 60, 67]. Along with the development of stronger methods, larger and more challenging benchmarks were proposed [15, 26, 28, 54]. Most datasets, however, used trimmed, temporally segmented videos, i.e: short clips which contain only a single action.

Recently, similar to the shift toward classification combined with localization in object recognition, some of the focus shifted toward more challenging and realistic scenarios of classifying untrimmed videos [10, 20]. In these datasets, a given video can be up to a few minutes in length, different actions occur at different times in the video and in some parts of the video no clear action occurs. These datasets are also used for classification, i.e. determining the main action taking place in the video. A more challenging task, however, is the combination of classification with temporal detection: determining which action, if any, is taking place at each time interval in the video.

In order to tackle temporal action detection in untrimmed videos, Yuan et al. [72] encode visual features at different temporal resolutions followed by a classifier to obtain classification scores at different time scales. Escorcia et al [9] focus instead on a fast method for obtaining action proposals from untrimmed videos, which later can be fed to an action classifier. Instead of using action classifiers, our method relies on matching against a gallery of temporally segmented action clips.

[1] See: `www.github.com/dot27/temporal-tessellation`

3. Preliminaries

Our approach assumes that test videos are partitioned into clips. It then matches each test clip with a reference (*training*) clip. Matching is performed with two goals in mind. First, at the clip level, we select reference clips which are visually similar to the input. Second, at the video level, we select a sequence of clips which best preserves the temporal semantic coherency. Taken in sequence, the order of selected, reference semantics should adhere to the temporal manner in which they appeared in the training videos.

Following this step, the semantics associated with selected reference clips can be transferred to test clips. This allows us to reason about the test video using information from our reference. This approach is general, since it allows for different types of semantics to be stored and transferred from reference, training videos to the test videos. This can include, in particular, textual annotations, action labels, manual annotations of interesting frames and others. Thus, different semantics represent different video understanding problems which our method can be used to solve.

3.1. Encoding video content

We assume that training and test videos are partitioned into sequences of clips. A clip $\mathbf{C}$ consists of a few consecutive frames $\mathbf{I}_i, i \in 1..n$ where n is the number of frames in the clip. Our tessellation approach is agnostic to the particular method chosen to represent these clips. Of course, The more robust and discriminative the representation, the better we expect our results to be. We, therefore, chose the following two step process, based on the recent state of the art video representations of [31].

Step 1: Representing a single frame. Given a frame $\mathbf{I}_i$ we use an off the shelf CNN to encode its appearance. We found the VGG-19 CNN to be well suited for this purpose. This network was recently proposed in [52] and used to obtain state of the art results on the ImageNet, large scale image recognition benchmark (ILSVRC) [47]. In their work, [52] used the last layer of this network to predict ImageNet class labels, represented as one-hot encodings. We instead treat this network as a feature transform function $f : \mathbf{I} \mapsto \boldsymbol{a}'$ which for image (frame) $\mathbf{I}$ returns the $4,096D$ response vector from the penultimate layer of the network.

To provide robustness to local translations, we extract these features by oversampling: $\mathbf{I}$ is cropped ten times at different offsets around the center of the frame. These cropped frames are normalized by subtracting the mean value of each color channel and then fed to the network. Finally, the ten $4,096D$ response vectors returned by the network are pooled into a single vector by element-wise averaging. Principle component analysis (PCA) is further used to reduce the dimensionality of these features to $500D$, giving us the final, per frame representation $\mathbf{a} \in \mathbb{R}^{500}$.

Step 2: Representing multiple frames. Once the frames are encoded, we pool them to obtain a representation for the entire clip. Pooling is performed by Recurrent Neural Network Fisher Vector (RNN-FV) encoding [31].

Specifically, We use their RNN, trained to predict the feature encoding of a future frame, $\mathbf{a}_i$, given the encodings for its k preceding frames, $(\mathbf{a}_{i-k}, ..., \mathbf{a}_{i-1})$. This RNN was trained on the training set from the Large Scale Movie Description Challenge [46], containing roughly 100K videos. We apply the RNN-FV to the representations produced for all of the frames in the clip. The gradient of the last layer of this RNN is then taken as a 100,500D representation for the entire sequence of frames in $\mathbf{C}$. We again use PCA for dimensionality reduction, this time mapping the features produced by the RNN-FV to 2,000D dimensions, resulting in our pooled representation $\mathbf{A} \in \mathbb{R}^{2,000}$. We refer to [31] for more details about this process.

3.2. Encoding semantics

As previously mentioned, the nature of the semantics associated with a video depends on the task at hand. For tasks such as action detection and video summarization, for which the supervision signal is of low dimension, the semantic space of the labels has only a few bits of information per segment and is not discriminative enough between segments. In this case, we take the semantic space $\mathbf{V}^S$ to be the same as the appearance space $\mathbf{V}^A$ and take both to be the pooled representation $\mathbf{A}$.

Textual semantics In video captioning, in which the text data provides a rich source of information, our method largely benefits from having a separate semantic representation that is based on the label data.

We tested several representations for video semantics and chose the recent Fisher Vector of a Hybrid Gaussian-Laplacian Mixture Model (FV-HGLMM) [23], since it provided the best results in our initial cross-validation experiments.

Briefly, we assume a textual semantic representation, $\mathbf{s}$ for a clip $\mathbf{C}$, where $\mathbf{s}$ is a string containing natural language words. We use word2vec [35] to map the sequence of words in $\mathbf{s}$ to a sequence of vectors, $(s_1, ..., s_m)$, where m is the number of words in $\mathbf{s}$ and can be different for different clips. FV-HGLMM then maps this sequence of numbers to a vector $\mathbf{S} \in \mathbb{R}^M$ of fixed dimensionality, M.

FV-HGLMM is based on the well-known Fisher Vectors (FV) [40, 50, 56]. The standard Gaussian Mixture Models (GMM) typically used to produce FV representations are replaced here with a Hybrid Gaussian-Laplacian Mixture Model which was shown in [23] to be effective for image annotation. We refer to that paper for more details.

3.3. The joint semantics video space (SVS)

Clip representations and their associated semantics are all mapped to the joint SVS. We aim to map the appearance of each clip and its assigned semantics to two neighboring points in the SVS. By doing so, given an *appearance* representation for a query clip, we can search for potential *semantic* assignments for this clip in our reference set using standard Euclidean distance. This property will later become important in Sec. 4.2.

In practice, all clip appearance representations $\mathbf{A}$ and their associated semantic representations $\mathbf{S}$ are jointly mapped to the SVS using regularized Canonical Correlation Analysis (CCA) [63] where the CCA mapping is trained using the given ground truth semantics. In our experiments, the CCA regularization parameter is fixed to be a tenth of the largest eigenvalue of the cross domain covariance matrix computed by CCA. For each clip, CCA projects $\mathbf{A}$ and $\mathbf{S}$ (appearance and semantics, respectively) to $\mathbf{V}^A$ and $\mathbf{V}^S$.

4. Tessellation

We assume a data set of training (reference) clips, $\mathbf{V}_j^A$, and their associated semantics, $\mathbf{V}_j^S$, represented as described in Sec. 3. Here, $j \in 1..N$ indexes the entire data set of N clips. Since these clips may come from different videos, j does not necessarily reflect temporal order.

Given a test video, we process its clips following 3.1 and 3.3, obtaining a sequence of clip representations, $\mathbf{U}_i^A$ in the SVS, where consecutive index values for $i \in M$, represent consecutive clips in a test video with M clips. Our goal is to match each $\mathbf{U}_i^A$ with a data set *semantic* representation $\mathbf{V}_{j_i}^S$ while optimizing the following two requirements:

1. **Semantics-appearance similarity.** The representation for the test clip *appearance* is similar to the representation of the selected *semantics*.

2. **Temporal coherence.** The selected semantics are ordered similar to their occurrences in the training set.

Drawing on the analogy to spatial correspondence estimation methods such as SIFT flow [34], the first requirement is a *data term* and the second is a *smoothness term*, albeit with two important distinctions: First, the data term matches test *appearances* to reference *semantics* directly, building on the joint embedding of semantics and appearances in the SVS. Second, we define the smoothness term in terms of associated semantics and not pixel coordinates.

4.1. Local Tessellation

Given the sequence of appearance representations $\mathcal{U} = (\mathbf{U}_1^A, ..., \mathbf{U}_M^A)$ for the test sequence, we seek a corresponding set of reference semantics $\mathcal{V} = (\mathbf{V}_{j_1}^S, ..., \mathbf{V}_{j_M}^S)$ (here, again, j indexes the N clips in the reference set). The local tessellation method employs only the semantics-appearance similarity. In other words, we associate each test clip $\mathbf{U}_i^A$, with the following training clip:

$$\mathcal{V}_{j_i}^* = \arg\min_{\mathcal{V}_j} ||\mathbf{U}_i^A - \mathbf{V}_j^S|| \tag{1}$$

4.2. Tessellation Distribution

We make the Markovian assumption that the semantics assigned to input clip i, only depend on the appearance of clip i and the semantics assigned to its preceding clip, $i-1$. This gives the standard factorization of the joint distribution for the clip appearances and their selected semantics:

$$\begin{aligned} P(\mathcal{V},\mathcal{U}) =& P(\mathbf{V}_{j_1}^S)P(\mathbf{U}_1^A|\mathbf{V}_{j_1}^S)\times \\ & \prod_{i=2}^{M} P(\mathbf{V}_{j_i}^S|\mathbf{V}_{j_{i-1}}^S)P(\mathbf{U}_i^A|\mathbf{V}_{j_i}^S). \end{aligned} \tag{2}$$

We set the priors $P(\mathbf{V}_{j_1}^S)$ to be the uniform distribution. Due to our mapping of both appearances and semantics to the joint SVS, we can define both posterior probabilities simply using the L2-norm of these representations:

$$P(\mathbf{U}_i^A|\mathbf{V}_j^S) \propto \exp\left(-||\mathbf{U}_i^A - \mathbf{V}_j^S||^2\right) \tag{3}$$

$$P(\mathbf{V}_{j_i}^S|\mathbf{V}_{j_{i-1}}^S) \propto \exp\left(-||\mathbf{V}_{j_i}^S - \mathbf{V}_{j_{i-1}}^S||^2\right) \tag{4}$$

Ostensibly, We can now apply the standard Viterbi method [42] to obtain a sequence $\mathcal{V}$ which maximizes this probability. In practice, we used a slightly modified version of this method, and, when possible, a novel method designed to better exploit our training data to predict database matches. These are explained below.

4.3. Restricted Viterbi Method.

Given the test clip appearance representations $\mathcal{U}$, the Viterbi method provides an assignment $\mathcal{V}^*$ such that,

$$\mathcal{V}^* = \arg\max_{\mathcal{V}} P(\mathcal{V},\mathcal{U}). \tag{5}$$

We found that in practice $P(\mathbf{U}_i^A|\mathbf{V}_j^S)$ is a long-tail distribution, with only a few dataset elements $\mathbf{V}_j^S$ near enough to any $\mathbf{U}_i^A$ for their probability to be more than near-zero. We, therefore, restrict the Viterbi method in two ways. First, we consider only the $r' = 5$ nearest neighboring database semantics features. Second, we apply a threshold on the probability of our data term, Eq. (3), and do not consider semantics $\mathbf{V}_j^S$ falling below this threshold, except for the first nearest neighbor. Therefore, the number of available assignments for each clip is $1 \leq r \leq 5$. This process is illustrated in Figure 2 (left).

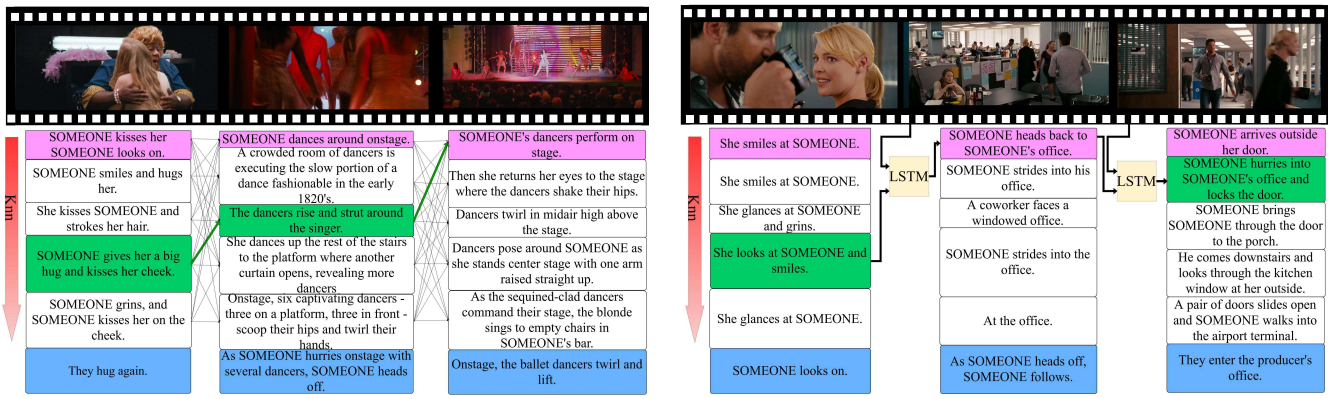

Figure 2. **Our two non-local tessellations**. **Left:** Tessellation by restricted Viterbi. For a query video (top), our method finds visually similar videos and selects the clips that preserve temporal coherence using the Viterbi Method. The ground truth captions are shown in blue, the closest caption is shown in pink. Note that our method does not always select clips with the closest captions but the ones that best preserve temporal coherence. **Right:** Tessellation by predicting the dynamics of semantics. Given a query video (top) and a previous clip selection, we use an LSTM to predict the most accurate semantics for the next clip.

Method	CIDEr-D	BLEU-4	BLEU-1	BLEU-2	BLEU-3	METEOR	ROUGE
BUPT CIST AI lab*	.072	.005	.151	.047	.013	**.075**	.152
IIT Kanpur*	.042	.004	.116	.003	.011	.070	.138
Aalto University*	.037	.002	.007	.001	.005	.033	.069
Shetty and Laaksonen [48]	.044	.003	.119	.024	.007	.046	.108
Yu et al [71]	.082	.007	.157	.049	**.017**	.070	.149
S2VT [62]	.088	.007	**.162**	**.051**	**.017**	.070	**.157**
Appearance Matching	.042	.003	.118	.026	.008	.046	.110
Local Tess. (mean pooling)	.091	.005	.134	.038	.013	.054	.125
Local Tessellation	.098	.007	.144	.042	.016	.056	.130
Unsupervised Tessellation	.102	.007	.146	.043	.016	.055	.137
Supervised Tessellation	**.109**	**.008**	.151	.044	**.017**	.057	.135

Table 1. *Video annotation results on the LSMDC'16 challenge [46].* CIDEr-D and BLEU-4 values were found to be the most correlated with human annotations in [45, 61]. Our results on these metrics far outperform others. * Denotes results which appear in the online challenge result board, but were never published. They are included here as reference.

4.4. Predicting the Dynamics of Semantics

The Viterbi method of Sec. 4.3 is efficient and requires only unsupervised training. Its use of the smoothness term of Eq. (3), however, results in potentially constant semantic assignments, where for any j_i, $\mathbf{V}^S_{j_i}$ can equal $\mathbf{V}^S_{j_{i-1}}$.

In cases where reference clips are abundant and come from continuous video sources, we provide a more effective method of ensuring smoothness. This is done by supervised learning of how the semantic labels associated with video clips change over time, and by using that to predict the assignment $\mathbf{V}^S_{j_i}$ for $\mathbf{U}^A_i$.

Our process is illustrated in Fig. 2 (right). We train an LSTM RNN [17] on the semantic and appearance representations of the training set video clips. We use this network as a function:

$$g(\mathbf{V}^S_0, \mathbf{V}^S_1, ..., \mathbf{V}^S_{i-1}, \mathbf{U}^A_1, ..., \mathbf{U}^A_{i-1}, \mathbf{U}^A_i) = \mathbf{V}^S_i, \quad \mathbf{V}^S_0 = \mathbf{0}, \tag{6}$$

which predicts the semantic representation $\mathbf{V}^S_i$ for the clip at time i given the semantic representation, $\mathbf{V}^S_{i-1}$, assigned to the preceding clip and the appearance of the current clip, $\mathbf{U}^A_i$. The labeled examples used to train g are taken from the training set, following the processing described in Sec. 3.2 and 3.3 in order to produce 2,000D post-CCA vectors. Each pair of previous ground truth semantics and current clip appearance in the training data provides one sample for training the LSTM. We employ two hidden layers, each with $1,000$ LSTM cells. The output, which predicts the semantics of the next clip, is also 2,000D.

Given a test video, we begin by processing it as in Sec. 4.3. In particular, for each of its clip representations $\mathbf{U}^A_i$, we select $r \leq 5$ nearest neighboring semantics from the training set. At each time step i, we feed the clip and its assigned semantics from the preceding clip at time $i-1$ to our LSTM predictor g. We thus obtain an estimate for the semantics we expect to see at time i, $\hat{\mathbf{V}}^S_i$.

Of course, the predicted vector $\hat{\mathbf{V}}^S_i$ cannot necessarily be interpreted as a semantic label: not all points in the SVS have semantic interpretations. We thus choose a rep-

GT: SOMEONE serves SOMEONE and SOMEONE.
ST: Now at a restaurant a waitress serves drinks.

GT: Then reaches in her bag and takes out a framed photo of a silver-haired woman.
ST: He spots a framed photo with SOMEONE in it.

GT: SOMEONE shifts his confused stare.
ST: He shifts his gaze then nods.

Figure 3. **Qualitative video captioning results**. Three caption assignments from the test set of the LSMDC16 benchmark. The Ground Truth captioning is provided along with the result of the Supervised Tessellation (ST) method.

resentation $\mathbf{V}^S_{j_i}$ out of the r selected for this clip, such that $||\hat{\mathbf{V}}^S_i - \mathbf{V}^S_{j_i}||^2$ is smallest.

5. Experiments

We apply our method to four separate video understanding tasks: video annotation, video summarization, temporal action detection, and sound prediction. Importantly, previous work was separately tailored to each of these tasks; we are unaware of any previously proposed single method which reported results on such a diverse range of video understanding problems. Contrary to the others, our method was applied to all of these tasks similarly.

5.1. Video Annotation

In our annotation experiments, we used the movie annotation benchmark defined by the 2016 Large Scale Movie Description and Understanding Challenge (LSMDC16) [46]. LSMDC16 presents a unified version of the recently published large-scale movie datasets, M-VAD [59] and MPII-MD [44]. The joint dataset contains 202 movies, divided to short (4-20 seconds) video clips with associated sentence descriptions.

Table 1 present annotation results. We focus primarily on the CIDEr-D [61] and the BLEU-4 [39] measures, since they are the only ones that are known to be well correlated with human perception [45, 61]. Other metrics are provided here for completeness. These measures are: BLEU1–3 [39], METEOR [7], and ROUGE-L [33]. We compare our method with several published and unpublished systems. The results include the following three variants of our pipeline.

Local tessellation. Our baseline system uses per-clip nearest neighbor matching in the SVS in order to choose reference semantics. We match each test clip with its closest semantics in the SVS. From Tab. 1, we see that this method already outperforms previous State-of-the-Art. As reference, we provide the performance of a similar method which matches clips in appearance space (*Appearance matching*). The substantial gap between the two underscores the importance of our semantics-appearance similarity matching. Instead of pooling using Fisher Vectors, we have also repeated the experiment with mean pooling both in the video and the text space. This results in local tesselation CIDEr score of .091. This is clear evidence for the power of even our simplest method to outperform the literature even with considerably weaker features.

Unsupervised tessellation. The graph-based method for considering temporal coherence, as presented in Sec. 4.3 is able to provide a slight improvement in results in comparison to the local method (Tab. 1).

Supervised tessellation. The model described in Sec. 4.4, with 2 layers of 1,000 LSTM units each. This method achieved the overall best performance on both CIDEr-D and BLEU-4, the metrics known to be most correlated with human perception [45, 61], outperforming previous state of the art with a gain of 23% on CIDEr-D. Qualitative results are provided in Fig. 3.

5.2. Video Summarization

Video summarization performance is evaluated on the SumMe [13] and TVSum [53] benchmarks. These benchmarks consist of 25 and 50 raw user videos, each depicting a certain event. The video frames are hand labeled with an importance score ranging from 0 (redundant) and 1 (vital) in SumMe and from 1 (redundant) and 5 (vital) in TVSum. The videos are about 1-5 minutes in length and the task is to produce a summary in the form of selected frames which is up-to 15% of the given video's length. Sample frames are shown in Fig. 4. The evaluation metric is the average f-measure of the predicted summary with the ground truth annotations. We follow [14, 75] in evaluating with multiple user annotations.

Similar to video annotation, our approach is to transfer the semantics (represented here by frame importance values) from the gallery to the tessellated video. Our method operates without incorporating additional computational steps, such as optimizing the selected set using the determinantal point process [29], commonly used for such applications [3, 11, 74].

Table 2 compares our performance with several recent reports on the same benchmarks. We again provide results

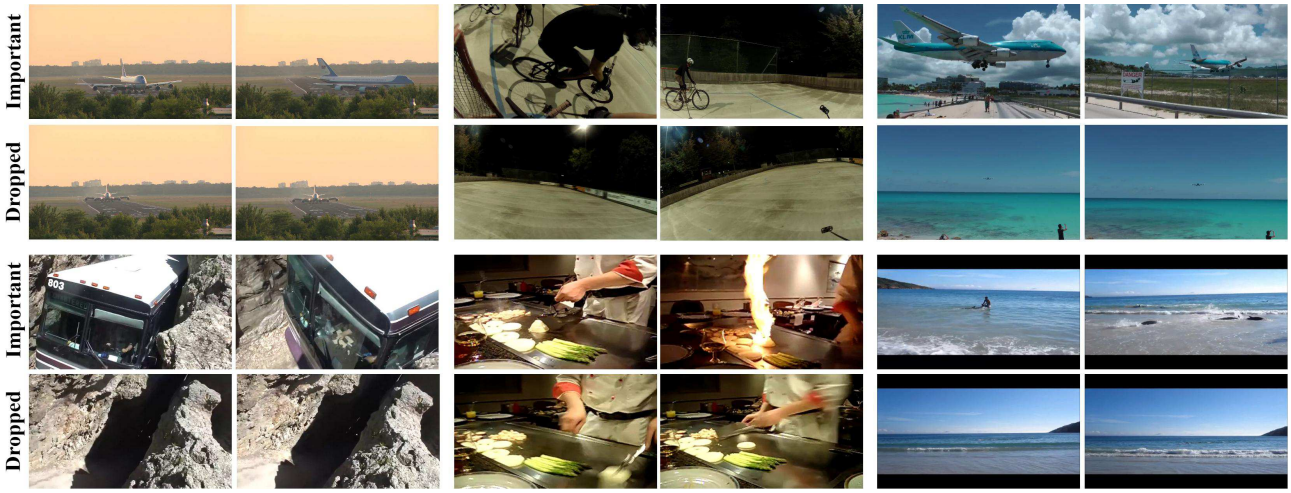

Figure 4. **Sample video summarization results**. Sample frames from six videos out of the SumMe benchmark. Each group of four frames contains two frames (top rows) from short segments that were deemed important by the unsupervised tessellation method and two (bottom rows) that were dropped out of our summaries.

Method	SumMe	TVSum
Khosla et al. [22] † ‡	–	36.0
Zhao et al. [76] † ‡	–	46.0
Song et al. [53] †	–	50.0
Gygli et. al [14]	39.7	–
Long Short-Term Memory [75]	39.8	54.7
Summary Transfer [74]	40.9	–
Local Tessellation	33.8	60.9
Unsupervised Tessellation	**41.4**	**64.1**
Supervised Tessellation	37.2	63.4

Table 2. *Video summarization results on the SumMe [13] and TVSum [53] benchmarks.* Shown are the average f-measures. Our unsupervised tessellation method outperforms previous methods by a substantial margin. † - unsupervised , ‡ - taken from [75]

for all three variants of our system. This time, the local and the supervised tessellation methods are both outperformed by previous work on SumMe but not on TVSum. Our unsupervised tessellation outperforms other tessellation methods as well as the state of the art on the summarization benchmarks by substantial margins.

We believe that unsupervised tessellation worked better than supervised because the available training examples were much fewer than required for the more powerful but data hungry LSTM. Specifically, for each benchmark we used only the labels from the same dataset, without leveraging other summarization datasets (e.g. [6]) for this purpose.

5.3. Temporal Action Detection

We evaluate our method on the task of action detection, using the THUMOS'14 [20] benchmark for this purpose.

This is one of the most recent and most challenging benchmarks released for this task. THUMOS'14 consists of a training set of 13,320 temporally trimmed videos taken from the UCF 101 dataset [54], a validation set of 1,010 temporally untrimmed videos with temporal action annotations, a background set with 2,500 videos which do not include of the 101 actions and finally a test set with 1,574 temporally untrimmed videos. In the temporal action detection benchmark, for every action class out of a subset of 20 actions, the task is to predict both the presence of the action in a given video and its temporal interval, i.e., the start and end times of its detected instances.

For each action, the detected intervals are compared against ground-truth intervals using the Intersection over Union (IoU) similarity measure. Denoting the predicted intervals by R_p and the ground truth intervals by R_{gt}, the IoU similarity is computed as $IoU = \frac{R_p \cap R_{gt}}{R_p \cup R_{gt}}$.

A predicted action interval is considered as true positive, if its IoT measure is above a predefined threshold and false positive otherwise. Ground truth annotations with no matching predictions are also counted as false positives. The Average Precision (AP) for each of the 20 classes is then computed and the mean Average Precision (mAP) serves as an overall performance measure. The process repeats for different IoT thresholds ranging from 0.1 to 0.5.

Our approach to detection is to tessellate a given untrimmed video with short clips from the UCF dataset [54]. With the resulting tessellation, we can determine which action occurred at each time in the video. Detection results on one sample video are shown in Fig. 5.

Tab. 3 lists the results of the three variants of our framework along with previous results presented on the benchmark. The tessellation methods outperforms the state of the art by a large margin, where the supervised tessellation achieves the best results among the three variants.

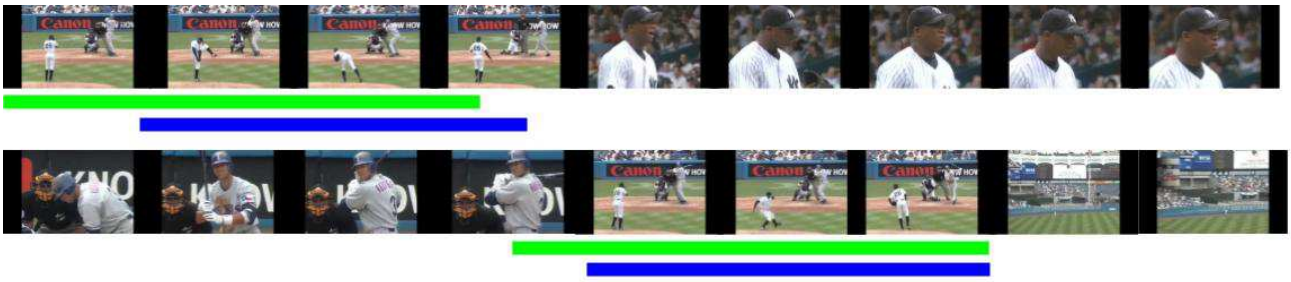

Figure 5. **Sample action detection results**. Detection results for the 'Baseball pitch' class. The predicted intervals by the supervised tessellation method are shown in green, the ground truth in blue.

Method	0.1	0.2	0.3	0.4	0.5
Wang et al. [66]	18.2	17.0	14.0	11.7	8.3
Oneata et al. [36]	36.6	33.6	27.0	20.8	14.4
Heilbron et al. [2]	–	–	–	–	13.5
Escorcia et al. [9]	–	–	–	–	13.9
Richard and Gall [43]	39.7	35.7	30.0	23.2	15.2
Shou et al. [49]	47.7	43.5	36.3	28.7	19.0
Yeung et al. [70]	48.9	44.0	36.0	26.4	17.1
Yuan et al. [72]	51.4	42.6	33.6	26.1	18.8
Local tessellation	56.4	51.2	43.8	32.5	20.7
Unsupervised t.	57.9	54.2	47.3	35.2	22.4
Supervised t.	**61.1**	**56.8**	**49.3**	**36.5**	**23.3**

Table 3. *Temporal Action Detection results on the THUMOS'14 [20] benchmark*. Shown are the mAP of the various methods for different IoT thresholds. Our proposed framework outperforms previous State-of-the-Art methods by a large margin. The supervised tessellation obtains the best results.

5.4. Predicting Sounds from Video

We test the capability of our method to predict sound from video using the Greatest Hits dataset [38]. This dataset consists of 977 videos of humans probing different environments with a drumstick: hitting, scratching, and poking different objects. Each video, on average, contains 48 actions and lasts 35 seconds. In [38], a CNN followed by an LSTM was used to predict sounds for each video. Following their protocol, we consider only the video segments centered on the audio amplitude peaks. We employ the published sound features that are available for 15 frame intervals around each audio peak, which we take to be our clip size. Each clip $\mathbf{C}$ is therefore associated with a visual representation, as presented in Sec. 3.1, and with a vector $\mathbf{a} \in \mathbb{R}^{1,890}$ concatenating the 15 sound features.

Matching is performed in a SVS that is constructed from the visual representation and the matching sound features. We predict sound features for *hit events* by applying tessellation and returning the selected sound feature vectors $\mathbf{a}$.

There are two criteria that are used for evaluating the results: Loudness and Centroid. In both cases both the MSE scores and correlations are reported. Loudness is taken to be the maximum energy (L2 norm) of the compressed subband envelopes over all timesteps. Centroid is measured by taking the center of mass of the frequency channels for a one-frame window around the center of the impact.

Method	Loudness		Centroid	
	Err	r	Err	r
Full system of [38]	**0.21**	**0.44**	**3.85**	0.47
Appearance matching	0.35	0.18	6.09	0.36
Local tessellation	0.27	0.32	4.83	0.47
Unsupervised tessellation	0.26	0.33	4.76	**0.48**
Supervised tessellation	0.24	0.35	4.44	0.46

Table 4. *Greatest Hits benchmark results*. Shown are the MSE and the correlation coefficient for two different success criteria.

Our results are reported in Tab. 4. The importance of the semantic space as can be observed from the gap between the appearance only matching to the Local Tessellation method. Leveraging our supervised and unsupervised tessellation methods improves the results even further. In three out of four criteria the supervised tessellation seems preferable to the unsupervised one in this benchmark.

6. Conclusions

We present a general approach to understanding and analyzing videos. Our design transfers per-clip video semantics from reference, training videos to novel test videos. Three alternative methods are proposed for this transfer: local tessellation, which uses no context, unsupervised tessellation which uses dynamic programming to apply temporal, semantic coherency, and supervised tessellation which employs LSTM to predict future semantics. We show that those methods, coupled with a recent video representation technique, provide state of the art results on three very different video analysis domains: video annotation, video summarization, and action detection and near state of the art on a fourth application, sound prediction from video. Our method is unique in being first to obtain state of the art results on such different video understanding tasks, outperforming methods tailored for these applications.

Acknowledgments

This research is supported by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI).

References

[1] H. Aradhye, G. Toderici, and J. Yagnik. Video2text: Learning to annotate video content. In *Int. Conf. on Data Mining Workshops*, pages 144–151. IEEE, 2009.

[2] F. Caba Heilbron, J. Carlos Niebles, and B. Ghanem. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1914–1923, 2016.

[3] W.-L. Chao, B. Gong, K. Grauman, and F. Sha. Large-margin determinantal point processes. UAI, 2015.

[4] D. L. Chen and W. B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proc. Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200. Association for Computational Linguistics, 2011.

[5] W.-S. Chu, Y. Song, and A. Jaimes. Video co-summarization: Video summarization by visual co-occurrence. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 3584–3592, 2015.

[6] S. E. F. De Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011.

[7] M. Denkowski and A. Lavie. Meteor universal: Language specific translation evaluation for any target language. In *In Proceedings of the Ninth Workshop on Statistical Machine Translation*. Citeseer, 2014.

[8] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 2625–2634, 2015.

[9] V. Escorcia, F. C. Heilbron, J. C. Niebles, and B. Ghanem. Daps: Deep action proposals for action understanding. In *European Conf. Comput. Vision*, pages 768–784. Springer, 2016.

[10] B. G. Fabian Caba Heilbron, Victor Escorcia and J. C. Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 961–970, 2015.

[11] B. Gong, W.-L. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. In *Neural Inform. Process. Syst.*, pages 2069–2077, 2014.

[12] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Proc. Int. Conf. Comput. Vision*, pages 2712–2719, 2013.

[13] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool. Creating summaries from user videos. In *European Conf. Comput. Vision*, pages 505–520. Springer, 2014.

[14] M. Gygli, H. Grabner, and L. Van Gool. Video summarization by learning submodular mixtures of objectives. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 3090–3098, 2015.

[15] T. Hassner. A critical review of action recognition benchmarks. In *Proc. Conf. Comput. Vision Pattern Recognition Workshops*, pages 245–250, 2013.

[16] T. Hassner and C. Liu. *Dense Image Correspondences for Computer Vision*. Springer, 2015.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[18] H. Huang, Y. Lu, F. Zhang, and S. Sun. A multi-modal clustering method for web videos. In *Int. Conf. on Trustworthy Computing and Services*, pages 163–169. Springer, 2012.

[19] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *Trans. Pattern Anal. Mach. Intell.*, 35(1):221–231, 2013.

[20] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`, 2014.

[21] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 1725–1732, 2014.

[22] A. Khosla, R. Hamid, C.-J. Lin, and N. Sundaresan. Large-scale video summarization using web-image priors. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 2698–2705, 2013.

[23] B. Klein, G. Lev, G. Sadeh, and L. Wolf. Associating neural word embeddings with deep image representations using fisher vectors. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 4437–4446, 2015.

[24] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion interchange patterns for action recognition in unconstrained videos. In *European Conf. Comput. Vision*, pages 256–269. Springer, 2012.

[25] O. Kliper-Gross, T. Hassner, and L. Wolf. One shot similarity metric learning for action recognition. *Similarity-based pattern recognition*, 2011.

[26] O. Kliper-Gross, T. Hassner, and L. Wolf. The action similarity labeling challenge. *Trans. Pattern Anal. Mach. Intell.*, 34(3):615–621, 2012.

[27] N. Krishnamoorthy, G. Malkarnenkar, R. J. Mooney, K. Saenko, and S. Guadarrama. Generating natural-language video descriptions using text-mined knowledge. In *AAAI Conf. on Artificial Intelligence*, volume 1, page 2, 2013.

[28] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *Proc. Int. Conf. Comput. Vision*, pages 2556–2563. IEEE, 2011.

[29] A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *arXiv preprint arXiv:1207.6083*, 2012.

[30] I. Laptev. On space-time interest points. 64(2-3):107–123, 2005.

[31] G. Lev, G. Sadeh, B. Klein, and L. Wolf. RNN fisher vectors for action recognition and image annotation. *arXiv preprint arXiv:1512.03958*, 2015.

[32] Y. Li, W. Li, V. Mahadevan, and N. Vasconcelos. Vlad3: Encoding dynamics of deep features for action recognition. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 1951–1960, 2016.

[33] C.-Y. Lin and F. J. Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proc. Annual Meeting on Association for Computational Linguistics*, page 605. Association for Computational Linguistics, 2004.

[34] C. Liu, J. Yuen, and A. Torralba. SIFT flow: Dense correspondence across scenes and its applications. *Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, 2011.

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

[36] D. Oneata, J. Verbeek, and C. Schmid. The lear submission at thumos 2014. 2014.

[37] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, B. Shaw, A. F. Smeaton, and G. Quénot. TRECVID 2012–an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID*, 2012.

[38] A. Owens, P. Isola, J. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman. Visually indicated sounds. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 2405–2413, 2016.

[39] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proc. annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[40] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *European Conf. Comput. Vision*, pages 143–156. Springer, 2010.

[41] D. Potapov, M. Douze, Z. Harchaoui, and C. Schmid. Category-specific video summarization. In *European Conf. Comput. Vision*, pages 540–555. Springer, 2014.

[42] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[43] A. Richard and J. Gall. Temporal action detection using a statistical language model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3131–3140, 2016.

[44] A. Rohrbach, M. Rohrbach, N. Tandon, and B. Schiele. A dataset for movie description. In *Proc. Conf. Comput. Vision Pattern Recognition*, 2015.

[45] A. Rohrbach, A. Torabi, T. Maharaj, M. Rohrbach, C. Pal, A. Courville, and B. Schiele. The large scale movie description and understanding challenge (LSMDC 2016), howpublished = Available: `http://tinyurl.com/zabh4et`, month = September, year = 2016.

[46] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, P. Chris, L. Hugo, C. Aaron, and B. Schiele. Movie description. *arXiv preprint*, 2016.

[47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115(3):211–252, 2015.

[48] R. Shetty and J. Laaksonen. Video captioning with recurrent networks based on frame-and video-level features and visual content classification. *arXiv preprint arXiv:1512.02949*, 2015.

[49] Z. Shou, D. Wang, and S.-F. Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016.

[50] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep fisher networks for large-scale image classification. In *Neural Inform. Process. Syst.*, pages 163–171, 2013.

[51] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Neural Inform. Process. Syst.*, pages 568–576, 2014.

[52] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[53] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes. Tvsum: Summarizing web videos using titles. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 5179–5187, 2015.

[54] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[55] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using LSTMs. In *Int. Conf. Mach. Learning*, volume 2, 2015.

[56] V. Sydorov, M. Sakurada, and C. H. Lampert. Deep fisher kernels-end to end learning of the fisher kernel GMM parameters. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 1402–1409, 2014.

[57] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *European Conf. Comput. Vision*, pages 140–153. Springer, 2010.

[58] J. Thomason, S. Venugopalan, S. Guadarrama, K. Saenko, and R. J. Mooney. Integrating language and vision to generate natural language descriptions of videos in the wild. In *COLING*, volume 2, page 9, 2014.

[59] A. Torabi, C. Pal, H. Larochelle, and A. Courville. Using descriptive video services to create a large data source for video annotation research. *arXiv preprint*, 2015.

[60] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proc. Int. Conf. Comput. Vision*, pages 4489–4497, 2015.

[61] R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 4566–4575, 2015.

[62] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 4534–4542, 2015.

[63] H. D. Vinod. Canonical ridge and econometrics of joint production. *Journal of Econometrics*, 4(2):147–166, May 1976.

[64] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 3156–3164, 2015.

[65] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proc. Int. Conf. Comput. Vision*, pages 3551–3558, 2013.

[66] L. Wang, Y. Qiao, and X. Tang. Action recognition and detection by combining motion and appearance features. *THUMOS14 Action Recognition Challenge*, 1:2, 2014.

[67] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 4305–4314, 2015.

[68] S. Wei, Y. Zhao, Z. Zhu, and N. Liu. Multimodal fusion for video search reranking. *Trans. on Knowledge and Data Engineering*, 22(8):1191–1199, 2010.

[69] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *Proc. Int. Conf. Comput. Vision*, pages 4507–4515, 2015.

[70] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2016.

[71] Y. Yu, H. Ko, J. Choi, and G. Kim. Video captioning and retrieval models with semantic attention. *arXiv preprint arXiv:1610.02947*, 2016.

[72] J. Yuan, B. Ni, X. Yang, and A. A. Kassim. Temporal action localization with pyramid of score distribution features. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 3093–3102, 2016.

[73] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern recognition*, 30(4):643–658, 1997.

[74] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman. Summary transfer: Exemplar-based subset selection for video summarizatio. In *Proc. Conf. Comput. Vision Pattern Recognition*, 2016.

[75] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman. Video summarization with long short-term memory. In *European Conf. Comput. Vision*, 2016.

[76] B. Zhao and E. P. Xing. Quasi real-time summarization for consumer videos. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 2513–2520, 2014.