Joint Visual-Temporal Embedding for Unsupervised Learning of Actions in Untrimmed Sequences

Rosaura G. VidalMata[1], Walter J. Scheirer[1], Anna Kukleva[2], David Cox[3], and Hilde Kuehne[3]
[1] University of Notre Dame, IN, USA [2] Max-Planck-Institute for Informatics, Saarbrucken, Germany [3] MIT-IBM Waston AI Lab, MA, USA
{rvidalma, walter.scheirer}@nd.edu, akukleva@mpi-inf.mpg.de, {david.d.cox, kuehne}@ibm.com

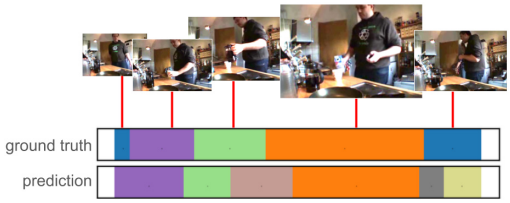
Abstract

Understanding the structure of complex activities in untrimmed videos is a challenging task in the area of action recognition. One problem here is that this task usually requires a large amount of hand-annotated minute- or even hour-long video data, but annotating such data is very time consuming and can not easily be automated or scaled. To address this problem, this paper proposes an approach for the unsupervised learning of actions in untrimmed video sequences based on a joint visual-temporal embedding space. To this end, we combine a visual embedding based on a predictive U-Net architecture with a temporal continuous function. The resulting representation space allows detecting relevant action clusters based on their visual as well as their temporal appearance. The proposed method is evaluated on three standard benchmark datasets, Breakfast Actions, INRIA YouTube Instructional Videos, and 50 Salads. We show that the proposed approach is able to provide a meaningful visual and temporal embedding out of the visual cues present in contiguous video frames and is suitable for the task of unsupervised temporal segmentation of actions.

1. Introduction

Research shows that humans usually understand complex activities through the ongoing temporal segmentation of perceived inputs into meaningful segments [39]. Nevertheless, replicating this behavior in fully automated systems is a challenging problem as it requires identifying the meaningful steps in a given task and how do they logically relate to each other. Fully supervised systems have been proposed to address this, but they rely on large amounts of training data. Manually annotating this data is especially expensive for temporal video segmentation as this task usually requires a dense frame-based annotation. Weakly supervised approaches attempt to alleviate this by incorporating the use of additional sources of human-generated information such as speech or video captions [2, 24]. However, differences between the alignment of additional modalities like audio, subtitles, or descriptive meta-data to the video frames might prove challenging for some of these approaches [34].

(a) Breakfast Actions (BF)



(b) INRIA YouTube Instructional Videos (YTI)

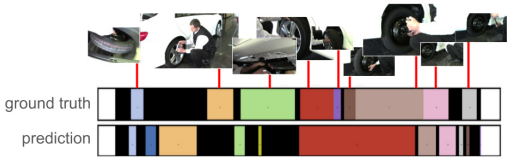


Figure 1: Temporal segmentation of two videos from the Breakfast Actions [13] and INRIA YouTube Instructional Videos (YTI) [2] Datasets. The black segments in the YTI video correspond to background frames whose content is not associated with a sub-activity relevant to the video task. Our approach maintains the logical ordering of sub-activities and a good estimate on their start and duration.

Therefore, other methods have been proposed to tackle the challenge of training such models without dense human-generated labels, spanning from learning with weak or sparse annotation [10, 15, 19] to completely unsupervised learning of temporal action segmentation [4, 33, 16, 1]. The work described in this paper deals with the latter problem, the unsupervised learning of action segments from unlabeled video data, which can be framed as the task of unsupervised temporal action segmentation. This follows the idea that, given a set of videos all capturing the same activ-

ity, it should be possible to identify temporal segments with similar sub-actions across all videos.

Previous work [4, 33, 16] has shown that temporal appearance can play a critical role in this task. For instance, in videos showing how to make pancakes the process of cracking eggs should not only look visually similar across different videos but would generally occur before other tasks such as mixing the eggs into the batter or pouring the batter onto the griddle. So, it can be assumed that actions, at least in task-oriented videos, not only share certain visual features but also occur in a similar temporal space. Recent approaches usually incorporate this property by learning a strong temporal regularization [33, 16], which helps to find clusters over time but may result in a lower ability to identify segments based on their visual representation.

To address this problem, this paper proposes a joint visual-temporal learning pipeline that combines the advantages of current temporal embedding systems with a visual embedding based on a combination of predictive visual and temporal learning tasks. To this end, we combine a state-of-the-art temporal embedding system [16], designed to estimate the relative timestamp of a given video frame, with a visual encoder-decoder pipeline that is trained on a combination of visual and temporal losses.

The intuition behind this is that the embedding should not only reconstruct the plain input signal, but it should also find a reconstruction that allows for a better estimation of the respective timestamp and, thus, a better temporal reconstruction. To prevent the learning of weak visual cues, we shift the output of the visual encoder by a number of frames, which turns it into a visual prediction architecture, similar to other self-supervised models [25, 7]. Combined with the temporal loss, the model predicts a frame-representation that is optimized to give the best timestamp prediction in the temporal embedding framework. The resulting embedding space thus captures visual and temporal representations of each individual frame.

We evaluate the proposed system on three challenging standard benchmark datasets, the Breakfast dataset [13], the INRIA YouTube Instructional Videos (YTI) dataset [4], and the 50 Salads dataset [36], achieving good temporal segmentation compared to state-of-the-art approaches. Figure 1 shows qualitative examples which demonstrate that the proposed architecture is able to adapt well to a diverse set of action tasks.

2. Related Work

2.1. Unsupervised learning of temporal sequences

While there has been plenty of work done in the area of action segmentation in video, a vast majority of the approaches are fully supervised methods relying on frame annotations [10, 8, 14, 30], or weakly supervised approaches that use some form of metadata [19, 5, 31]. Supervised models achieve high-quality temporal segmentations but their training is heavily dependent on vast amounts of good quality hand-annotated minute-or even hour-long video data which can become prohibitive in most real-life scenarios as this process can be time-consuming and can not easily be automated or scaled.

To overcome the dependence on labeled data, unsupervised methods have seen increased interest recently. Chen *et al*. [6] propose a domain adaptation approach based on self-supervised auxiliary tasks (SSTDA) that is able to obtain state-of-the-art performance with a reduced amount of labeled data (they show that even when using 65% of the labeled data their method still produces state-of-the-art results). One of the first unsupervised methods for action segmentation is the one proposed by Bojanovski *et al*. [4] based on a Frank-Wolfe optimization algorithm. Sener *et al*. [33] later proposed the modeling of the temporal structure of sub-activities using a combination of Generalized Mallows Model (GMM) sampling and the estimation of the action length calculated using the frame distribution to estimate the action segmentation in complex action videos. Following a similar segmentation pipeline, Kukleva *et al*. [16] proposed instead a combination of temporal encoding generated using a frame timestamp prediction network and Viterbi decoding for consistent frame-to-cluster assignment. Another interesting take on the problem is further proposed by Aakar *et al*. [1]. Different from the methods so far, the idea here is not to cluster the feature space to identify actions, but to detect action boundaries in an unsupervised way by learning a predictive framework which uses the difference between observed and predicted frame features as a means to determine event boundaries. As such the approach is focusing on a per video-based segmentation instead of the identification of action classes across multiple videos.

2.2. Unsupervised and self-supervised learning of visual representations

Various approaches have been proposed for the learning of visual representations without labels [12]. Particularly, in the case of learning video representations in an unsupervised way, various approaches make use of temporal properties of the data *e.g*., in the form of shuffling [27], or similar to the idea proposed in this paper, temporal prediction [35, 7]. Temporal prediction has been established as a way to achieve a deeper understanding of the data as it requires an implicit understanding of the structure of the observed visual features and the rules they follow while they change over time [35, 23, 38]. However, it has been pointed out that the use of traditional losses does not translate well for video frame prediction. Srivastava *et al*. [35] observed that predictive models trained solely with an MSE loss have a tendency of blurring regions with uncertainty.

This has given way to the introduction of promising al-

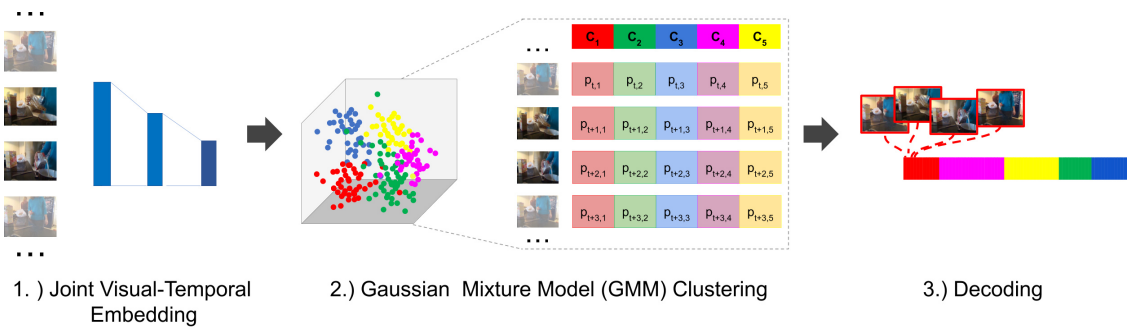


Figure 2: Joint visual-temporal training pipeline. If the feature embedding has a good representation of the visual and temporal attributes of each frame, the frames that cluster together will have similar temporal locations and share visual attributes that represent a given sub-activity.

ternatives such as the adversarial loss present in Generative Adversarial Networks (GANs) [9] and Conditional GANs [26], which have led to significant advances in the performance of video prediction [7, 22, 25]. GAN-based methods have incorporated the use of U-Net architectures [32] into their generator module given their good performance in image-to-image translation [11, 40] and image segmentation [32, 3, 21] tasks. Our proposed approach does not focus on learning feature representations as is the case in those methods but instead borrows other ideas from the field to improve the joint visual-temporal learning of an embedding space for unsupervised actions recognition.

3. System Description

Given a collection of complex activity videos $D_A = \{v_i\}_{i=1}^{V}$ with V videos belonging all to the same activity class A, we want to learn k action classes C_k that appear in this activity, as well as the action class label of all the frames (N_{v_i}) of each video $v_i = \{f_n\}_{n=1}^{N_{v_i}}$. Note that the frame input used here is a pre-extracted one-dimensional feature vector, that can be derived from hand-crafted feature representations or from a network that has been trained in an unsupervised way.

To learn an embedding space for this input, we propose a two-stage pipeline (see Figure 3). The first stage (3a) consists of the independent training of two models: one encoding a visual embedding and the second encoding a temporal embedding only. We pre-train both models separately and then join them in the second stage (3b), where the temporal model becomes a discriminator for the frames generated by the visual model. We update the components of this model in an alternate fashion, keeping the visual model constant during the training phase of the temporal model in order to allow it to identify the loss of valuable temporal information from the reconstructed frame. Similarly, we then keep the temporal model constant while the visual model is training to improve the results of its outputs when they are evaluated by the temporal discriminator.

This joint training allows our system to learn an embedding space that not only captures visual properties by predicting the next frame but also ensures that we retain enough temporal information to estimate a consistent timestamp for the output. The embedding space of the model is then used for clustering to form the respective classes and segment the videos accordingly. We illustrate our clustering and segmentation pipeline during testing in Figure 2. We discuss each step of the pipeline in the following sections.

3.1. Stage 1: Disjoint training of the Visual and Temporal Models

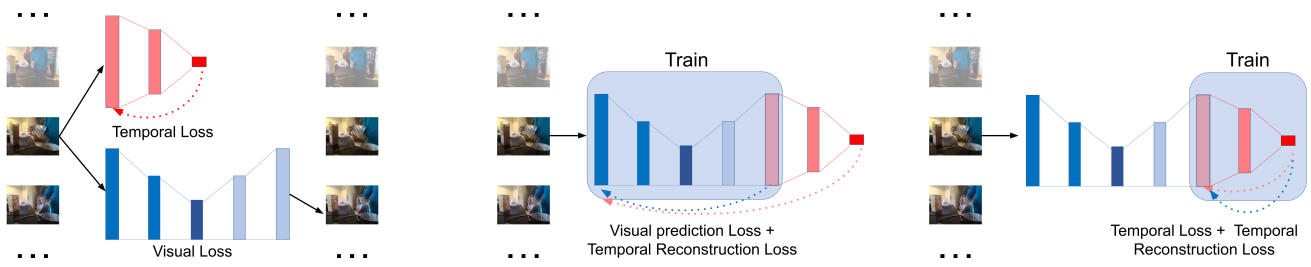
3.1.1 Visual Embedding Model

The visual embedding model is designed to learn a low dimensional embedding of the frame-based input vector. For this, we make use of an encoder-decoder structure to model a frame prediction framework which, using as a prior the visual features of the frame at time t (frame f_t) learns to predict the features at a future time $t + s$ (f_{t+s}).

As is shown in Figure 3a, our visual model (displayed in blue) consists of three down-sampling layers followed by three up-sampling layers with skip connections between the down-sampling and up-sampling layers to preserve fine-grained details from being discarded by the encoding process. As such, we are able to suppress visual noise and find visual cues that are more relevant to the respective task while ensuring that the encoded embedding x_t maintains enough relevant information in order to reconstruct a viable future frame in the video sequence. This linear U-Net is then used during testing to generate an abstract representation of the frame's features x_t (generated by the last down-sampling layer), which is then used as the embedding of the frame f_t during the later clustering process (see Figure 2).

This visual embedding model learns to generate accurate future frame predictions by minimizing the mean squared difference between the predicted and the real frames:

$$Loss_{vis} = \frac{1}{N-s} \sum_{t=1}^{N-s} (f_{t+s} - \hat{f}_{t+s})^2 \qquad (1)$$



(a) Stage 1 (b) Stage 2

Figure 3: Two-stage visual-embedding pipeline: 3a) Visual+Temporal embedding: next frame prediction U-Net to generate a visual-temporal embedding (output of the last down-sampling layer, denoted by a red arrow), 3b) Temporal discriminator: timestamp predictor MLP used to identify the loss of temporal information in the frames predicted by stage 1.

For ease of notation, we use N as the number of all frames of all videos V belonging to the same activity class A, and s as the prediction time step. That is, for $s = 5$ given an input frame captured at timestamp $t = 0$ ($f_{t=0}$) the network will return a predicted frame s frames into the future ($\hat{f}_{t=5}$). For our experiments we used $s = 5$ as the prediction goal of our visual embedding model. However, we study the impact of different step sizes in Section 5.1.

3.1.2 Temporal Embedding Model

The temporal embedding model (displayed in red in Figure 3a) is designed to extract valuable temporal information out of a given frame. For this we implemented a three-layer Multilayer Perceptron (MLP) with the learning goal of predicting the relative timestamp t of a given frame (where $t = frameIdx/N$). The MLP receives as input a video frame f_t captured at a timestamp t and provides a prediction for the relative timestamp of the frame $T(f_t)$. As such, to ensure the MLP has a working knowledge on the temporal structure of the data, it must minimize the difference between the timestamp prediction $T(f_n)$ it provides for a given f_t frame, and the actual timestamp t of such frame:

$$Loss_{temp} = \frac{1}{N}\sum_{t=1}^{N}(T(f_t) - t)^2 \quad (2)$$

3.2. Stage 2: Joint visual-temporal training

In this stage we joined our visual and temporal models in a single framework, as such, our final architecture is composed of (1) a frame predicting U-Net that is able to retain enough visual and temporal features in its predictions, and (2) a timestamp predictor, trained to recognize any discrepancy between the temporal quality of the frame predicted by the U-Net $\hat{f}_{t+s}$ and the observed frame f_{t+s}.

3.2.1 Joint Visual-Temporal Embedding Model

The learning objective of the U-Net during stage 1 was to predict a frame that seemed visually plausible (Equation 1), this helps to ensure that the valuable visual cues in the frame embedding are retained. In the second stage of our training, we build on this objective to ensure that the temporal cues of the frame are also preserved. For this we use our temporal model: the timestamp predictor evaluates the quality of the temporal features encoded into our embedding by measuring the difference between the predicted timestamp of the U-Net's output $T(\hat{f}_{t+s})$ and the predicted timestamp of the observed frame $(T(f_{t+s})$, allowing us to measure the temporal reconstruction loss of our generated frames:

$$Loss_{trec} = \frac{1}{N-s}\sum_{t=1}^{N-s}(T(f_{t+s}) - T(\hat{f}_{t+s}))^2 \quad (3)$$

We incorporate this temporal reconstruction loss into the U-Net's learning objective:

$$Loss_{joint} = Loss_{vis} + Loss_{trec} \quad (4)$$

This joint loss (Equation 4) ensures that our embedding model maintains a good balance of visual and temporal cues into its embedding. Its learning objective is now minimizing both the visual and temporal reconstruction losses.

3.2.2 Temporal Discriminator

The goal of our temporal embedding model is now to serve as a discriminator for the reconstructed frames generated by our Joint Visual-Temporal Embedding Model. For this, our temporal discriminator must be able to identify the loss of the "temporal quality" between the frame predicted by the U-Net $\hat{f}_{t+s}$ and the ground-truth frame f_{t+s}. Given a frame f_t we consider that the U-Net's embedding x_t has led to the loss of temporal cues if the predicted timestamp $T(\hat{f}_{t+s})$ (estimated by the temporal embedding model) of the U-Net's output $\hat{f}_{t+s}$ is not similar to the timestamp of the ground-truth frame f_{t+s}. The temporal quality ($TQual$) of the U-Net embedding can then be measured as follows:

$$TQual = 1 - \left(\frac{1}{N-s}\sum_{t=1}^{N-s}T(\hat{f}_{t+s}) - (t+s))^2\right) \quad (5)$$

We can then use the estimation of the temporal quality loss as a means for the MLP to discriminate between a real video frame f_{t+s} and a low-quality estimation $\hat{f}_{t+s}$. As a result, the timestamp predictor can learn to discriminate between the temporal differences of the U-Net's prediction and the ground-truth. The loss of the MLP temporal discriminator is then evaluated as follows:

$$Loss_{MLP} = \frac{\frac{1}{N}\sum_{t=1}^{N}(T(f_t) - (t))^2}{TQual} \quad (6)$$

In the above formulation, $T(f_i)$ represents the output of the timestamp predictor given an input frame f_i.

3.3. Clustering and decoding

For all further processing, we follow the protocol of [16, 33, 4]. Once we have the temporally enhanced embedding, we cluster the embedded features of all videos into k clusters. For this, we employ a single Gaussian Mixture Model to model all the embedded features of a given task. To fit them into k clusters, we then use the logarithm of the probability of each frame belonging to each of the clusters. As such, for every video frame n, we have a corresponding k dimensional vector with each of the frame's scores for each of the clusters (see step 2 in Figure 2).

A naive approach for segmentation would be using the scores obtained by the clustering method in order to assign a sub-activity to each video frame. This, however, generates a non-homogeneous segmentation with few groups of continuous frames being assigned to the same sub-activity cluster. We use a modified version of the frame labeling method presented in [16], Viterbi decoding with length model as proposed by [15] to alleviate this effect. We evaluate the probability of each frame n belonging to a cluster c_x with respect to the probabilities of the neighboring frames and seek to maximize the probability of the sequence following a fixed cluster order. The cluster order is determined by the mean time-stamp of each cluster.

A fixed cluster ordering $c_1, c_2, ..., c_i, c_j, ..., c_k$, would constrain the possible clusters a frame f_t can be assigned to. Frame f_t could belong to either the same cluster c_i as it preceding sampled frame $f_{t-\gamma}$ -where γ is the frame sampling size (in the method introduced by [16] $\gamma = 1$) or to the next cluster c_j in the predetermined ordering.

4. Experimental Evaluation

4.1. Datasets

We evaluate our method using three datasets: Breakfast Actions dataset (BF) [13], INRIA YouTube Instructional Videos (YTI) [2], and 50 Salads (50S) [36]. We are using the reduced Fisher Vector features as proposed by [14] and used by [16, 33, 28] for the evaluation of the three datasets.

The Breakfast Actions (BF) dataset contains 70 hours of cooking activities of varying complexity. It contains 10 different cooking tasks (with about 170 videos per task), which can be further split into 48 sub-activities. The length of each video is highly dependent on the type of task, ranging from 30 seconds to a few minutes. The videos are recorded in different real-life environments with 52 people performing each of the 10 different actions. They have a fixed viewpoint through all activities for each person, which leads to a high intra-class and low inter-class variance.

The INRIA YouTube Instructional Videos (YTI) dataset contains five tasks of different instructional domains: "making coffee", "changing a car tire", "CPR", "jumping a car", and "potting a plant" (with about 30 videos per task), which can be divided into 47 sub-activities. As opposed to Breakfast Actions, the videos might have been edited and include shot boundaries, as well as different and changing viewpoints or zooming. The videos in this dataset are in average longer than the videos in Breakfast Actions, however, there is a significant presence of background frames whereas the Breakfast Action actions are densely labeled without intermediate background classes.

The 50 Salads (50S) dataset contains over 4.5 hours of video captures of different actors preparing 2 kinds of mixed salads (with 25 videos for each type of salad). While similar in nature to the data present in the Breakfast dataset, the videos in 50 Salads tend to be longer and contain more sub-activities. These videos have an average length of 10k frames and contain complex interactions between hands, utensils, and ingredients. In addition to this, different actions in this dataset often contain similar motions, differing only on the ingredient or utensil used and can be further classified into three stages: preparing the dressing, cutting and mixing ingredients, and serving. As such we have similar actions like "adding oil" or "adding vinegar" and "cutting tomato" or "cutting cucumber" which follow similar hand motions (but differ in the interacting objects) and can happen at interchangeable times.

4.2. Training

Following the two-stage training protocol described in Section 3, we first train the visual and temporal models independent from each other. The visual embedding model is trained for 160 (for the BF Dataset) and 140 epochs (for the YTI and 50S datasets) while the temporal embedding model is trained for 10, 20, and 30 epochs (for 50S, YTI, and BF respectively).

During our second training stage, we alternate the training of the visual and temporal models: we train the visual model for 40 consecutive epochs (in which the temporal model is frozen) and then train the temporal discriminator for 5 epochs (where the visual embedding model remains constant. We run these alternating periods for 60 (for the BF dataset) and 120 epochs (for the YTI and 50S datasets).

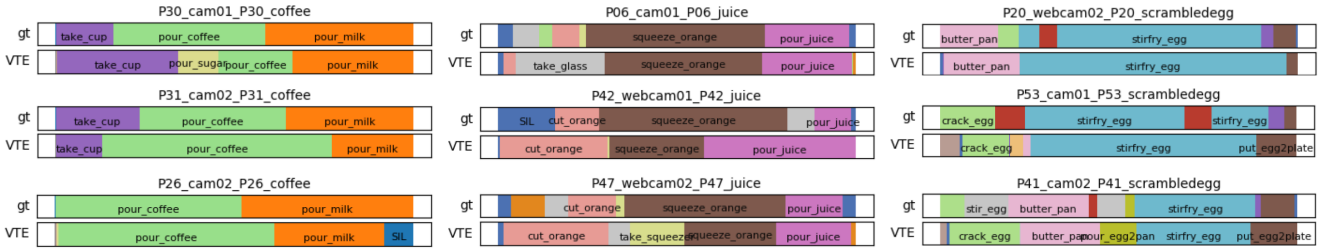


Figure 4: Segmentation examples from our approach on videos from the Breakfast Actions dataset. We omit some of the ground-truth labels in the visualizations to facilitate reading the labels of the more prominent segments.

4.3. Mapping

To evaluate the segmentation returned by our unsupervised approach, we need to map the segmented clusters of sub-activities to the ground-truth sub-activities related to each specific task. For this, we use Hungarian matching to provide a one-to-one mapping that maximizes the similarity between the segmented clusters and the ground-truth sub-activities. We follow the protocol of [4, 33, 16] and compute the Hungarian matching over all the videos of one activity. Note that this is different from the Hungarian matching for every single video as used by [1], which optimizes the matching for every single video and usually leads to higher accuracy, but also allows clusters to change their label from one video to another.

We evaluate the accuracy of our method using two common evaluation metrics used in [33, 16, 1], 1) *Mean Over Frames (MoF)* to indicate the percentage of frames in the segmentation that were correctly labeled over all the frames of videos assigned to a given task for the Breakfast Actions and 50 Salads Datasets, and 2) *F1* score for the INRIA YouTube Instructional Videos (YTI) Dataset to compare with other state-of-the-art approaches.

5. Results and Analysis

We compare the performance of our two-stage visual embedding pipeline to state-of-the-art approaches (including weakly-supervised and fully supervised setups) on three challenging action segmentation datasets. As can be seen in Figure 4 our approach is able to maintain the logical ordering of sub-activities and a good estimate on their start and duration in actions with different levels of complexity.

Breakfast Actions Dataset: Our method is able to outperform the state-of-the-art on the unsupervised learning methods applied to this dataset (see Table 1) and performs competitively against weakly-supervised approaches. It is important to note that our cluster-to ground-truth mapping and evaluation is done in a global manner, that is, we use all the predicted labels and ground-truth for all the videos of a given task to do the Hungarian matching, and then evaluate calculating the MoF using the count of all the true predictions on the whole dataset. On the other hand, approaches such as LSTM+AL [1] employ a per-video (local) cluster to ground-truth mapping, which might account for the difference in the performance of the two approaches. For comparison purposes, when applying a video-based Hungarian matching our method obtains a MoF of 52.25%.

Supervision	Approach	MoF
Full	TCFPN [8]	52.0%
	HTK [14]	56.3%
	GRU [30]	60.6%
	MS-TCNN++ [20]	67.6%
	Local SSTDA [6]	70.2%
	SSTDA[18]	**70.3%**
Weak	Fine2Coarse [29]	33.3%
	GRU [30]	36.7%
	TCFPN+ISBA [8]	38.4%
	NN-Viterbi [31]	43%
	D3TW [5]	45.7%
	CDFL [19]	**50.2%**
Unsupervised	GMM [33]	34.6%
	CTE-MLP [16]	41.8%
	(LSTM+AL [1])	(42.9%*)
	Ours	**48.08%**

Table 1: Segmentation results on the Breakfast Action dataset compared to supervised, weakly supervised, and unsupervised state-of-the-art methods. (*denotes results with video-based Hungarian matching).

Approach	F1
GMM [33]	27.0%
CTE-MLP [16]	28.3%
(LSTM+AL [1])	(39.7%*)
Ours	**29.86%**

Table 2: Segmentation results on the YTI dataset compared to unsupervised state-of-the-art methods. (*denotes results with video-based Hungarian matching).

INRIA YouTube Instructional Videos (YTI): The YTI Dataset is particularly challenging as the majority of its frames consist of "background" information (see Figure 5). That is, most of the frames in this dataset do not provide any relevant visual cues related to the action we are segmenting. In line with other approaches, our evaluation on

Supervision	Approach	Eval	Mid
Full	S-CNN [18]	68.0%	54.9%
	ST-CNN[18]	72.00%	58.06%
	ED-TCN [17]	73.4%	64.7%
	MS-TCNN++ [20]	–	83.7%
	Local SSTDA [6]	–	82.8%
	SSTDA[18]	–	**83.8%**
Unsupervised	CTE-MLP [16]	**35.5%**	**30.2%**
	(LSTM+AL [1])	(60.6%)*	-
	Ours	30.59%	24.19%

Table 3: MoF Segmentation results on the 50 Salads dataset at mid and eval granularity compared to state-of-the-art methods. (*denotes results with video-based Hungarian matching) Blank fields were left for methods that did not report a score for a type of granularity.

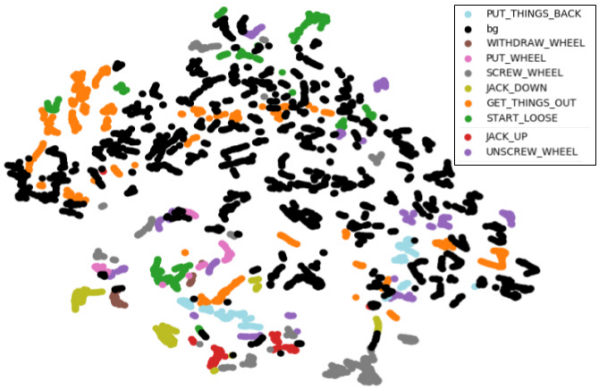


Figure 5: t-SNE[37] Visualization of the cluster distribution on a video from the YTI Dataset. Notice the heavy presence of background (bg) frames.

this dataset excludes any frames our method assigned to the label background so as to avoid any bias towards the over-segmentation of this sub-activity. Despite the reduced amount of valuable visual cues we can gather from single frames in this data, our method outperforms other global-based Hungarian matching approaches (see Table 2). We observed a considerable improvement when using larger frame prediction steps (predicting further into the future) for this dataset, which probably helped our visual embedding model to discriminate the minutiae associated with irrelevant frames.

50 Salads (50S): While more similar in nature to the videos in Breakfast Actions, the order of the sub-activities in 50 Salads is not as important as with the previous datasets. Sub-activities belonging to the same category have similar temporal locations (*e.g.*, "cutting tomato", and "cutting cucumber"), and often the order in which they are performed does not have a big impact on the performance of the task. Consequently the benefits we see from following the two training stages (Table 3) are not as significant as

Step size	BF	YTI	50S (eval)
U-Net (s=0)	33.34%	12.16%	31.17%
U-Net (s=1)	42.06%	13.51%	28.56%
U-Net (s=3)	47.17%	21.32%	28.93%
U-Net (s=5)	**47.27%**	**25.72%**	**31.24%**

Table 4: U-Net Learning task impact: Given an input frame captured at a time t the U-Net was trained to estimate the frame s steps after the input frame. MoF is reported for BF and 50 Salads, F1 score is reported for YTI.

for the other two datasets, just using the visual embedding from the first stage seemed to provide fewer distractions (brought on by the weak temporal ordering of the actions in this dataset) and resulted on a better performance (see Table 4).

5.1. Ablation Studies

5.1.1 Future frame prediction task

In order to assess the value of future frame prediction as a good learning task to encode both the visual and temporal cues of the data, we tested the performance of our visual embedding model (Section 3.1) using a frame prediction step sizes s of 0, 1, 3, and 5 (see Table 4). When $s = 0$ the network would learn to reconstruct the same input frame from the generated encoding (rather than generating a prediction of a future frame), with $s > 0$ the network would predict the next s_{th} frame in the video sequence following the input frame (we should be able to generate a given frame t from the encoding of a previously seen frame at time $t-s$).

We observed a distinctive improvement for both the BF and YTI Datasets as the step size increases, with the lowest performance obtained by just reconstructing the original input frame ($s = 0$). The strong correlation between temporal and visual cues in these datasets allowed us to extract a more significant encoding using our approach. However, it is important to note that, while our approach benefits from the future frame prediction task, when testing overly large step sizes (see Fig. 6) the performance degrades as the step size increases. Overly large step sizes could exacerbate the erosion of shorter sub-activities from the temporal segmentation as well as disrupt the learning of the logical ordering of the sub-activities in a given action.

This was not the case for the 50 Salads dataset, while overall the U-Net with the largest step size obtained the best performance, the further enforcement of the temporal encoding brought on by the second stage of our training pipeline proved to be detrimental as the temporal cues are not that significant in this type of data.

5.1.2 Feature Embedding

The purpose of our embedding algorithm is to encode valuable visual and temporal information derived from our input features, to evaluate the impact of such embedding, we first

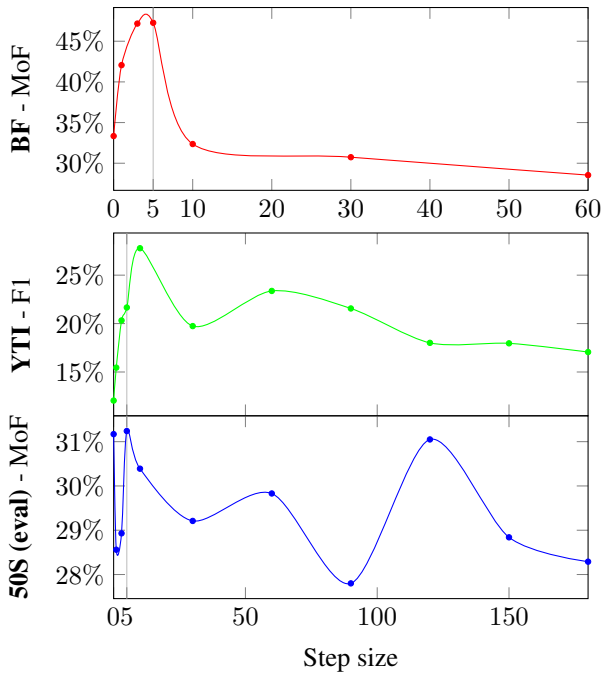


Figure 6: U-Net Embedding Performance when predicting longer step sizes (s). We do not test $s > 60$ for BF as some of the videos in the dataset have as few as 180 frames.

analyze the performance of the segmentation pipeline when omitting the feature embedding step. For this, we skipped the embedding process and directly clustered the reduced Fisher Vectors following the process described in Sec. 3.3. We observed a significant decrease in the performance obtained for all three datasets, with a 13.5% F1 score for the INRIA Youtube Instructional Videos (YTI) Dataset, and an MoF score of 28.13% and 29.07% for the Breakfast Actions (BF) and 50 Salads (50S) Datasets respectively.

As described in Sec. 3, we train our visual (U-Net) and temporal (MLP) components in two stages to obtain a joint visual and temporal embedding (extracted from the visual model after the second stage). We now evaluate the impact of different types of embedding (visual, temporal, or joint) during both stages of our training. Table 5 shows the result using the embedding of the visual and temporal components, the U-Net and the MLP respectively. At stage 1 both components were trained in a disjoint fashion, following the losses described in Equations 1 (U-Net) and 3 (MLP). At stage 2 the loss of both models is updated so that they are trained jointly (alternating the training of the U-Net and MLP for 40 and 5 epochs respectively) following Equations 4 (U-Net) and 6 (MLP). We used the output of the U-Net's encoder as our embedding for the U-Net, and the output of the inner layer of the MLP as its embedding. Overall the visual embedding of the U-Net after the first stage already led to an increased performance of the whole system. Nevertheless, further enforcing a stronger temporal embedding through stage 2 provided the best performance for the Breakfast Actions Dataset.

Training	Embedding Source	MoF
–	Fisher Vector (no embedding)	28.13%
Stage 1	U-Net	47.27%
	MLP	40.91%
Stage 2	U-Net	**48.08%**
	MLP	39.04%

Table 5: Analysis of the feature embedding sources on the Breakfast Actions dataset at both stages of training. Stage 1 Models were trained independently of each other.

6. Discussion

Various approaches have been proposed for learning visual representations without labels [12, 16, 1, 33]. Particularly, in the case of learning video representations in an unsupervised way, the temporal properties of the data have been commonly employed to understand the structure of the observed visual features and the rules they follow while they change over time. Following the idea that given a set of videos all capturing the same activity it should be possible to identify temporal segments with similar sub-actions across all videos, we have studied the suitability of a joint visual-temporal approach to learn to identify different action segments from unlabeled video data.

The results of our experiments have shown that a combination of predictive visual and temporal learning tasks encourages the encoding of valuable information for temporal segmentation that might have been otherwise disregarded by a solely temporal approach. The segmentations obtained with our method maintain the logical ordering of the analyzed tasks and are able to produce coherent segments that have led us to significant improvements over previous approaches that have exclusively used temporal embeddings.

We have observed that incorporating the visual cues that might be present in a frame into our joint approach has shown promise in the area of temporal segmentation, particularly for cases in which the sub-activities have either 1) *distinct visual* appearances (e.g., "cracking egg" vs. "buttering pan"), and/or 2) a *strong temporal coherence* (e.g., the "making coffee" sub-activities "pouring coffee" and "pouring milk" would have similar visual aspects, however, in most cases the actors in the videos pour the coffee (into the mug) first before adding milk to it), as was the case for datasets like Breakfast Actions and INRIA Youtube Instructions (YTI)). However, our approach struggles with data that has a weak temporal coherence and low visual variance (like the data in 50 Salads). We suggest as a future research direction incorporating an evaluation of the temporal coherence of sub-activities, both to address this issue and to incorporate the segmentation of repetitive actions (which lead to inconsistent results in their temporal location).

References

[1] Sathyanarayanan N. Aakur and Sudeep Sarkar. A perceptual prediction framework for self supervised event segmentation. In *CVPR*, June 2019.

[2] J. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *CVPR*, 2016.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *PAMI*, 39(12):2481–2495, 2017.

[4] Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In *ECCV*, 2014.

[5] Chien-Yi Chang, De-An Huang, Yanan Sui, Li Fei-Fei, and Juan Carlos Niebles. D3tw: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation. In *CVPR*, pages 3546–3555, 2019.

[6] Min-Hung Chen, Baopu Li, Yingze Bao, Ghassan AlRegib, and Zsolt Kira. Action segmentation with joint self-supervised temporal domain adaptation, 2020.

[7] Emily L Denton and vighnesh Birodkar. Unsupervised learning of disentangled representations from video. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NIPS*, pages 4414–4423. Curran Associates, Inc., 2017.

[8] Li Ding and Chenliang Xu. Weakly-supervised action segmentation with iterative soft boundary assignment. In *CVPR*, 2018.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[10] De-An Huang, Li Fei-Fei, and Juan Carlos Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *ECCV*, pages 137–153. Springer, 2016.

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, pages 1125–1134, 2017.

[12] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *CoRR*, abs/1902.06162, 2019.

[13] H. Kuehne, A. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *CVPR*, 2014.

[14] Hilde Kuehne, Juergen Gall, and Thomas Serre. An end-to-end generative framework for video segmentation and recognition. In *WACV*, 2016.

[15] H. Kuehne, A. Richard, and J. Gall. A hybrid rnn-hmm approach for weakly supervised temporal action segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019.

[16] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *CVPR*, 2019.

[17] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *CVPR*, pages 156–165, 2017.

[18] Colin Lea, Austin Reiter, René Vidal, and Gregory D Hager. Segmental spatiotemporal cnns for fine-grained action segmentation. In *European Conference on Computer Vision*, pages 36–52. Springer, 2016.

[19] Jun Li, Peng Lei, and Sinisa Todorovic. Weakly supervised energy-based learning for action segmentation. In *ICCV*, pages 6243–6251, 2019.

[20] Shijie Li, Yazan Abu Farha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. Ms-tcn++: Multi-stage temporal convolutional network for action segmentation, 2020.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[22] William Lotter, Gabriel Kreiman, and David D. Cox. Unsupervised learning of visual structure using predictive generative networks. *ICLR*, abs/1511.06380, 2015.

[23] William Lotter, Gabriel Kreiman, and David D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. *ICLR*, abs/1605.08104, 2016.

[24] Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nick Johnston, Andrew Rabinovich, and Kevin Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. *NAACL*, 2015.

[25] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. 2015.

[26] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. 2014.

[27] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV*, 2016.

[28] Xiaojiang Peng, Changqing Zou, Yu Qiao, and Qiang Peng. Action recognition with stacked fisher vectors. In *European Conference on Computer Vision*, pages 581–595. Springer, 2014.

[29] Alexander Richard and Juergen Gall. Temporal action detection using a statistical language model. In *CVPR*, pages 3131–3140, 2016.

[30] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *CVPR*.

[31] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *CVPR*, 2018.

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, abs/1505.04597, 2015.

[33] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *CVPR*, pages 8368–8376, 2018.

[34] Ozan Sener, Amir Roshan Zamir, Chenxia Wu, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic action discovery from video collections. *CoRR*, abs/1605.03324, 2016.

[35] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015.

[36] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738, 2013.

[37] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-sne effectively. *Distill*, 2016.

[38] William F. Whitney, Michael Chang, Tejas D. Kulkarni, and Joshua B. Tenenbaum. Understanding visual concepts with continuation learning. *ICLR*, abs/1602.06822, 2016.

[39] Jeffrey M. Zacks and Khena M. Swallow. Event segmentation. *Current Directions in Psychological Science*, 16(2):80–84, 2007.

[40] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *NIPS*, pages 465–476, 2017.