Learning Event Representations for Temporal Segmentation of Image Sequences by Dynamic Graph Embedding

Mariella Dimiccoli, Herwig Wendt

Learning event representations for temporal segmentation of image sequences by dynamic graph embedding

Mariella Dimiccoli and Herwig Wendt
[*][†][‡][§]

December 8, 2020

Abstract

Recently, self-supervised learning has proved to be effective to learn representations of *events* suitable for temporal segmentation in image sequences, where events are understood as sets of temporally adjacent images that are semantically perceived as a whole. However, although this approach does not require expensive manual annotations, it is data hungry and suffers from domain adaptation problems. As an alternative, in this work, we propose a novel approach for learning event representations named *Dynamic Graph Embedding* (DGE). The assumption underlying our model is that a sequence of images can be represented by a graph that encodes both semantic and temporal similarity. The key novelty of DGE is to learn jointly the graph and its graph embedding. At its core, DGE works by iterating over two steps: 1) updating the graph representing the semantic and temporal similarity of the data based on the current data representation, and 2) updating the data representation to take into account the current data graph structure. The main advantage of DGE over state-of-the-art self-supervised approaches is that it does not require any training set, but instead learns iteratively from the data itself a low-dimensional embedding that reflects their temporal and semantic similarity. Experimental results on two benchmark datasets of real image sequences captured at regular time intervals demonstrate that the proposed DGE leads to event representations effective for temporal segmentation. In particular, it achieves robust temporal segmentation on the EDUBSeg and EDUBSeg-Desc benchmark datasets, outperforming the state of the art. Additional experiments on two Human Motion Segmentation benchmark datasets demonstrate the generalization capabilities of the proposed DGE.

Index Terms: *clustering, event representations, geometric learning, graph embedding, temporal context prediction, temporal segmentation*

1 Introduction

Temporal segmentation of videos and image sequences has a long story of research since it is crucial not only to video understanding but also to video browsing, indexing and summarization [1–3]. With

[*] Manuscript received August 10, 2019; revised June 19, 2020; October 30, 2020, December 4, 2020, December 8, 2020; accepted December 9, 2020.

[†] M. Dimiccoli is with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain (`mdimiccoli@iri.upc.edu`).

[‡] H. Wendt is with the Institut de Recherche en Informatique de Toulouse, CNRS, University of Toulouse, France (`herwig.wendt@irit.fr`).

[§] This work was partially supported by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund (MINECO/ERDF, EU) through the program Ramon y Cajal and the national Spanish projects PID2019-110977GA-I00, RED2018-102511-T and 2017 SGR 1785. We acknowledge the support of NVIDIA Corporation with the donation of Titan Xp GPUs.

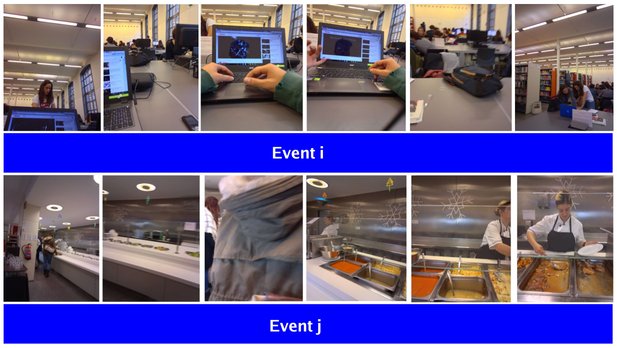


Figure 1: Temporally adjacent frames in two events in first-person image sequences.

the proliferation of wearable cameras in recent years, the field is facing new challenges. Indeed, wearable cameras allow to capture, from a first-person (egocentric) perspective, and "in the wild", long unconstrained videos (≈35fps) and image sequences (aka *photostreams*, ≈2fpm). Due to their low temporal resolution, the segmentation of first-person image sequences is particularly challenging, and has received special attention from the community [4–14]. Indeed, abrupt changes in appearance may arise even between temporally adjacent frames within an event due to sudden camera movements and the low frame rate, making it difficult to distinguish them from event transitions. While for a human observer it is relatively easy to segment egocentric image sequences into discrete units, this poses serious difficulties for automated temporal segmentation (see Figure 1 for an illustration). In particular, classical spatio-temporal video representations, that typically rely on motion information [15–17], cannot be reliably computed on photostreams due to this lack of temporal continuity [18].

Given the limited amount of annotated data, current state-of-the-art approaches for the temporal segmentation of first-person image sequences aim at obtaining event representations by encoding the temporal context of each frame in an unsupervised fashion [12, 13]. These methods rely on neural or recurrent neural networks and are generally based on the idea of learning event representations by training the network to predict past and future frame representations. Recurrent neural networks have proved to be more efficient than simple neural networks for the temporal prediction task. The main limitation of these approaches is that they must rely on large training datasets to yield state-of-the-art performance. Even if, in this case, training data do not require manual annotations, they can nevertheless introduce a bias and the learnt models can suffer from the domain adaptation problem. For instance, in the case of temporal segmentation of image sequences, the models will be difficult to generalize to data acquired with a camera with different field of view or for people having different lifestyles.

In this paper, we aim at overcoming this limitation with a novel approach that is able to unveil a representation that encodes the temporal and semantic similarity of an image sequence from the single sequence itself. With this goal in mind, we propose to learn event representations as an embedding on a graph. Our model is based on the assumption that each event belongs to a particular semantic context that can be shared across semantically similar events. These semantic contexts can be represented as communities on an unknown underlying graph. In particular, graph nodes correspond to individual frames, and edges between nodes encode frame similarity. The communities are understood here as sets of nodes (frames) that are interconnected by edges with large weights. Moreover, the graph weights reflect not only temporal proximity, but also semantic similarity, which is understood here as similarity in terms of high-level visual features. This is motivated by neuroscientific findings which show that neural representations of events arise from temporal community structures [19] and suggest that frames which share the context are grouped together in the representational space. In Figure 2 we illustrate this idea by means of an egocentric image sequence capturing the full day of a person: going from *home* to *work* using *public transports*, having a lunch break in a *restaurant* and going back to *home* after doing some *shopping*, etc. Each point cloud corresponds with images similar in appearance and most of them are visited multiple times. This means that every pair of images in a point cloud is related semantically, but

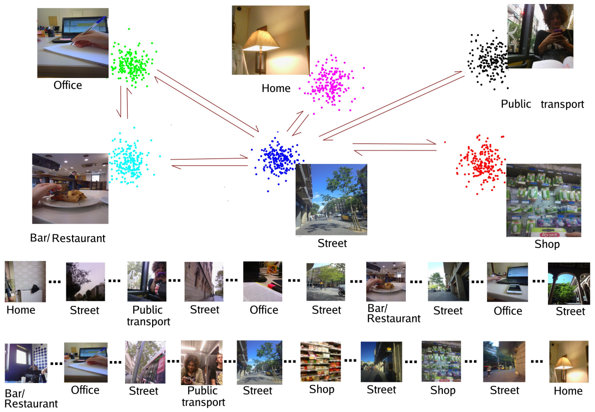


Figure 2: The assumption underlying our learning approach is that image sequences captured at regular time intervals (2fpm) can be organized into a graph structure, where each community in the graph corresponds to a particular semantic context. Points of the same color in the figure are related semantically and may be more or less related at temporal level. The arrows indicate temporal transitions between communities. They have only a visualization purpose, since temporal transition are between pairs of points.

they could or could not be related at temporal level.

Based on this model, the proposed solution consists in learning simultaneously the graph structure (encoded by its weights) and the data representation. This is achieved by iterating over two alternate steps: 1) update of the graph structure as a function of the current data representation, where the graph structure is assumed to encode a finite number of communities, and 2) update of the data representation as a function of the current graph structure in a low-dimensional embedding space. We term this solution *dynamic graph embedding* (DGE). We provide illustrative experiments on synthetic data, and we validate the proposed approach on two real world benchmark datasets for first-person image sequence temporal segmentation. Our framework is the first attempt to learn simultaneously graph structure and data representations for temporal segmentation of image sequences.

Our main contributions are: (i) we re-frame the event learning problem as the problem of learning a graph embedding, (ii) we introduce an original graph initialization approach based on the concept of temporal self-similarity, (iii) we propose a novel technical approach to solve the graph embedding problem when the underlying graph structure is unknown, (iv) we demonstrate that the learnt graph embedding is suitable for the task of temporal segmentation, achieving state-of-the-art results on two challenging reference benchmark datasets [11,20], without relying on any training set for learning the representation, (v) we show that the proposed DGE generalizes to other problems, yielding state-of-the-art results also on two reference benchmark datasets for the Human Motion Segmentation problem.

The structure of the paper is as follows. Section 2 highlights related work on data representation learning on graphs and on the temporal segmentation of videos and image sequences. In Section 3.1 we introduce our problem formulation while in Sections 3.2 to 3.5 we detail the proposed graph embedding model. The performance of our algorithm on real world data are evaluated in Section 4. In Section 5, we conclude on our contributions and results.

2 Related work

2.1 Geometric learning

The proposed approach lies in the field of geometric learning, which is an umbrella term for those techniques that work in non-Euclidean domains such as graphs and manifolds. Following [21], geometric learning approaches either deal with analyzing functions defined on a given non-Euclidean domain or address the problem of characterizing the structure of the data. The former case includes methods dealing with manifolds [22, 23] as well as methods of signal processing on graphs [24]. These allow to generalize CNN to graphs [25, 26] by defining an operation similar to convolution in the graph spectral domain. In the latter case, which is more closely related with the method proposed in this paper, the goal is to learn an embedding of the data in a low-dimensional space such that the geometric relations in the embedding space reflect the graph structure. These methods are commonly referred to as *node-embedding* and can be understood from an encoder-decoder perspective [27]. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ represent the set of nodes and edges of the graph respectively, the encoder maps each node $v \in \mathcal{V}$ of $\mathcal{G}$ in a low-dimensional space. The decoder is a function defined in the embedding space that acts on node pairs to compute a similarity S between the nodes. Therefore the graph embedding problem can be formulated as the problem of optimizing decoder and encoder mappings such that they minimize the discrepancy between similarity values in the embedding and original feature space.

Within the general encoder-decoder framework, node embedding algorithms can be roughly classified into two classes. The first class covers shallow embedding methods, including matrix factorization [28–30] and random-walk based approaches [31–34]. The second class subsumes generalized encoder-decoder architectures [35–38]. In shallow embedding approaches, the encoder function acts simply as a lookup function and the input nodes $v_i \in \mathcal{V}$ are represented as one-hot vectors, so that they cannot leverage node attributes during encoding. Instead, in generalized encoder-decoder architectures [35–37] the encoders depend on the structure and attributes of the graph. In particular, convolutional encoders [38] rely on node features to generate embeddings for a node by aggregating information from its local neighborhood, in a manner similar to the receptive field of a convolutional kernel in image processing. As the process is iterated, the node embedding contains information aggregated from further and further reaches of the graph. Closely related to convolutional encoders are Graph Neural Networks (GNNs). The main difference is that GNNs capture the graph's internal dependencies via message passing between its nodes. Moreover, GNNs can utilize node attributes and node labels to train model parameters end-to-end for a specific task in a semi-supervised fashion [25, 26, 39, 40].

In all these methods, the graph structure is assumed to be given by the problem domain. For instance, the graph structure for social networks can be inferred from the connections between people. However, in the case of temporal segmentation considered here the problem is non-structural since the graph structure is not given by the problem domain. Instead, it needs to be determined together with the node embedding.

2.2 Event segmentation

Extensive research has been conducted to temporally segment videos and image sequences into events. Early approaches aimed at segmenting edited videos such as TV programs and movies [41–45] into commercial, news-related or movie events. This includes the use of the concept of Logical Story Units (LSU), defined as *a series of shots that communicate a unified action with a common setting and time*; in particular, [41] proposed a method to segment TV programs into LSUs by firstly clustering given video shots and then building a Scene Transition Graph with nodes corresponding to the clusters and edges to temporal transitions. More recently, with the advent of wearable cameras and camera equipped smartphones, there has been an increasing interest in segmenting untrimmed videos or image sequences captured by nonprofessionals into semantically homogeneous units [46–48]. In particular, videos or image sequences captured by a wearable camera are typically long and unconstrained [18]. Therefore it is important for the user to have them segmented into semantically meaningful chapters. In addition to appearance-based features [49, 50], motion features have been extensively used for temporal segmentation of both third-person videos [1, 47] and first-person videos [51–54]. In [52], motion cues from a wearable inertial sensor are leveraged for the temporal segmentation of human motion into actions. Lee and Grauman [53] used temporally constrained clustering of motion and visual features to determine whether the differences in appearance correspond to event boundaries or just to abrupt head movements. Poleg et al. [54] proposed to use integrated motion vectors to segment egocentric videos into a hierarchy of long-term activities whose first level corresponds to static/transit activities.

However, motion information is not available in first-person image sequences that are the main focus of this paper. In addition, given the limited amount of annotated data, event segmentation is very often performed by using a clustering approach that takes as input hand-crafted visual features such as color [4], MPEG7 descriptors [6], a combination of environmental sensor data, SIFT, SURF and MPEG7 descriptors [5], or a combination of CNN-based features [9, 10]. Tavalera et al. [7] proposed to combine agglomerative clustering with a change detection method within a graph-cut energy minimization framework. Later on, [11] extended this framework and proposed an improved feature representation by building a vocabulary of concepts. Paci et al. [8] proposed a Siamese ConvNets based approach that aims at learning a similarity function between low temporal resolution egocentric images. Recently, [13] proposed to learn event representations as the byproduct of learning to predict the temporal context. In this work, the single image sequence itself is employed to learn the new representation by using an autoencoder model or a LSTM encoder-decoder model, without relying on a training dataset. Molino et al. [12] later proposed a similar LSTM based model that achieved impressive results on the EDUBSeg dataset by leveraging on more powerful initial features and by relying on a large training dataset (over 1.2 million images).

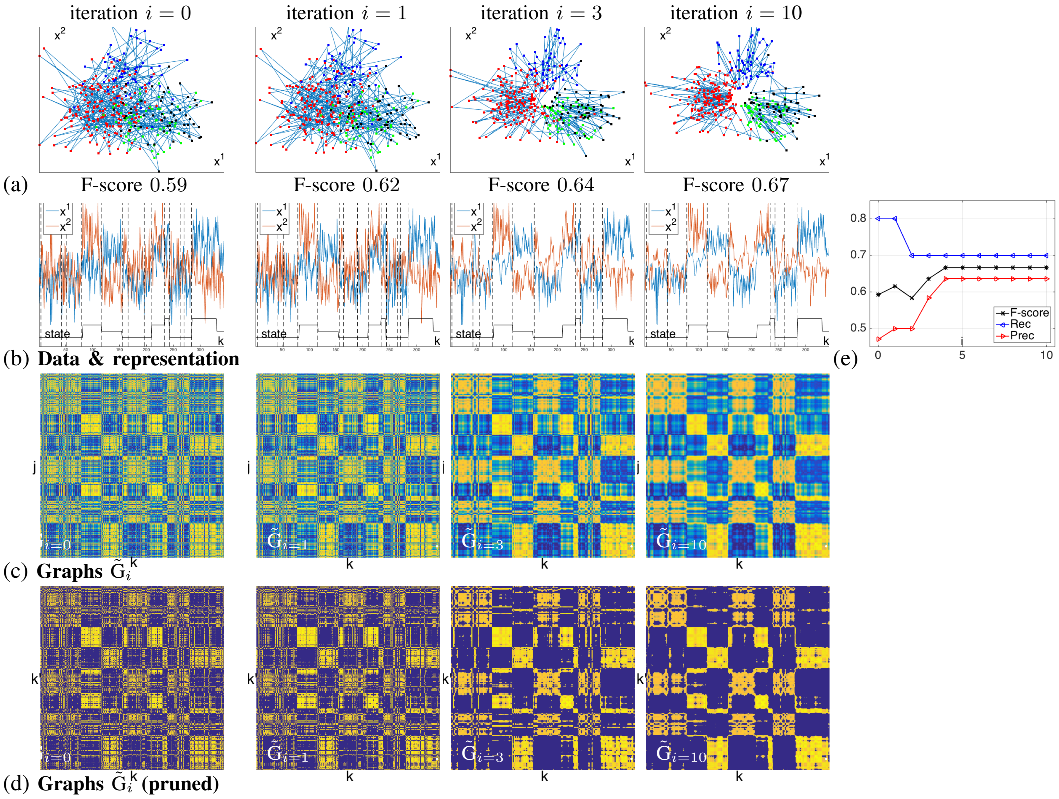
Here, we propose a new model that as in [13] does not make use of any training set for learning the temporal event representation, but achieves state-of-the-art results. In particular, it outperforms [12] on the EDUBSeg and EDUBSeg-Desc benchmarks [11, 20].

3 Dynamic Graph Embedding (DGE)

3.1 Problem formulation and proposed model

We formulate the event learning problem as a geometric learning problem. More specifically, given a set of data points (the frames of the image sequence) embedded into a high-dimensional Euclidean space (the initial data representation), we assume that these data points are organized in a graph in an underlying low-dimensional space. Here, graph nodes correspond to individual frames, and edges between nodes encode frame similarity. Our *a priori* on the structure of the graph is that it consists of a finite number of communities (sets of nodes (frames) that are interconnected by edges with large weights) corresponding to different semantic contexts. Since along an image sequence a same community can be visited several times at different time intervals, we assume that edges between nodes belonging to different communities correspond to transitions between different semantic contexts. In contrast, edges between nodes belonging to the same community correspond to transitions between nodes sharing the same semantic context, being them temporally adjacent or not. This structure implicitly assumes that the graph topology *models jointly temporal and semantic similarity relations.*

More formally, let $X \in \mathbb{R}^N \times \mathbb{R}^n$ denote the n-dimensional feature vectors for a given sequence of N images, and S a similarity kernel. We aim at finding a fully connected, weighted graph $\tilde{\mathcal{G}} = (\tilde{X}, \mathcal{E}, \tilde{\mathcal{W}})$ with node embedding $\tilde{X} \in \mathbb{R}^N \times \mathbb{R}^d$ in a low-dimensional space, $d \ll n$, and edge weights $\tilde{\mathcal{W}}$ given by the entries of an affinity matrix $\tilde{\mathrm{G}} = S(\tilde{X})$ such that the similarity $\tilde{\mathrm{G}}_{kj}$ between any pair k, j of nodes of the graph reflects both semantic relatedness and temporal adjacency between the images. Semantic relatedness is captured by a similarity function between high-level visual image descriptors, whereas temporal adjacency is imposed through temporal constraints on the edge weights. The above constraints lead to easily grouping the graph nodes in a finite number of communities that each correspond to a different semantic context. As seen in the previous section, in classical node embedding the low-dimensional representation of each node encodes information about the position and the structure of the local neighborhood in the graph. Since all these methods incorporate graph structure in some way, the construction of the underlying graph is extremely important but relatively little explored. In our problem at hand the graph structure is initially unknown since it arises from unknown events. Therefore, we aim at learning jointly the structure of the underlying graph and the node embedding.



(a)

(b) **Data & representation**

(c) **Graphs** $\tilde{G}_i^k$

(d) **Graphs** $\tilde{G}_i^k$ **(pruned)**

(e)

Figure 3: **Illustration of DGE on synthetic data modeling 4 communities in $d = 2$ dimensions**: (a) scatter plot of features $\tilde{X}_i$ for iterations $i = (0, 1, 3, 10)$ (left to right column, respectively); color indicates community ground truth, solid lines indicate temporal adjacency; (b) the features $\tilde{X}_i$ plotted as time series, with estimated segment boundaries (vertical dashed lines) and community ground truth (solid black); (c) the corresponding graphs $\tilde{\mathcal{G}}_i$ and (d) $\tilde{\mathcal{G}}_i$ after removal of edges with weight smaller than 0.7 times that of the strongest edge (yellow and blue color correspond with strong and weak similarity, respectively); (e) F-score, precision and recall for estimated segmentation as a function of iteration number i.

3.2 Graph initialization by nonlocal self-similarity

Temporal nonlocal self-similarity. To obtain a first coarse estimate of the graph, we apply a nonlocal self-similarity algorithm in the temporal domain to the initial data X that we normalize to the interval $[-1, 1]$ [14]. The nonlocal self-similarity filtering creates temporal neighborhoods of frames that are likely to be in the same event. Let $X(k) \in \mathbb{R}^n$ denote the k-th row of X, that is, the vector of n image features at time k, $k = 1, \ldots, N$. Further, let $\mathcal{N}_k^M = \{k - M, \ldots, k - 1, k + 1, \ldots, k + M\}$ and $\mathcal{N}_k^L = \{k - L, \ldots, k - 1, k + 1, \ldots, k + L\}$ denote the indices of the $2M$ and $2L$ neighboring feature vectors of $X(k)$, respectively, with $L > M$. In analogy with 2D data (images) [55], the self-similarity function of $X(k)$ in a temporal sequence, conditioned to its temporal neighborhood $j \in \mathcal{N}_k^M$, is given by the quantity [14]

$$S^{NL}(k, j) = \frac{1}{\mathcal{Z}(k)} \exp\left(-\frac{dist(X(\mathcal{N}_k^M), X(\mathcal{N}_j^M))}{h} \right). \tag{1}$$

Here $\mathcal{Z}(k)$ is a normalizing factor such that $\sum_{j\in\mathcal{N}_k^L} S_{kj}^{NL} = 1$, ensuring that S_{kj}^{NL} can be interpreted as a conditional probability of $X(j)$ given $X(\mathcal{N}_k)$, as detailed in [55], $dist(X(\mathcal{N}_k), X(\mathcal{N}_j)) = \sum_{i=1}^{2M} ||X(\mathcal{N}_k(i)) - X(\mathcal{N}_j(i))||_{\ell_1}$ is the sum of the ℓ_1 distances of the vectors in the neighborhoods of k and j, and h is the parameter that tunes the decay of the exponential function. The key idea of our graph initialization is to model each frame k by its denoised version, obtained as

$$\begin{aligned}\hat{X}(k) &= \text{NLmeans}^{1D}(X(k)) = \sum\nolimits_{j\in\mathcal{N}_k^L} S_{kj}^{NL} \cdot X(j),\\ \hat{X}_0 &= (\hat{X}(1)^T\ \hat{X}(2)^T\ \ldots\ \hat{X}(N)^T)^T. \end{aligned} \quad (2)$$

A numerical illustration on real data is provided in Figure 4.

Initial graph and initial embedding. An initial graph $\mathcal{G}_0$ is obtained by computing the $\mathbb{R}^N \times \mathbb{R}^N$ affinity matrix $\mathrm{G}_0 = S_{\hat{l}}(\hat{X}_0)$ of $\hat{X}_0$, defined elementwise as the pairwise similarity

$$(S_l(X))_{kj} = \exp\left(-\frac{1 - cdist(X(j), X(k))}{l}\right) \quad (3)$$

where $cdist(\cdot,\cdot)$ is the cosine distance and l the filtering parameter of the exponential function. In the following, we will not distinguish any longer between a graph $\mathcal{G}$ and the representation by its affinity matrix G and make use of both symbols synonymously. In our model, $\mathcal{G}_0$ represents the initial data structure in the original high dimensional space as a fully connected graph, from which we aim to learn a graph in the embedding space that better encodes temporal and semantic constraints, denoted $\tilde{\mathcal{G}}$. To obtain an initial embedding $\tilde{X}_0$ for the graph, we apply PCA on $\hat{X}_0$, keep the d major principal components $\tilde{X}$ and minimize the cross-entropy loss $\mathcal{L}$ between the affinity matrices $\mathrm{G}_0 = S_{\hat{l}}(\hat{X}_0)$ and $S_{\tilde{l}}(\tilde{X}_0)$

$$\tilde{X}_0 = \operatorname{argmin}_{\tilde{X}} \mathcal{L}(S_{\tilde{l}}(\tilde{X}), \mathrm{G}_0) \quad (4)$$

where the different filtering parameters $\hat{l}$ and $\tilde{l}$ account for the different dimensionality of $\hat{X}_0$ and $\tilde{X}_0$. Even if PCA is a linear operator and for small sets of high-dimensional vectors dual PCA could be more appropriate [56], we found it sufficient here for initializing the algorithm. The initial graph $\tilde{\mathcal{G}}_0$ in the embedding space is then given by $\tilde{\mathrm{G}}_0 = S_{\tilde{l}}(\tilde{X}_0)$.

3.3 DGE core alternating steps

Given the initial embedding $\tilde{X}_0$ and graphs $\mathcal{G}_0$ and $\tilde{\mathcal{G}}_0$, the main loop of our DGE alternates over the following two steps:

1. Assuming that $\tilde{\mathcal{G}}_{i-1}$ is fixed, update the node representations $\tilde{X}_i$.

2. Assuming that $\tilde{X}_i$ is fixed, update the graph $\tilde{\mathcal{G}}_i$.

Step (1) is inspired from graph embedding methods, such as the ones reviewed in Section 2.1, which have proved to be very good at encoding a given graph structure. Step (2) aims at enforcing temporal constraints and at fostering semantic similarity in the graph structure.

Graph embedding update. To estimate the graph embedding $\tilde{X}_i$ at iteration i assuming that $\tilde{\mathcal{G}}_{i-1}$ is given, we solve

$$\tilde{X}_i = \underset{\tilde{X}}{\operatorname{argmin}} (1-\alpha)\mathcal{L}_1(S_{\tilde{l}}(\tilde{X}), \tilde{\mathrm{G}}_{i-1}) + \alpha\mathcal{L}_2(S_{\tilde{l}}(\tilde{X}), \mathrm{G}_0). \quad (5)$$

Here $\mathcal{L}_1$ and $\mathcal{L}_2$ are cross-entropy losses and $S_{\tilde{l}}(\cdot,\cdot)$ is the cosine-distance based similarity defined in (3). The first loss term controls the fit of the representation $\tilde{X}$ with the learnt graph $\tilde{\mathcal{G}}_i$ in low-dimensional embedding space. The second loss term quantifies the fit of the representation $\tilde{X}$ with the fixed initial graph $\mathcal{G}_0$ in high dimensional space and is reminiscent of shallow graph embedding. The regularization parameter $\alpha \in [0, 1]$ controls the relative weight of each loss. Standard gradient descent can be used to solve (5). In the numerical experiments reported below, 150 iterations of gradient descent with Barzilai-Borwein adaptive step size [57] were used.

Graph structure update. To obtain an estimate of the graph structure at the i-th iteration, say $\tilde{\mathcal{G}}_i$, assuming that $\tilde{X}_i$ is given, we start from an initial estimate for $\tilde{\mathcal{G}}_i$ as $\tilde{\mathrm{G}}_i = S_{\tilde{l}}(\tilde{X}_i)$, and then make use of the model assumptions described in Section 3.1 to modify the graph: temporal adjacency, and semantic similarity.

i) To foster similarity for temporally adjacent nodes, we apply two operations. First, local averaging of the edge weights defined as $\tilde{\mathrm{G}}_i \leftarrow \tilde{\mathrm{G}}_i * \mathcal{K}_p$, where $*$ is the 2D convolution operator and $\mathcal{K}_p$ a $p \times p$

kernel that is here simply the normalized bump function. Second, application of the shrinkage operation

$$(\tilde{\mathrm{G}}_i)_{kj} \leftarrow (f_\eta(\tilde{\mathrm{G}}_i))_{kj} = \begin{cases} (1-\eta)(\tilde{\mathrm{G}}_i)_{kj} & \text{if } |k-j|>1 \\ (\tilde{\mathrm{G}}_i)_{kj} & \text{otherwise,} \end{cases} \tag{6}$$

which leaves the similarities of directly temporally adjacent nodes of $\tilde{\mathcal{G}}_i$ unchanged, but shrinks the weights of edges between nodes k and j that are not direct temporal neighbors by a factor η, $0 < \eta < 1$, thus strengthens the temporal adjacency of the graph.

ii) To reinforce the semantic similarity of $\tilde{X}_i$, we first obtain a coarse estimate of the community structure of the graph $\tilde{\mathcal{G}}_i$. To this end, we apply a clustering algorithm on $\tilde{X}_i$, which yields estimated cluster labels $\mathcal{C} = (c_j)_{j=1}^{N_C}$, $c_j \in \{1, ..., N_C\}$, for each frame, that roughly correspond to semantic contexts, i.e., communities. Then we modify $\tilde{\mathcal{G}}_i$ using the non-linear operation defined by

$$(\tilde{\mathrm{G}}_i)_{kj} \leftarrow (g_\mu(\tilde{\mathrm{G}}_i, \mathcal{C}))_{kj} = \begin{cases} (1-\mu)(\tilde{\mathrm{G}}_i)_{kj} & \text{if } c_j \neq c_k \\ (\tilde{\mathrm{G}}_i)_{kj} & \text{otherwise.} \end{cases} \tag{7}$$

This graph update reduces the similarity between nodes k and j that do not belong to the same cluster, $c_j \neq c_k$, by a factor μ, $0 < \mu < 1$, and does not change similarities within clusters, hence reinforces within-event semantic similarity.

Thus, both the embedding step and the clustering jointly contribute to learn representations that account for the semantics encoded by the initial CNN features. DGE aims at revealing the temporal and semantic relatedness for each pair of data vectors, and therefore the estimated graphs $\tilde{\mathcal{G}}_i$, $i = 0, \ldots, K$, are fully connected at each stage. A high-level overview of our DGE approach can be found on ALGORITHM 1.

3.4 Graph post-processing: Event boundary detection

Depending on the problem, applicative context and objective, different standard and graph signal processing tools can be applied to the estimated graph $\tilde{\mathcal{G}}_K$ in order to extract the desired information or to transform $\tilde{\mathcal{G}}_K$ [24, 58]. To evaluate the effectiveness of the learnt representation $\tilde{X}_K$ corresponding to $\tilde{\mathcal{G}}_K$ for event segmentation, we use it as the input of a boundary detector.

Boundary detector. To ensure a fair comparison of the learnt event representations with [12], we use the same boundary detector as therein, with the same parameter values and thresholds. It is based on the idea that when a vector $\tilde{X}(k)$ representing frame k corresponds to an event boundary, the distance between the predictions computed for it from past ($j < k$) and future ($j > k$) representation vectors is likely to be large. Consequently, [12] defines the boundary prediction function as the (cosine) distance between these contextual forward and backward predictions for frame k. Those frames for which the values of the boundary prediction function exceed a threshold are the detected event boundaries, see [12] for details.

Hereafter we call our temporal segmentation model relying on the features learnt by using the proposed DGE approach CES-DGE, in analogy with CES-VCP in [12] (where CES stands for contextual event segmentation and VCP for visual context prediction).

3.5 Numerical illustration of CES-DGE on synthetic data

To illustrate the main idea behind our modeling, we show with a synthetic example in $n = 2$ dimensions how the original data X and the associated initial graph $\tilde{\mathcal{G}}_0$ change over $K = 10$ iterations of the DGE algorithm (here, $d = n = 2$, and we assume $\tilde{X}_0 = \hat{X}_0 = X$, i.e., no preprocessing). The data consist of a temporal sequence of $N = 350$ feature vectors that are drawn from four different Gaussian distributions with mean vectors $(-7.8\ 5.1)$, $(-1.4\ 2.8)$, $(-2.9\ 12.1)$, $(2.5\ 3.4)$ and diagonal covariance matrix $\sigma\mathbf{I}$ with $\sigma = 3.7$. The different distributions are selected according to a Markov switching model in which the probability to remain in a state decreases from 1 at its onset at an exponential rate with time. This makes it likely that a reasonable number of temporally adjacent vectors are drawn from the same distribution, thus modeling a community corresponding with a semantic context.

Results are plotted in Figure 3 as scatter plots of the learnt representations $\tilde{X}_i$ (panel (a)) and as time series $\tilde{X}_i$ (panel (b)) for iterations $i = (0, 1, 3, 10)$. Clearly, it is difficult to obtain a good segmentation for the initial, cleaned data (iteration $i = 0$, left column). Yet, after a few DGE iterations the learnt representation vectors $\tilde{X}_i$ that belong to the same context are aggregated at similar values (Figure 3 (a-b), columns 2 to 4, respectively). At the same time, those vectors that belong to different contexts are pushed away from each other in the representational space. This effectively increases the similarity within each semantic context. An alternative view is provided by the corresponding learnt graphs $\tilde{\mathcal{G}}_i$ that are plotted in panels (c-d) in Figure 3. It can be observed how increasingly homogeneous diagonal and off-diagonal blocks with clear boundaries emerge with progressing iteration number i, reflecting the temporal community structure underlying the data. This improved representation leads in turn to significantly better event segmentation results, with F-score values increasing from initially 0.59 to 0.62, 0.64, 0.67 for iterations $i = 1, 3, 4 - 10$, respectively (cf., Figure 3, panel (e)).

4 Performance evaluation

4.1 Datasets and experimental setup

Datasets. We used two temporal segmentation benchmark datasets for performance evaluation. The EDUBSeg dataset, introduced in [11], consists of 20 first-person image sequences acquired by seven people with a total of 18,735 images. The dataset has been used to validate image sequence temporal segmentation in several recent works [8, 11–13]. For each image sequence, three manual segmentations have been obtained by three different persons. In line with previous works, the first segmentation was used here as the ground truth. The average performance of the two remaining segmentations is used as an estimate of the manual segmentation performance. The second dataset is the larger and more recent EDUBSeg-Desc dataset [20], with 46 sequences (42,947 images) acquired by eight people.

On average, the sequences of both datasets contain roughly the same number of ground truth event boundaries (28 for the former vs. 26 for the latter). However, those of EDUBSeg-Desc consist of 25% longer continuously recorded segments than EDUBSeg (3h46m25s vs. 3h1m29s continuous “camera on” time, and 3.0 vs. 3.55 continuously recorded segments per sequence). Since EDUBSeg-Desc is 50% larger than EDUBSeg and event transitions within the increased number of continuously recorded segments are more difficult to detect, EDUBSeg-Desc is considered more challenging [20].

Other publicly available datasets of egocentric image sequences, such as CLEF [59], NTCIR [60] and the more recent R3 [12], do not have ground truth event segmentations. They can therefore not be used for performance evaluation. These datasets with more than 1.2 million images were used for training in [12]. We emphasize that in contrast, our algorithm operates without training dataset.

Feature extraction. As in [12], each frame of the egocentric image sequences was described here using the output of the pre-pooling layer of InceptionV3 [61] pretrained on ImageNet, resulting in $n = 2048$ raw features $X(k)$ per frame k.

Performance evaluation. Following previous work [8, 11–13], we consider a detected event boundary to be correct when it falls within a range of $\pm\tau$ around the position of a true boundary. We use F-score, Precision (Prec) and Recall (Rec) to evaluate the performance of our approach. While previous work considered only a single level of tolerance $\tau = 5$, we here report results for several values $\tau \in (1, 2, 3, 4, 5)$.

DGE hyperparameters. The hyperparameter values for the graph initialization and embedding (i.e., for the non-local self-similarity kernel (1) and for the similarity (3)) have been chosen a priori based on visual inspection of the similarity matrices of $\hat{X}_0$ and $\tilde{X}_0$ for a few sequences of EDUBSeg. The values are fixed to $L = 3$, $M = 1$, $h = 0.25$ for Eq. (1), and $\hat{l} = 0.0025$, $\tilde{l} = 0.02d$ for Eq. (3). The embedding dimension is set to $d = 15$, which is found sufficient for the representation to faithfully reproduce the graph topology underlying the data. The influence of d is reported in the next section. The DGE core loop hyperparameters are set to $K = 2$ (DGE iterations), $\alpha = 0.1$ (graph embedding update), $p = 3$, $\eta = 0.3$ (temporal prior), and $\mu = 0.1$, $N_C = 10$ (semantic prior; a k-means algorithm is used to estimate cluster labels). These hyperparameter values have been selected by a grid search using the EDUB-Seg dataset. Our grid search strategy consisted in first tuning the embedding dimension d based on the quality of the initial embedding, and to perform preliminary individual line searches to determine search ranges and granularity for the remaining hyperparameters. The values retained by the grid search for the EDUB-Seg dataset are also used for the EDUB-SegDesc dataset, without modification; robustness of these choices is also investigated in the next section.

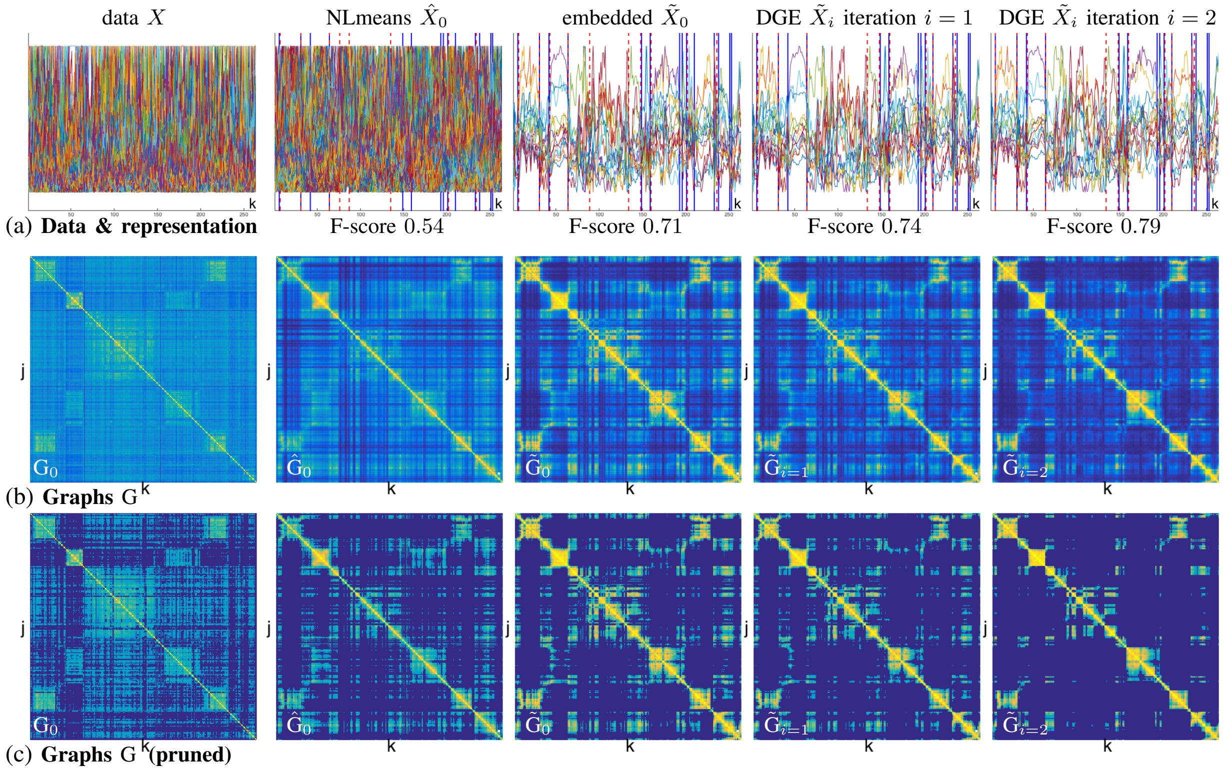
4.2 Robustness to changes in hyperparameter values

Table 1 reports F-score values obtained on the EDUBSeg dataset when the embedding dimension d, the DGE iterations K and the DGE core parameters α, p, η, μ and N_C are varied each individually. It can be appreciated that the performance of CES-DGE is overall robust w.r.t. precise hyperparameter values. As long as the embedding dimension d is chosen not too small and not too large, F-score values vary little (no more than 3 percentage points below the best observed F-score values for the range of embedding dimensions $10 < d < 50$) ; this corroborates similar findings on the existence of a trade-off between low-dimensionality and fidelity to the graph structure for node embedding in different contexts, cf., e.g., [27–38]. The highest sensitivity to DGE core loop hyperparameters is observed for the DGE iteration number K, whose optimal value is a compromise between increasing the similarity (i.e., choosing K large) between $\tilde{X}_i$ within correctly detected communities but not increasing it too much (i.e., K small) within incorrectly alienated communities. If chosen as $1 < K < 5$, F-score values are at most 3 percentage points below the best observed F-score. Results are also very robust w.r.t. temporal regularization for reasonably small parameter values of $p \leq 5$ and $\eta \leq 0.4$. For larger values the learnt representation is over-smoothed. Similarly, F-score values vary little when changing the semantic similarity parameter as long as $0 < \mu \leq 0.2$. Note that these variations are significant because $\eta, \mu \in (0, 1)$. Finally, F-score values drop by less than 3 percentage points when the number of clusters is selected within a reasonably large range $N_C \in (6, \ldots, 20)$. Overall, these results suggest that CES-DGE is quite insensitive to hyperparameter tuning and yields robust segmentation results for a wide range of hyperparameter values.

4.3 Comparative results for EDUB-Seg dataset

Table 2 reports comparisons with five state-of-the-art methods for a fixed value of tolerance ($\tau = 5$, results reproduced from [12]). The first four (k-means smoothed with k = 30, AC-color [53], SR-Clustering [11], KTS [62]) are standard/generic approaches and achieve modest performance, with F-scores no better than 0.53. CES-VCP of [12] yields significantly better F-score of 0.69, thanks to the use of a large training set for learning the event representation. The proposed CES-DGE approach further improves on this state-of-the-art result and yields F-score of 0.70. Interestingly, CES-DGE also achieves more balanced Rec and Prec values of 0.70 and 0.72, as compared to 0.77 and 0.66 for CES-VCP. Moreover, even when compared to average manual segmentation performance our results are only 2 percentage points below. Finally, we report results obtained by combining the Temporal Subspace Clustering (TSC) algorithm of [63] with the CES segmentation of [12] (CES-TSC), yielding F-score values 4 percentage points below the proposed CES-DGE.

In Table 3, we provide comparisons between our proposed approach and CES-VCP for different values of tolerance. We can observe that CES-DGE achieves systematically better results than CES-VCP in terms of F-score for all values of tolerance. These improvements with respect to the state of the art reach up to 4 percentage points. Besides, Rec/Prec values for CES-DGE are more balanced and within ±3 percentage points of the values for F-score for all tolerance levels (±8 percentage points of F-score for



(a) **Data & representation**

(b) **Graphs G^k**

(c) **Graphs G^k(pruned)**

Figure 4: **Illustration of DEG on Subject 1 Day 3 of EDUBSeg.** Panel (a): Data and learnt representations with boundary estimates (red dashed vertical bars), ground truth (blue solid vertical bars) and resulting F-score values; from left to right initial features X (1st column), denoised features $\hat{X}_0$ (2nd column), initial embedded features $\tilde{X}_{i=0}$ (3rd column), learnt representation $\tilde{X}_i$ at iterations $i = 1$ and $i = 2$ (4th & 5th column, respectively). Panel (b) plots the corresponding graphs $\mathcal{G}_0$, $\hat{\mathcal{G}}_0$ and $\tilde{\mathcal{G}}_i$ (from left to right). Panel (c) shows the graph after edges with weight smaller than 0.3 times the strongest edge have been removed (dark blue corresponds with small weights and yellow corresponds with large weights, respectively).

CES-VCP).

Overall, this leads to conclude that the proposed CES-DGE approach is effective in learning event representations for image sequences and yields robust segmentation results. These results are even more remarkable considering that CES-DGE learns the feature representation from the image sequence itself, without relying on any training dataset.

4.4 Comparative results for EDUBSeg-Desc dataset

Table 4 summarizes the event boundary detection performance of the proposed CES-DGE approach and of CES-VCP for the larger EDUBSeg-Desc dataset. Since CES-VCP reported state-of-the-art results with F-score values 16 percentage points above other methods, cf., Table 2, we omit comparison with other methods in what follows for space reasons. The same (hyper)parameter values as for EDUBSeg are used, without any modification. It can be observed that the performance for both CES-VCP and CES-

DGE are inferior to those reported for the EDUBSeg dataset in the previous section, for all tolerance values. These results corroborate that EDUBSeg-Desc contains more difficult image sequences than EDUBSeg (see Section 4.1 and [20]). Interestingly, while the F-scores achieved by CES-VCP are up to 12 percentage points (and more than 11 percentage points on average) below that reported for EDUBSeg, the F-scores of the proposed CES-DGE approach are at worst 5 percentage points smaller. In other words, CES-DGE yields up to 8 percentage points (on average 6 percentage points) better F-score values than the state of the art for the EDUBSeg-Desc dataset. Our CES-DGE also yields systematically better Rec and Prec values, for all levels of tolerance. Overall, these findings corroborate those obtained for the EDUBSeg dataset and confirm the excellent practical performance of the proposed approach. In particular, the results suggest that the proposed approach effectively avoids domain adaptation problems since it does not rely on a training dataset. It is interesting to note that CES-TSC also yields quite robust results (F-score/Rec/Prec values 0.63/0.60/0.70 for $\tau = 5$) for similar reasons, yet with performance significantly below our CES-DGE.

4.5 Ablation study

Graph initialization. We investigate performance obtained by applying the boundary detector to the features obtained at different stages of our method. First, the original features X (denoted CES-raw) and the features $\hat{X}_0$ obtained by applying NLmeans on the temporal dimension (denoted CES-NLmeans-1D), both of dimension $n = 2048$. Second, the initial embedded features $\tilde{X}_0$ (denoted CES-Embedding) and the features obtained after running the DGE main loop for $K = 2$ iterations (denoted CES-DGE), both of dimension $d = 15$. The results obtained for the EDUBSeg dataset are reported in Table 5. They indicate that CES-NLmeans-1D increases F-score by 8 percentage points w.r.t. CES-raw, and CES-Embedding adds another 1 percentage points in F-score. This confirms that the graph initialization and the reduction of the dimension of the graph representation is beneficial. CES-DGE gains an additional 9 percentage points in F-score value, hence significantly improves upon this initial embedding. An illustration of the effect of the graph initialization and of the DGE steps for EDUBSeg Subject 1 Day 3 is provided in Figure 4. It can be observed how the boundaries between temporally adjacent frames along the diagonal in the graph are successively enhanced as the original features X (column 1) are first replaced with the denoised version $\hat{X}_0$ (column 2), then with the embedded features $\tilde{X}_0$ (column 3), and finally with the DGE representation estimates (columns 4 & 5 for DGE iterations $i = 1, 2$, respectively). Moreover, the boundaries of off-diagonal blocks, which indicate frames at different temporal locations that presumably belong to the same community, are sharpened.

DGE core operations. In Table 6, we report the performance that is obtained on the EDUBSeg dataset when the different operations in the DGE core iterations are removed one-by-one by setting the respective parameter to zero: graph embedding update regularization (α), edge local averaging (p), temporal edge weighting (η), and extra-cluster penalization (μ). It is observed that the overall DGE F-score drops by 1 to 3 percentage points when one single of these operations is deactivated (versus a drop of 9 percentage points from 0.70 to 0.61 when no DGE operation is performed at all, as discussed in the previous paragraph). The fact that removing any of the operations individually does not lead to a knock-out of the DGE loop suggests that the associated individual (temporal & semantic) model assumptions are all and independently important. Among all operations, the largest individual F-score drop (3 percentage points) corresponds with deactivating the extra-cluster penalization (i.e., $\mu = 0$). This points to the essential role of semantic similarity in the graph model. The graph temporal edge weighting is also effective for encoding the temporal prior (2 percentage points F-score drop if deactivated, i.e., $\eta = 0$). The smallest F-score difference (1 percentage point) is associated with edge local averaging (i.e., $p = 0$). To improve this additional temporal regularization step, future work could use nonlocal instead of local averaging, or learnt kernels.

4.6 Generalization capabilities

Human Motion Segmentation. To study the generalization capabilities of DGE to learn representations suitable for temporal segmentation beyond first-person image sequences, we applied it to two well established benchmark datasets for Human Motion Segmentation (HMS), Keck [64] and MAD [65]. These datasets consist of 3rd person shots with static background of a single person acting short motions, captured at high temporal resolution. Here, events correspond with typical motions (e.g., jumping, walking), and our approach models frames corresponding with a sequence of body poses of a motion (e.g., squad+standing+squad+. . .) as semantically related. The Keck and MAD datasets have been chosen among the four HMS benchmarks used by the state-of-the-art works because they are considered particularly challenging due to variable background (Keck) and large number of motions and subjects (MAD), respectively [66, 67]. The state of the art for HMS has been reported in [66] using a Low Rank Transfer subspace (LRT) model, and very recently in [67] using a Multi-mutual transfer subspace learning (MTS) model. We use the same HOG features as therein[1]. Unlike [66] and [67], we use the same set of parameters for both datasets. Moreover, with respect to the previous sections, we only updated the embedding dimension and temporal prior (to $d = 35$, and $p = 50$, using grid search) and maintain the same values as above for the remaining hyperparameters ($K = 2$, $N_C = 10$, $\alpha = 0.1$, $\eta = 0.3$, $\mu = 0.1$). Results are reported in Table 7 in terms of Clustering Accuracy (ACC) and Normalized Mutual Information (NMI). Our DGE achieves state-of-the-art performance also for this different problem. This shows that it is a general framework that can be used in other contexts.

Computational cost. DGE has complexity $\mathcal{O}\left(N^2\right)$ since all of the operations in Algorithm 1 have at most that complexity. To give a concrete example of execution time, it required ∼2 minutes on a standard Dell Precision 7920 Tower with one NVIDIA Titan XP to process the EDUB-Seg dataset (20 sequences with a total of $18,735$ frames).

[1] https://github.com/wanglichenxj/Low-Rank-Transfer-Human-Motion-Segmentation

5 Conclusion

This paper proposed a novel approach to learn representations of events in low temporal resolution image sequences, named Dynamic Graph Embedding (DGE). Unlike state-of-the-art work, which requires (large) datasets for training the model, DGE operates without any training set and learns the temporal event representation for an image sequence directly from the sequence itself.To this end, we introduced an original model based on the assumption that the sequence can be represented as a graph that captures both the temporal and the semantic similarity of the images, which is understood here as similarity in terms of high-level visual features. The key novelty of our DGE approach is then to learn the structure of this unknown underlying graph jointly with a low-dimensional graph embedding. Experimental results have shown that DGE yields robust and effective event representations for temporal segmentation. It outperforms the state of the art in terms of event boundary detection precision, improving F-score values by 1 and 8 percentage points on the EDUBSeg and EDUBSeg-Desc event segmentation benchmark datasets, respectively. Moreover, we showed the generalization capabilities of the proposed DGE to the problem of Human Motion Segmentation. Future work will include exploring the use of more sophisticated methods than the k-means algorithm in the semantic similarity estimation step, and the study of extensions and applications in the field of video analysis such as video and motion segmentation, action detection, and action proposal generation.

References

[1] I. Koprinska and S. Carrato, “Temporal video segmentation: A survey,” *Signal Processing: Image Communication*, vol. 16, no. 5, pp. 477–500, 2001.

[2] A. G. Money and H. Agius, “Video summarisation: A conceptual framework and survey of the

state of the art," *Journal of Visual Communication and Image Representation*, vol. 19, no. 2, pp. 121–143, 2008.

[3] A. Garcia del Molino, C. Tan, J.-H. Lim, and A.-H. Tan, "Summarization of egocentric videos: A comprehensive survey," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 1, pp. 65–76, 2017.

[4] W.-H. Lin and A. Hauptmann, "Structuring continuous video recordings of everyday life using time-constrained clustering," in *Multimedia Content Analysis, Management, and Retrieval 2006*, vol. 6073. International Society for Optics and Photonics, 2006, p. 60730D.

[5] A. R. Doherty, C. Ó Conaire, M. Blighe, A. F. Smeaton, and N. E. O'Connor, "Combining image descriptors to effectively retrieve events from visual lifelogs," in *Proc. 1st ACM Int. Conf. on Multimedia information retrieval*, 2008, pp. 10–17.

[6] A. R. Doherty, D. Byrne, A. F. Smeaton, G. J. Jones, and M. Hughes, "Investigating keyframe selection methods in the novel domain of passively captured visual lifelogs," in *Proc. Int. Conf. on Content-based image and video retrieval*, 2008, pp. 259–268.

[7] E. Talavera, M. Dimiccoli, M. Bolanos, M. Aghaei, and P. Radeva, "R-clustering for egocentric video segmentation," in *Proc. Iberian Conf. Pattern Recognition and Image Analysis*, 2015, pp. 327–336.

[8] F. Paci, L. Baraldi, G. Serra, R. Cucchiara, and L. Benini, "Context change detection for an ultra-low power low-resolution ego-vision imager," in *Proc. IEEE European Conf. Computer Vision Workshops (ECCVW)*, 2016, pp. 589–602.

[9] J. Lin, A. G. del Molino, Q. Xu, F. Fang, V. Subbaraju, and J.-H. Lim, "Vci2r at the ntcir-13 lifelog semantic access task," *Proc. NTCIR-13, Tokyo, Japan*, 2017.

[10] S. Yamamoto, T. Nishimura, Y. Akagi, Y. Takimoto, T. Inoue, and H. Toda, "Pbg at the ntcir-13 lifelog-2 lat, lsat, and lest tasks," *Proc. NTCIR-13, Tokyo, Japan*, 2017.

[11] M. Dimiccoli, M. Bolaños, E. Talavera, M. Aghaei, S. G. Nikolov, and P. Radeva, "Sr-clustering: Semantic regularized clustering for egocentric photo streams segmentation," *Computer Vision and Image Understanding*, vol. 155, pp. 55–69, 2017.

[12] A. Garcia del Molino, J.-H. Lim, and A.-H. Tan, "Predicting visual context for unsupervised event segmentation in continuous photo-streams," in *Proc. 26th ACM Int. Conf. on Multimedia*. ACM, 2018, pp. 10–17.

[13] C. Dias and M. Dimiccoli, "Learning event representations by encoding the temporal context," in *Proc. IEEE European Conf. on Computer Vision Workshops (ECCVW)*, 2018, pp. 587–596.

[14] M. Dimiccoli and H. Wendt, "Enhancing temporal segmentation by nonlocal self-similarity," in *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, Taipei, Taiwan, 2019.

[15] G. Palou and P. Salembier, "Hierarchical video representation with trajectory binary partition tree," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2099–2106.

[16] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 4489–4497.

[17] J. Wang, J. Jiao, L. Bao, S. He, Y. Liu, and W. Liu, "Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4006–4015.

[18] M. Bolanos, M. Dimiccoli, and P. Radeva, "Toward storytelling from visual lifelogging:

An overview," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 1, pp. 77–90, 2017.

[19] A. C. Schapiro, T. T. Rogers, N. I. Cordova, N. B. Turk-Browne, and M. M. Botvinick, "Neural representations of events arise from temporal community structure," *Nature Neuroscience*, vol. 16, no. 4, p. 486, 2013.

[20] M. Bolaños, Á. Peris, F. Casacuberta, S. Soler, and P. Radeva, "Egocentric video description based on temporally-linked sequences," *Journal of Visual Communication and Image Representation*, vol. 50, pp. 205–216, 2018.

[21] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[22] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein, "Learning shape correspondence with anisotropic convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 3189–3197.

[23] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst, "Geodesic convolutional neural networks on riemannian manifolds," in *Proc. of the IEEE international conference on computer vision workshops (ICCVW)*, 2015, pp. 37–45.

[24] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[25] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2016, pp. 2014–2023.

[26] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2016, pp. 3844–3852.

[27] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *IEEE Data Engineering Bulletin*, 2017.

[28] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *Proc. 22nd Int. Conf. on World Wide Web*. ACM, 2013, pp. 37–48.

[29] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, 2016, pp. 1105–1114.

[30] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *Proc. 24th ACM Int. Conf. on information and knowledge management*, 2015, pp. 891–900.

[31] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, 2014, pp. 701–710.

[32] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, 2016, pp. 855–864.

[33] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proc. 24th Int. Conf. on World Wide Web*, 2015, pp. 1067–1077.

[34] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, "Harp: Hierarchical representation learning for networks," in *Proc. 32nd AAAI Conf. on Artificial Intelligence*, 2018.

[35] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proc. 30th AAAI Conf. on Artificial Intelligence*, 2016.

[36] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, 2016, pp. 1225–1234.

[37] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[38] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 1024–1034.

[39] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*, 2015.

[40] V. Garcia and J. Bruna, "Few-shot learning with graph neural networks," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2018.

[41] M. Yeung, B.-L. Yeo, and B. Liu, "Segmentation of video by clustering and graph analysis," *Computer vision and image understanding*, vol. 71, no. 1, pp. 94–109, 1998.

[42] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, and B. Zhang, "A formal study of shot boundary detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 2, pp. 168–186, 2007.

[43] V. Chasanis, A. Kalogeratos, and A. Likas, "Movie segmentation into scenes and chapters using locally weighted bag of visual words," in *Proc. ACM Int. Conf. on Image and Video Retrieval*, 2009, p. 35.

[44] N. Liu, Y. Zhao, Z. Zhu, and H. Lu, "Exploiting visual-audio-textual characteristics for automatic tv commercial block detection and segmentation," *IEEE Transactions on Multimedia*, vol. 13, no. 5, pp. 961–973, 2011.

[45] C. Liu, D. Wang, J. Zhu, and B. Zhang, "Learning a contextual multi-thread model for movie/tv scene segmentation," *IEEE Transactions on Multimedia*, vol. 15, no. 4, pp. 884–897, 2013.

[46] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A survey on visual content-based video indexing and retrieval," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 797–819, 2011.

[47] K. Tang, L. Fei-Fei, and D. Koller, "Learning latent temporal structure for complex event detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1250–1257.

[48] L. H. Iwan and J. A. Thom, "Temporal video segmentation: detecting the end-of-act in circus performance videos," *Multimedia Tools and Applications*, vol. 76, no. 1, pp. 1379–1401, 2017.

[49] V. Bettadapura, D. Castro, and I. Essa, "Discovering picturesque highlights from egocentric vacation videos," in *Proc. IEEE Winter Conf. Applications of Computer Vision (WACV)*, 2016, pp. 1–9.

[50] J. Xu, L. Mukherjee, Y. Li, J. Warner, J. M. Rehg, and V. Singh, "Gaze-enabled egocentric video summarization via constrained submodular maximization," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 2235–2244.

[51] S. Huang, W. Wang, S. He, and R. W. Lau, "Egocentric temporal action proposals," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 764–777, 2017.

[52] E. H. Spriggs, F. De La Torre, and M. Hebert, "Temporal segmentation and activity classification from first-person sensing," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2009, pp. 17–24.

[53] Y. J. Lee and K. Grauman, "Predicting important objects for egocentric video summarization," *International Journal of Computer Vision*, vol. 114, no. 1, pp. 38–55, 2015.

[54] Y. Poleg, C. Arora, and S. Peleg, "Temporal segmentation of egocentric videos," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2537–2544.

[55] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2005, pp. 60–65.

[56] G. B. Giannakis, Y. Shen, and G. V. Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 787–807, 2018.

[57] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988.

[58] X. Wang, Y. Tang, S. Masnou, and L. Chen, "A global/local affinity graph for image segmentation," *IEEE Transactions on Image Processing*, vol. 24, no. 4, pp. 1399–1411, 2015.

[59] D.-T. Dang-Nguyen, L. Piras, M. Riegler, G. Boato, L. Zhou, and C. Gurrin, "Overview of imagecleflifelog 2017: Lifelog retrieval and summarization." in *CLEF (Working Notes)*, 2017.

[60] C. Gurrin, H. Joho, F. Hopfgartner, L. Zhou, R. Gupta, R. Albatal, D. Nguyen, and D. Tien, "Overview of NTCIR-13 lifelog-2 task," in *Proc. 13th Conf. on Evaluation of Information Access Technology*, 2017.

[61] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Computer vision and pattern recognition (CVPR)*, 2016, pp. 2818–2826.

[62] D. Potapov, M. Douze, Z. Harchaoui, and C. Schmid, "Category-specific video summarization," in *Proc. European Conf. on Computer Vision (ECCV)*, 2014, pp. 540–555.

[63] S. Li, K. Li, and Y. Fu, "Temporal subspace clustering for human motion segmentation," in *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 4453–4461.

[64] Z. Lin, Z. Jiang, and L. S. Davis, "Recognizing actions by shape-motion prototype trees," in *Proc. of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2009, pp. 444–451.

[65] D. Huang, S. Yao, Y. Wang, and F. De La Torre, "Sequential max-margin event detectors," in *Proc. European Conf. on Computer Vision (ECCV)*. Springer, 2014, pp. 410–424.

[66] L. Wang, Z. Ding, and Y. Fu, "Low-rank transfer human motion segmentation," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 1023–1034, 2018.

[67] T. Zhou, H. Fu, C. Gong, J. Shen, L. Shao, and F. Porikli, "Multi-mutual consistency induced transfer subspace learning for human motion segmentation," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10 277–10 286.

ALGORITHM 1: Dynamic Graph Embedding (DGE)

```
        $N$ − *length of the image sequence*
        $n$ − *original feature dimension*
        $d$ − *embedding feature dimension* ($d \ll n$)
**Input**  : $X \in \mathbb{R}^N \times \mathbb{R}^n$          *initial feature matrix*
**Output**: $\tilde{X} \in \mathbb{R}^N \times \mathbb{R}^d$          *graph embedded feature matrix*
/* Graph initialization          Eqs. (1-3)          */
  $\hat{X}_0 = \mathrm{NLmeans}^{1D}(X) \in \mathbb{R}^N \times \mathbb{R}^n$          *denoise initial features*
  $\mathrm{G}_0 = S_{\tilde{l}}(\hat{X}_0) \in \mathbb{R}^N \times \mathbb{R}^N$ *initialize graph in original space*
/* Graph embedding initialization Eqs. (3-4) */
  $\tilde{X} = \mathrm{PCA}_d(\hat{X}_0) \in \mathbb{R}^N \times \mathbb{R}^d$
  $\tilde{X}_0 = \mathrm{argmin}_{\tilde{X}}\ \mathcal{L}(S_{\tilde{l}}(\tilde{X}), \mathrm{G}_0)$          *initialize embedding features*
  $\tilde{\mathrm{G}}_0 = S_{\tilde{l}}(\tilde{X}_0) \in \mathbb{R}^N \times \mathbb{R}^N$          *initialize graph in embedding space*
/* DGE core loop          Eqs. (5-7)          */
**for** $i \leftarrow 1$ **to** $K$ **do**
    — **graph embedding update**
    $\tilde{X}_i = \mathrm{argmin}_{\tilde{X}}(1-\alpha)\mathcal{L}_1(S_{\tilde{l}}(\tilde{X}), \tilde{\mathrm{G}}_{i-1}) + \alpha\mathcal{L}_2(S_{\tilde{l}}(\tilde{X}), \mathrm{G}_0)$
        *update embedding features given current graph* $\tilde{\mathcal{G}}_{i-1}$
    — **graph structure update: temporal prior**
    $\tilde{\mathrm{G}}_i \leftarrow S_{\tilde{l}}(\tilde{X}_i) * \mathcal{K}_p$          *local average of weights of graph of* $\tilde{X}_i$
    $\tilde{\mathrm{G}}_i \leftarrow f_\eta(\tilde{\mathrm{G}}_i)$          *strengthen temporally adjacent edges*
    — **graph structure update: semantic prior**
    $\mathcal{C} = \mathrm{kmeans}(\tilde{X}_i; N_C)$          *estimate semantic communities*
    $\tilde{\mathrm{G}}_i = g_\mu(\tilde{\mathrm{G}}_i, \mathcal{C})$          *encode semantic similarity in graph*
**end**
```

Parameter	**F-score**				
d	5	7	10	15	20
	0.59	0.61	0.65	**0.70**	0.69
(embedding dimension)	25	35	45	100	150
	0.68	0.67	0.68	0.65	0.64
K	1	2	3	4	5
(DGE iterations)	0.66	**0.70**	0.68	0.67	0.66
α	0.05	0.1	0.2	0.3	0.4
(graph embedding update)	0.679	**0.70**	0.69	0.69	0.68
p	2	3	4	5	6
(2D local average size)	0.69	**0.70**	0.69	0.69	0.68
η	0.01	0.1	0.3	0.4	0.6
(graph temporal regularization)	0.69	0.69	**0.70**	0.69	0.67
μ	0.03	0.05	0.1	0.2	0.3
(extra-cluster penalty)	0.68	0.70	**0.70**	0.69	0.67
N_C	3	4	6	8	10
	0.65	0.66	0.67	0.68	**0.70**
(cluster number)	12	14	16	18	20
	0.69	0.69	0.68	0.67	0.68

Table 1: Robustness of CES-DGE with respect to hyperparameter values (EDUBSeg, tolerance $\tau = 5$): F-scores obtained when varying the DGE hyperparameter indicated in the first column, with all others held fixed (best results in bold).

Method	**F-score**	**Rec**	**Prec**
k-Means smoothed	0.51	0.39	0.82
AC-color	0.38	0.25	0.90
SR-ClusteringCNN	0.53	0.68	0.49
KTS	0.53	0.40	0.87
CES-TSC	0.66	0.69	0.67
CES-VCP	0.69	**0.77**	0.66
CES-DGE	**0.70**	0.70	**0.72**
Manual segmentation	**0.72**	0.68	**0.80**

Table 2: Comparison of CES-DGE with state-of-the-art methods & manual segmentation for EDUBSeg and tolerance $\tau = 5$.

Method	CES-VCP			CES-DGE		
Tolerance	**F-score**	**Rec**	**Prec**	**F-score**	**Rec**	**Prec**
$\tau = 5$	0.69	**0.77**	0.66	**0.70**	0.70	**0.72**
$\tau = 4$	0.67	**0.75**	0.63	**0.68**	0.69	**0.70**
$\tau = 3$	0.64	0.62	**0.71**	**0.65**	**0.64**	0.68
$\tau = 2$	**0.59**	**0.67**	0.56	**0.59**	0.59	**0.61**
$\tau = 1$	0.44	0.44	0.49	**0.48**	**0.48**	**0.50**

Table 3: Comparison of CES-DGE with state-of-the-art CES-VCP for different values of tolerance for EDUBSeg.

Method	CES-VCP			CES-DGE		
Tolerance	**F-score**	**Rec**	**Prec**	**F-score**	**Rec**	**Prec**
$\tau = 5$	0.57	0.59	0.60	**0.65**	**0.67**	**0.65**
$\tau = 4$	0.56	0.58	0.58	**0.63**	**0.66**	**0.63**
$\tau = 3$	0.52	0.54	0.54	**0.60**	**0.62**	**0.60**
$\tau = 2$	0.49	0.50	0.50	**0.54**	**0.56**	**0.54**
$\tau = 1$	0.43	0.44	**0.45**	**0.45**	**0.46**	**0.45**

Table 4: Comparison of CES-DGE with state-of-the-art CES-VCP for different values of tolerance for EDUBSeg-Desc.

Method	**F-score**	**Rec**	**Prec**
CES-raw	0.52	0.56	0.56
CES-NLmeans-1D	0.60	0.63	0.61
CES-Embedding	0.61	0.61	0.65
CES-DGE	0.70	0.70	0.72

Table 5: CES-DGE ablation study for EDUBSeg and tolerance $\tau = 5$.

Deactivated DGE parameter	$\alpha = 0$	$p = 0$	$\eta = 0$	$\mu = 0$
F-score	0.69	0.69	0.68	0.67
difference with full DGE (0.70)	-0.01	-0.01	-0.02	-0.03

Table 6: F-scores obtained when single core steps are removed from DGE (indicated by a zero value for the respective parameter).

Dataset	Keck		MAD	
Method	**ACC**	**NMI**	**ACC**	**NMI**
TSC	0.48	0.71	0.56	0.77
LRT	0.55	0.82	0.60	0.82
MTS	0.60	**0.83**	0.62	**0.83**
DGE	**0.72**	**0.83**	**0.67**	0.82

Table 7: Comparison with state of the art for Human Motion Segmentation for the Keck and MAD benchmark datasets (baselines taken from [66] and [67]).