Cut-Based Graph Learning Networks to Discover Compositional Structure of Sequential Video Data

Kyoung-Woon On,[1] Eun-Sol Kim,[2] Yu-Jung Heo,[1] Byoung-Tak Zhang[1]
[1]Department of Computer Science and Engineering
Seoul National University, Seoul, South Korea
[2]Kakao Brain, Seongnam, South Korea
epsilon.kim@kakaobrain.com, {kwon, yjheo, btzhang}@bi.snu.ac.kr

Abstract

Conventional sequential learning methods such as Recurrent Neural Networks (RNNs) focus on interactions between consecutive inputs, i.e. first-order Markovian dependency. However, most of sequential data, as seen with videos, have complex dependency structures that imply variable-length semantic flows and their compositions, and those are hard to be captured by conventional methods. Here, we propose Cut-Based Graph Learning Networks (CB-GLNs) for learning video data by discovering these complex structures of the video. The CB-GLNs represent video data as a graph, with nodes and edges corresponding to frames of the video and their dependencies respectively. The CB-GLNs find compositional dependencies of the data in multilevel graph forms via a parameterized kernel with graph-cut and a message passing framework. We evaluate the proposed method on the two different tasks for video understanding: Video theme classification (Youtube-8M dataset (Abu-El-Haija et al. 2016)) and Video Question and Answering (TVQA dataset (Lei et al. 2018)). The experimental results show that our model efficiently learns the semantic compositional structure of video data. Furthermore, our model achieves the highest performance in comparison to other baseline methods.

Introduction

One of the fundamental problems in learning sequential video data is to find semantic structures underlying sequences for better representation learning. As most semantic flows cannot be modeled with simple temporal inductive biases, i.e., Markov dependencies, it is crucial to find the complex temporal semantic structures from long sequences to understand the video data. We believe there are two ingredients to solving this problem: segmenting the whole long-length sequence into (multiple) semantic units and finding their compositional structures. In this work, we propose a new graph-based method which can discover composite semantic flows in video inputs and utilize them for representation learning of videos. The compositional semantic structures are defined as multilevel graph forms, which make it possible to find long-length dependencies and their hierarchical relationships effectively.

In terms of modeling sequential semantic flows, the related work can be summarized with the following three categories: neural networks, clustering, and self-attention based methods.

In terms of neural network architectures, many problems involving sequential inputs are resolved by using Recurrent Neural Networks (RNNs) as the networks naturally take sequential inputs frame by frame. However, as RNN-based methods take frames in (incremental) order, the parameters of methods are trained to capture patterns in transition between successive frames. This makes it hard to find long-term dependencies through overall frames. To consider the long-term dependency, Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) and Gated Recurrent Units (GRU) (Chung et al. 2014) introduced switches to the RNN structures and Hierarchical RNN (Chung, Ahn, and Bengio 2018) stacked multiple layers to find hierarchical structures. Even though those methods ignore noisy (unnecessary) frames and maintain the semantic flow through the whole sequence, it is still hard for RNN variants to retain multiple semantic flows and to learn their hierarchical and compositional relationships. Interestingly, Convolutional Neural Network (CNN) based methods, such as ByteNet (Kalchbrenner et al. 2016), ConvS2S (Gehring et al. 2017) and WaveNet (Oord et al. 2016), applied 1D convolution operator to temporal sequences for modeling dependencies of adjacent frames and their compositions. However these operators hardly captured variable-length dependencies which play a significant role as semantic units.

Recent researches revisited the traditional idea of clustering successive frames into representative clusters. Deep Bag of Frame (DBoF) (Abu-El-Haija et al. 2016) randomly selects k frames from whole sequences as the representatives. NetVLAD (Arandjelovic et al. 2016) divides all features into k clusters and calculates residuals, which are difference vectors between the feature vectors and their corresponding cluster center, as the new representations for each feature vector. Even though the main idea of this type of research is quite simple, it helps to understand the semantics of long sequences by focusing on a small number of representative frames. However, it is hard to consider the complex temporal relationships.

Along with enormous interest of attention mechanism, a number of significant researches (Vaswani et al. 2017; Devlin et al. 2019) have been aimed to understanding the long (sequential) inputs relying purely on self-attention mechanisms. With real-world applications on natural language understanding, such as question and answering (QA) and paragraph detection, those methods focus on meaningful words (or phrases, sentences) among the long reading passages and ignore irrelevant words to the passage by stacking multiple attention layers. These methods consist of a large number of layers with a huge number of parameters and require a huge amount of training dataset.

In this paper, we propose a method to learn representations of video by discovering the compositional structure in multilevel graph forms. A single video data input is represented as a graph, where nodes and edges represent frames of the video and relationships between all node pairs. From the input representations, the Cut-Based Graph Learning Networks (CB-GLNs) find temporal structures in the graphs with two key operations: temporally constrained normalized graph-cut and message-passing on the graph. A set of semantic units is found by parameterized kernel and cutting operations, then representations of the inputs are updated by message passing operations.

We thoroughly evaluate our method on the large-scale video theme classification task, YouTube-8M dataset (Abu-El-Haija et al. 2016). As a qualitative analysis of the proposed model, we visualize compositional semantic dependencies of sequential input frames, which are automatically constructed. Quantitatively, the proposed method shows a significant improvement on classification performance over the baseline models. Furthermore, as an extension of our work, we apply the CB-GLNs to another video understanding task, Video Question and Answering on TVQA dataset (Lei et al. 2018). With this experiment, we show how the CB-GLNs can fit into other essential components such as attention mechanism, and demonstrate the effectiveness of the structural representations of our model.

The remainder of the paper is organized as follows. As preliminaries, basic concepts of Graph Neural Networks (GNNs) and graph-cut mechanisms are introduced. Next, the problem statement of this paper is described to make further discussion clear. After that, the proposed Cut-Based Graph Learning Networks (CB-GLNs) are suggested in detail and the experimental results with the real datasets, YouTube-8M and TVQA are presented.

Preliminaries

In this section, basic concepts related to graphs are summarized. First, mathematical definitions and notations of graphs are clarified. Second, the normalized graph-cut method is described. Lastly, variants of Graph Neural Networks (GNNs) are introduced.

Graph notations

A graph G is denoted as a pair (V, E) with $V = \{v_1, ..., v_N\}$ the set of nodes (vertices), and $E \in V \times V$ the set of edges. Each node v_i is associated with a feature vector $x_i \in \mathbb{R}^m$. To make notation more compact, the feature matrix of graph G is denoted as $X = [x_1, x_2, ..., x_N]^\top \in \mathbb{R}^{N \times m}$. Also, a graph has an N-by-N weighted adjacency matrix A where $A_{i,j}$ represents the weight of the edge between v_i and v_j.

Normalized graph-cut algorithm

A graph $G = (V, E)$ can be partitioned into two disjoint sets V_1, V_2 by removing edges connecting the two parts[1]. The partitioning cost is defined as the total weight of the edges that have been removed. In addition to the cut cost, the normalized graph-cut method (Shi and Malik 2000) considers the total edge weight connecting a partition with the entire graph (the degree of the partition) to avoid the trivial solutions which can make extremely imbalanced clusters. The objective of the normalized graph-cut can be formally described as follows.

$$Ncut(V_1, V_2) = \frac{cut(V_1, V_2)}{assoc(V_1, V)} + \frac{cut(V_1, V_2)}{assoc(V_2, V)} \tag{1}$$

with

$$cut(V_1, V_2) = \sum_{v_1 \in V_1, v_2 \in V_2} w(v_1, v_2) \tag{2}$$

$$assoc(V_1, V) = \sum_{v_1 \in V_1, v \in V} w(v_1, v) \tag{3}$$

where $w(v_1, v_2)$ is an edge weight value between node v_1 and v_2. It is formulated as a discrete optimization problem and usually relaxed to continuous, which can be solved by eigenvalue problem with the $O(n^2)$ time complexity. By applying the cut method recursively, an input graph is divided into fine-grained sub-graphs.

Graph Neural Networks (GNNs)

Since the first proposed by (Gori, Monfardini, and Scarselli 2005), interest in combining deep learning and structured approaches has steadily increased. This has led to various graph-based neural networks being proposed over the years.

Based on spectral graph theory (Chung and Graham 1997), spectral approaches which convert the graph to the spectral domain and apply the convolution kernel of the graph were proposed (Bruna et al. 2013; Henaff, Bruna, and LeCun 2015; Kipf and Welling 2016). (Gilmer et al. 2017) suggested the message passing neural networks (MPNNs), which encompass a number of previous neural models for graphs under a differentiable message passing interpretation.

In detail, the MPNNs have two phases, a message passing phase and a update phase. In the message passing phase, the message for vertex v in l-th layer is defined in terms of message function M^l:

$$m_v^l = \sum_{w \in N(v)} M^l(h_v, h_w) \tag{4}$$

[1] $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \emptyset$

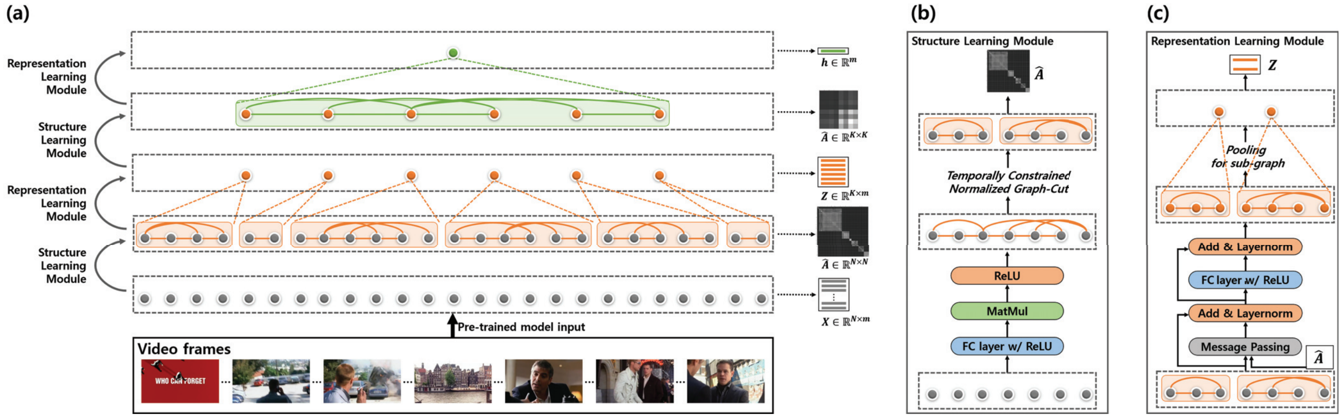
(a)
Representation Learning Module
Structure Learning Module
Representation Learning Module
Structure Learning Module
$h \in \mathbb{R}^m$
$\hat{A} \in \mathbb{R}^{K\times K}$
$Z \in \mathbb{R}^{K\times m}$
$\hat{A} \in \mathbb{R}^{N\times N}$
$X \in \mathbb{R}^{N\times m}$
Pre-trained model input
Video frames
WHO CAN FORGET
(b)
Structure Learning Module
$\hat{A}$
Temporally Constrained Normalized Graph-Cut
ReLU
MatMul
FC layer w/ ReLU
(c)
Representation Learning Module
Z
Pooling for sub-graph
Add & Layernorm
FC layer w/ ReLU
Add & Layernorm
Message Passing
$\hat{A}$

Figure 1: (a): The overall architecture of the Cut-Based Graph Learning Networks (CB-GLNs) for a video representation learning task. (b), (c): sophisticated illustrations of Structure Learning Module and Representation Learning Module.

where in the sum, $N(v)$ denotes the neighbors of v in a graph G. With the message m_v^l, the representations of vertex v in $(l+1)$-th layer are obtained via the update function U^l.

$$h_v^{l+1} = U^l(h_v^l, m_v^l) \tag{5}$$

The message functions M^l and the update functions U^l are differentiable so that the MPNNs can be trained in an end-to-end fashion. As extensions, there are some attempts to have been made to improve message passing by putting a gating or attention mechanism into a message function, which can have computational benefits (Monti et al. 2017; Duan et al. 2017; Hoshen 2017; Veličković et al. 2018; Satorras and Estrach 2018; van Steenkiste et al. 2018).

We further note other previous research for learning a structure of a graph. Neural Relational Inference (NRI) (Kipf et al. 2018) used a Variational Autoencoder (VAE) (Kingma and Welling 2013) to infer the connectivity between nodes with latent variables. Other generative models based approaches also have been well studied (Bojchevski et al. 2018; De Cao and Kipf 2018; Simonovsky and Komodakis 2018). However, those suffer from availability of structural information in training data or have complex training procedures. To our knowledge, it is the first time to suggest graph-cut based neural networks to discover the inherent structure of videos without supervision of the structural information.

Problem Statement

The problem to be tackled in this work can be clearly stated with the notations in the previous section as below.

We consider videos as inputs, and a video is represented as a graph G. The graph G has nodes corresponding to each frame respectively in the video with feature vectors and the dependencies between two nodes are represented with weight values of corresponding edges.

Suppose that video data X has N successive frames and each frame has an m-dimensional feature vector $x \in \mathbb{R}^m$. Each frame corresponds to a node $v \in V$ of graph G, and the dependency between two frames v_i, v_j is represented by a weighted edge $e_{ij} \in E$. From $G = (V, E)$, the dependency structures among video frames are defined as the weighted adjacency matrix A, where $A_{ij} = e_{ij}$. With aforementioned notations and definitions, we can now formally define the problem of video representations learning as follows:

Given the video frames representations $X \in \mathbb{R}^{N\times m}$, we seek to discover a weighted adjacency matrix $A \in \mathbb{R}^{N\times N}$ which represents dependency among frames.

$$f : X \rightarrow A \tag{6}$$

With X and A, final representations for video $h \in \mathbb{R}^l$ are acquired by g.

$$g : \{X, A\} \rightarrow h \tag{7}$$

The obtained video representations h can be used for various tasks of video understanding. In this paper, the video theme classification and the video question and answering tasks are mainly considered.

Cut-Based Graph Learning Networks

The Cut-Based Graph Learning Networks (CB-GLNs) consist of two sub-modules: a structure learning module with the graph-cuts and a representation learning module with message-passing operations. The key idea of the method is to find inherent semantic structures using the graph-cuts and to learn feature vectors of the video with the message-passing algorithm on the semantic structures. Stacking these modules leads to the subsequent discovery of compositional structures in the form of a multilevel graph. Figure 1(a) illustrates the whole structure of the CB-GLNs. In the next sections, operations of each of these modules are described in detail.

Structure Learning Module

In the structure learning module, the dependencies between frames $\hat{A}$ are estimated via parameterized kernels and the temporally constrained graph-cut algorithm.

As the first step, the initial temporal dependencies over all frames are constructed via the parameterized kernel $\mathcal{K}$:

$$\hat{A}_{ij} = \mathcal{K}(x_i, x_j) = ReLU(f(x_i)^\top f(x_j)) \tag{8}$$

where $f(x)$ is a single-layer feed-forward network without non-linear activation.

Then, as the second step, the meaningful dependency structure among all pairwise relationships is refined by applying normalized graph-cut algorithm to the $\hat{A}$. The objective of the normalized graph-cut for CB-GLNs is:

$$Ncut(V_1, V_2) = \frac{\sum_{v_i \in V_1, v_j \in V_2} \hat{A}_{ij}}{\sum_{v_i \in V_1} \hat{A}_{i\cdot}} + \frac{\sum_{v_i \in V_1, v_j \in V_2} \hat{A}_{ij}}{\sum_{v_j \in V_2} \hat{A}_{j\cdot}} \tag{9}$$

To reduce the complexity of the equation (9) and to keep the inherent characteristics of the video data, an additional constraint is added to the graph-cut algorithm. As the video data is composed of time continuous sub-sequences, no two partitioned sub-graphs have an overlap in physical time. This characteristic is implemented by applying the temporal constraint (Rasheed and Shah 2005; Sakarya and Telatar 2008) as follows.

$$(i < j \text{ or } i > j) \quad \text{for all } v_i \in V_1, v_j \in V_2 \tag{10}$$

Thus, a cut can only be made along the temporal axis and complexity of the graph partitioning is reduced to linear time while keeping the characteristics of the video data. Also, as the gradients can flow through the surviving edges, it is end-to-end trainable.

The graph-cut can be recursively applied to the $\hat{A}$, so K partitioned sub-graphs can be obtained. The number of cut operations is determined by the length of the video N, $\lfloor \log_2 \sqrt{N} \rfloor$, and we also add the constraint that sub-cluster should not be partitioned if it is no longer than pre-specified length. Figure 1(b) depicts the detailed operations of the structure learning module.

Representation Learning Module

After estimating the weighted adjacency matrix $\hat{A}$, the representation learning module updates the representations of each frame via a differentiable message-passing framework (Gilmer et al. 2017). For the message function F_M, we simply use the weighted sum of adjacent nodes' representations after linear transformation similar to (Kipf and Welling 2016):

$$M = F_M(X, \hat{A}_{\text{cut}}) = D^{-1}\hat{A}_{\text{cut}} X W_M \tag{11}$$

where D is a degree matrix of the graph and $\hat{A}_{\text{cut}}$ is an adjacency matrix after cut operations.

For the update function F_U, we integrate the message with node representations by using low-rank bilinear pooling B (Kim et al. 2017) followed by a position-wise fully connected network.

$$H = F_U(X, M) = f(B(X, M)) \tag{12}$$

where f is a single-layer position-wise fully connected network. We also employ a residual connection (He et al. 2016) around each layer followed by layer normalization (Ba, Kiros, and Hinton 2016).

Once the representations of all frames are updated, a pooling operation for each partitioned sub-graph is applied. Then we can obtain higher level representations $Z \in \mathbb{R}^{K \times m}$, where K is the number of partitioned sub-graphs (Figure 1(c)). If we have additional information such as query (e.g. a question feature vector in video QA setting), we can pool the sub-graph with attentive pooling similar to (Santos et al. 2016).

In the same way, Z is fed into the new structure learning module and we can get the video-level representation $h \in \mathbb{R}^m$.

Experiments

In this section, the experimental results on the two different video datasets, YouTube-8M (video theme classification) and TVQA (video question and answering), are provided.

Video Theme Classification Task on YouTube-8M

Data specification YouTube-8M (Abu-El-Haija et al. 2016) is a benchmark dataset for video understanding, where the main task is to determine the key topical themes of a video. The dataset consists of 6.1M video clips collected from YouTube. Each video is labeled with one or multiple tags referring to the main topic of the video. Each video is encoded at 1 frame-per-second up to the first 300 seconds. The volume of video data is too large to be treated in it's raw form. As such the input is pre-processed with pretrained models by the author of the dataset.

Global Average Precision (GAP) is used for the evaluation metric for the multi-label classification task as used in the YouTube-8M competition. For each video, 20 labels are predicted with confidence scores, then the GAP score computes the average precision and recall across all of the predictions and all the videos.

Model setup The frame-level visual and audio features are extracted by inception-v3 network (Szegedy et al. 2016) trained on imagenet and VGG-inspired architecture (Hershey et al. 2017) trained for audio classification. These features construct an input feature matrix X of video sequences, then X is fed into a sequence model to extract final video representation h. Including our model, all baseline sequence models are composed of two layers and average pooling is used for final representations. With the representation h, a simple logistic regression is used as a final classifier.

Quantitative results Firstly, we evaluate the classification performance of the proposed model against four types of representative sequential models described in the Introduction section and two state-of-the-art models (Lin, Xiao, and Fan 2018; Mao et al. 2018) previously reported.

The results with GAP score are summarized in Table 1. The proposed model considerably outperforms all the comparative models. The second best performing model is the self-attention based approach, followed by RNNs, CNNs and Clustering based approaches. In the "Qualitative results" Section, automatically constructed compositional structures are discussed for better understanding of the model.

Table 1: Comparison on classification accuracy with the GAP measure on validation dataset. Logistic regression is as a classifier for all of the presented methods.

Frame-level model	GAP
Average pooling	0.7824
DeepBoF (4096 clusters)	0.8079
NetVLAD (256 clusters)	0.8396
1D CNN (2 layers, kernel size 3)	0.8254
1D CNN (2 layers, kernel size 5)	0.8245
1D CNN (2 layers, kernel size 7)	0.8247
LSTM (2 layers)	0.8446
GRU (2 layers)	0.8160
BiLSTM (2 layers)	0.8410
BiGRU (2 layers)	0.8079
Self-Attention (4 heads, 2 layers)	0.8553
NeXtVLAD (Lin, Xiao, and Fan 2018)	0.8499
DCGN (Mao et al. 2018)	0.8450
CB-GLNs (2 layers)	**0.8597**

Table 2: An ablation study of Cut-Based Graph Learning Networks.

Ablation model	GAP
(a) layer normalization	0.8486
(b) residual connection	0.8447
(c) residual connection with layer normalization	0.8370
(d) graph-cut after learning representations	0.8576
CB-GLNs	0.8597

For ablation studies, we selected three critical characteristics of CB-GLNs for in-depth analysis: layer normalization, residual connection and graph-cut after learning representations. The GAP scores on the validation set for each ablation experiments are shown in Table 2. As can be seen from (a) to (c) in the Table 2, the residual connections followed by layer normalization are crucial for the representation learning module. Also, to see the effect of sparsening an adjacency matrix via the graph-cut, reversed order of representation learning and graph-cut is also conducted ((d) in Table 2). By doing so, the representations of each node are updated with $\hat{A}$ obtained only using kernel $\mathcal{K}$ (in Equation 8) and graph-cut algorithm is used just for sub-graph pooling. Thus, the model has to learn representations with dense and noisy connections, degrading the performance of the model. From this result, we can argue that the temporally constrained graph-cut effectively reduce the noisy connections in the graph.

Qualitative results: Learning compositional dependency structure In this section, we demonstrate compositional learning capability of CB-GLNs by analyzing constructed multilevel graphs. To make further discussion clear, four terms are used to describe the compositional semantic flows: semantic units, scenes, sequences and a video for each level. In Figure 2, a real example with the usage of video titled "Rice Pudding[2]" is described to show the results.

In Figure 2(a), the learned adjacency matrices in each layer are visualized in gray-scale images: the two leftmost images are from the 1st layer and the two rightmost images are from the 2nd layer. To denote multilevel semantic flows, four color-coded rectangles (blue, orange, red and green) are marked and those colors are consistent with Figure 2(b).

Along with diagonal elements of the adjacency matrix in the 1st layer (Figure 2(a)-1), a set of semantic units are detected corresponding to bright blocks (blue). Interestingly, we found that each semantic unit contains highly correlated frames. For example, the #1 and #2 are each shots introducing the YouTube cooking channel and how to make rice pudding, respectively. The #4 and #5 are shots showing a recipe of rice pudding and explaining about the various kinds of rice pudding. The #6 and #7 are shots putting ingredients into boiling water in the pot and bringing milk to boil along with other ingredients. At the end of the video clip, #11 is a shot decorating cooked rice pudding and #12 is an outro shot that invites the viewers to subscribe.

These semantic units compose variable-length scenes of the video, and each scene corresponds to a sub-graph obtained via graph-cut (Figure 2(a)-2.). For example, #13 is a scene introducing this cooking channel and rice pudding. Also, #15 is a scene of making rice pudding with detailed step by step instructions, and #16 is an outro scene wrapping up with cooked rice pudding. The 1st-layer of the model updates representations of frame-level nodes with these dependency structures, then aggregates frame-level nodes to form scene-level nodes (Layer 1 in the Figure 2(b)).

In Figure 2(a)-3 and (a)-4, the sequence-level semantic dependencies (red) are shown. #17 denotes a sequence of making rice pudding from beginning to end, which contains much of the information for identifying the topical theme of this video. Finally, the representations of scenes are updated and aggregated to get representations of the whole video (Layer 2 in the Figure2(b)).

Video Question & Answering Task on TVQA

Data specification TVQA (Lei et al. 2018) is a video question and answering dataset on TV show domain. It consists of total 152.5k question-answer pairs on six TV shows: The Big Bang Theory, How I Met Your Mother, Friends, Grey's Anatomy, House and Castle. Also, it contains 21.8k short clips of 60-90 seconds segmented from the original TV show for question-answering. The provided inputs are 3 fps image frames, subtitles and multiple choice questions with 5 candidate answers for each question, for which only one is correct.

The questions in the dataset are localized to a specific subpart in the video clips by restricting questions to a composition of two parts, e.g., "Where was Sheldon sitting / before he spilled the milk?". Models should answer questions using both visual information and associated subtitles from the video.

Model setup The input visual features were extracted by the pooled 2048D feature of the last block of ResNet101 (He et al. 2016) trained on imagenet and the text-based features for subtitles, questions and answers were extracted by

[2] https://youtu.be/cD3enxnS-JY

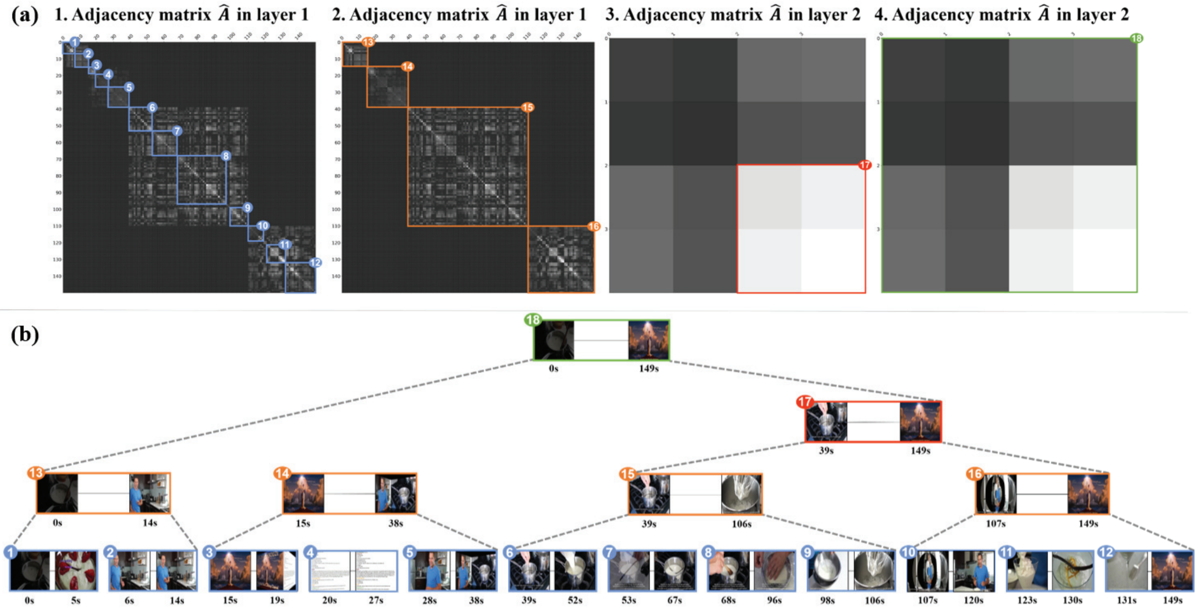
(a)
1. Adjacency matrix $\hat{A}$ in layer 1
2. Adjacency matrix $\hat{A}$ in layer 1
3. Adjacency matrix $\hat{A}$ in layer 2
4. Adjacency matrix $\hat{A}$ in layer 2
(b)
18
0s
149s
17
39s
149s
13
0s
14s
14
15s
38s
15
39s
106s
16
107s
149s
1
0s
5s
2
6s
14s
3
15s
19s
4
20s
27s
5
28s
38s
6
39s
52s
7
53s
67s
8
68s
96s
9
98s
106s
10
107s
120s
11
123s
130s
12
131s
149s

Figure 2: An example of the constructed temporal dependency structures for a real input video in YouTube-8M, titled “Rice Pudding” (https://youtu.be/cD3enxnS-JY) is visualized. The topical themes (true labels) of this video are {Food, Recipe, Cooking, Dish, Dessert, Cake, baking, Cream, Milk, Pudding and Risotto}. (a): Learned adjacency matrices in the layer 1 and 2 are visualized. The strength of connections are encoded in a gray-scale where 1 is white and 0 is black. (a)-1: 12 bright blocks in layer 1 are detected (blue rectangles), each block (highly connected frames) represents a semantic unit. (a)-2: Sub-graphs of the input are denoted by orange rectangles. It shows that semantically meaningful scenes are found by temporally constrained graph-cut. (a)-3 and (a)-4: learned high-level dependency structures in layer 2 are revealed with red and green rectangles. (b): The whole composite temporal dependencies are presented.

GloVe (Pennington, Socher, and Manning 2014). The visual features and subtitle features are manually aligned with time-stamp and answer features also aligned with visual and subtitle features by attention mechanism to construct input X. Then the X is fed into the CB-GLNs to extract final representations h. Different from the YouTube-8M dataset case, we use a question feature vector as a query of attentive pooling, so that the representations of the sequence are pooled via weighted sum with the attention values.

Qualitative results: Attention hierarchy with learned compositional structure In this section, we show how the attention mechanism can be fit into the CB-GLNs to learn representations more effectively. Basically, the attention mechanism places more weight on important parts to aggregate values, given a query. By virtue of the compositionality, the attention mechanism can be naturally applied to the CB-GLNs in a hierarchical fashion. Figure 3 presents learned attention hierarchy in a real video clip of “Friends”. In this example, The question is “What did Chandler say he was going to get, when he got up?” and the answer for the question is that “Chandler said he was going to get cigarettes.”.

In Figure 3(a), we can see that scenes with high attention values (Scene 5 and Scene 6 (coded by a red rectangle)) in layer 2 are aligned well with localized video portions relevant to a given question. Scene 4, where “Chandler is searching his pocket to find cigarettes while sitting down”, is also in the localized section. Therefore, we can say our model finds a sensitive portion of the video relevant to the given question. In layer 1 (Figure 3(a)), the model gives a keener attention to the frame-level within each scenes (coded by orange rectangles), such as a moment where Chandler gets up or Chandler yells ‘I gotta smoke‘.

Because the attention operation for each frame is conducted hierarchically, we can calculate cumulative attention values by multiplying them in layer 1 and layer 2. Figure 3(b) shows the cumulative attention values for each frame. The most important frame in a viewpoint of the model is the ”getting up moment” frame because the model should answer the question by identifying the meaning of “when he got up” in the question.

In Figure 3(c), the learned adjacency matrices in each layer are visualized. Same color-coded rectangles with (a) are used for cut scenes (orange rectangles in layer 1) and scenes with high attention values (a red rectangle in layer 2). We also coded red rectangle in layer 1, which is corresponding to scenes with high attention values in layer 2. Even though scene 5 and scene 6 in the frame level (layer 1) are considerably short when compared to the whole video sequences, the CB-GLNs can find important moments and aggregate them in an effective way.

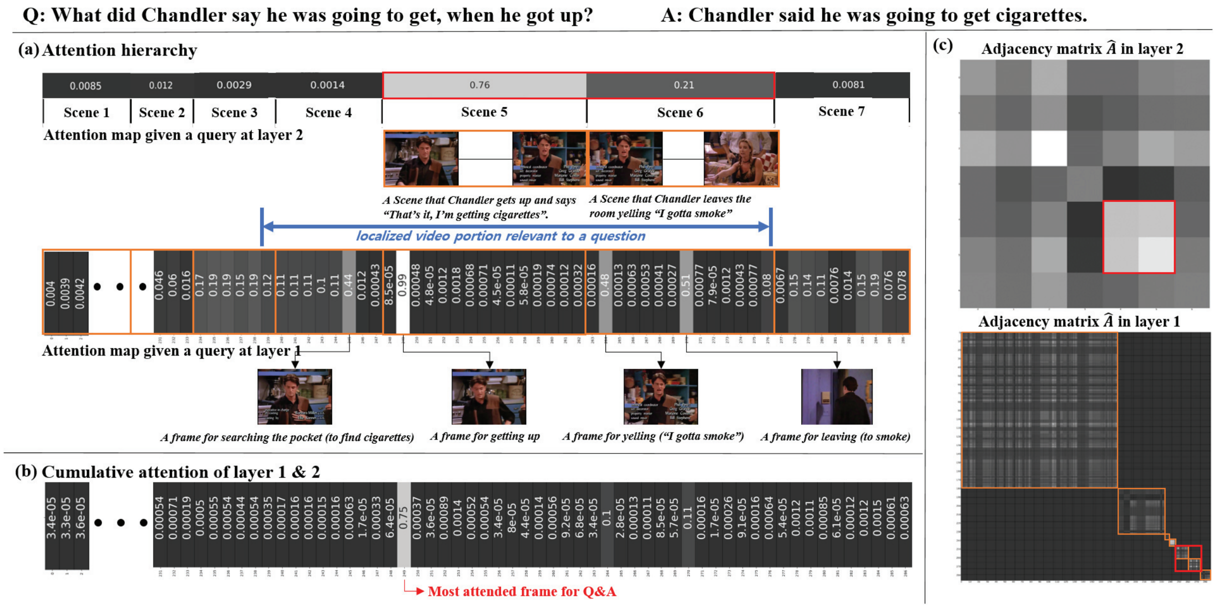
Q: What did Chandler say he was going to get, when he got up?
A: Chandler said he was going to get cigarettes.
(a) Attention hierarchy
0.0085
0.012
0.0029
0.0014
0.76
0.21
0.0081
Scene 1
Scene 2
Scene 3
Scene 4
Scene 5
Scene 6
Scene 7
Attention map given a query at layer 2
A Scene that Chandler gets up and says "That's it, I'm getting cigarettes".
A Scene that Chandler leaves the room yelling "I gotta smoke"
localized video portion relevant to a question
0.004 0.0039 0.0042 • • • 0.046 0.06 0.016 0.17 0.19 0.19 0.15 0.19 0.12 0.11 0.11 0.11 0.1 0.11 0.44 0.012 0.00043 8.5e-05 0.99 0.00048 4.8e-05 0.0012 0.0018 0.00068 0.00071 4.5e-05 0.0011 5.8e-05 0.00019 0.00074 0.00012 0.00032 0.00016 0.48 0.00013 0.00063 0.00053 0.00041 0.00027 0.51 0.00077 7.9e-05 0.0012 0.00043 0.00077 0.08 0.0067 0.15 0.14 0.11 0.0076 0.014 0.15 0.19 0.076 0.078
Attention map given a query at layer 1
A frame for searching the pocket (to find cigarettes)
A frame for getting up
A frame for yelling ("I gotta smoke")
A frame for leaving (to smoke)
(b) Cumulative attention of layer 1 & 2
3.4e-05 3.3e-05 3.6e-05 • • • 0.00054 0.00071 0.00019 0.0005 0.00055 0.00054 0.00044 0.00054 0.00035 0.00017 0.00016 0.00016 0.00015 0.00016 0.00063 1.7e-05 0.00033 6.4e-05 0.75 0.00037 3.6e-05 0.00089 0.0014 0.00052 0.00054 3.4e-05 8e-05 4.4e-05 0.00014 0.00056 9.2e-05 6.8e-05 3.4e-05 0.1 2.8e-05 0.00013 0.00011 8.5e-05 5.7e-05 0.11 0.00016 1.7e-05 0.00026 9.1e-05 0.00016 0.00064 5.4e-05 0.0012 0.0011 0.00085 6.1e-05 0.00012 0.0012 0.0015 0.00061 0.00063
Most attended frame for Q&A
(c)
Adjacency matrix $\hat{A}$ in layer 2
Adjacency matrix $\hat{A}$ in layer 1

Figure 3: An example of the learned attention hierarchy for a real video clip of "Friends" in TVQA is visualized. The question and answer for this clip are "What did Chandler say he was going to get, when he got up?" and "Chandler said he was going to get cigarettes." each. We dropped the Scene 1 and Scene 2 parts in (a) and (b), because the attention values in these scenes are nearly identical, making them non-informative. (a): Attention maps for each layer given query. the orange rectangles in layer 1 denote the cut scenes discovered by CB-GLNs and the red rectangle in layer 2 highlights scenes with high attention values given a question. (b): Cumulative attentions to the frame-level are visualized by multiplying attention values in layer 1 & 2. (c): Visualization of the learned adjacency matrices in layer 1 & 2. The orange and red rectangles are consistent with (a).

Conclusion

In this paper, we proposed Cut-Based Graph Learning Networks (CB-GLNs) which learn not only the representations of video sequences, but also composite dependency structures within the sequence. To explore characteristics of CB-GLNs, various experiments are conducted on a real large-scale video dataset YouTube-8M and TVQA. The results show that the proposed model efficiently learns the representations of sequential video data by discovering inherent dependency structure of itself.

Acknowledgements

The authors would like to thank Woo Suk Choi and Chris Hickey for helpful comments and editing. This work was partly supported by the Korea government (2015-0-00310-SW.StarLab, 2017-0-01772-VTT, 2018-0-00622-RMI, 2019-0-01367-BabyMind, P0006720-GENKO).

References

Abu-El-Haija, S.; Kothari, N.; Lee, J.; Natsev, P.; Toderici, G.; Varadarajan, B.; and Vijayanarasimhan, S. 2016. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*.

Arandjelovic, R.; Gronat, P.; Torii, A.; Pajdla, T.; and Sivic, J. 2016. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5297–5307.

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bojchevski, A.; Shchur, O.; Zügner, D.; and Günnemann, S. 2018. Netgan: Generating graphs via random walks. In *International Conference on Machine Learning*, 609–618.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chung, J.; Ahn, S.; and Bengio, Y. 2018. Hierarchical multiscale recurrent neural networks. In *International Conference on Learning Representations*.

Chung, F. R., and Graham, F. C. 1997. *Spectral graph theory*. Number 92. American Mathematical Soc.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

De Cao, N., and Kipf, T. 2018. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

Duan, Y.; Andrychowicz, M.; Stadie, B.; Ho, O. J.; Schneider, J.; Sutskever, I.; Abbeel, P.; and Zaremba, W. 2017. One-shot imitation learning. In *Advances in neural information processing systems*, 1087–1098.

Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; and Dauphin, Y. N. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1263–1272. JMLR. org.

Gori, M.; Monfardini, G.; and Scarselli, F. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, 729–734. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

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

Hershey, S.; Chaudhuri, S.; Ellis, D. P.; Gemmeke, J. F.; Jansen, A.; Moore, R. C.; Plakal, M.; Platt, D.; Saurous, R. A.; Seybold, B.; et al. 2017. Cnn architectures for large-scale audio classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, 131–135. IEEE.

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

Hoshen, Y. 2017. Vain: Attentional multi-agent predictive modeling. In *Advances in Neural Information Processing Systems*, 2701–2711.

Kalchbrenner, N.; Espeholt, L.; Simonyan, K.; Oord, A. v. d.; Graves, A.; and Kavukcuoglu, K. 2016. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*.

Kim, J.; On, K. W.; Lim, W.; Kim, J.; Ha, J.; and Zhang, B. 2017. Hadamard product for low-rank bilinear pooling. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T.; Fetaya, E.; Wang, K.-C.; Welling, M.; and Zemel, R. 2018. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, 2693–2702.

Lei, J.; Yu, L.; Bansal, M.; and Berg, T. 2018. Tvqa: Localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1369–1379.

Lin, R.; Xiao, J.; and Fan, J. 2018. Nextvlad: An efficient neural network to aggregate frame-level features for large-scale video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 0–0.

Mao, F.; Wu, X.; Xue, H.; and Zhang, R. 2018. Hierarchical video frame sequence representation with deep convolutional graph network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 0–0.

Monti, F.; Boscaini, D.; Masci, J.; Rodola, E.; Svoboda, J.; and Bronstein, M. M. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5115–5124.

Oord, A. v. d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Rasheed, Z., and Shah, M. 2005. Detection and representation of scenes in videos. *IEEE transactions on Multimedia* 7(6):1097–1105.

Sakarya, U., and Telatar, Z. 2008. Graph-based multilevel temporal video segmentation. *Multimedia systems* 14(5):277–290.

Santos, C. d.; Tan, M.; Xiang, B.; and Zhou, B. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Satorras, V. G., and Estrach, J. B. 2018. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* 22(8):888–905.

Simonovsky, M., and Komodakis, N. 2018. Graphvae: Towards generation of small graphs using variational autoencoders. In *International Conference on Artificial Neural Networks*, 412–422. Springer.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

van Steenkiste, S.; Chang, M.; Greff, K.; and Schmidhuber, J. 2018. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. In *International Conference on Learning Representations*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. In *International Conference on Learning Representations*.