Movie Summarization via Sparse Graph Construction

Pinelopi Papalampidi, Frank Keller and Mirella Lapata
Institute for Language, Cognition and Computation, School of Informatics, University of Edinburgh
p.papalampidi@sms.ed.ac.uk, {keller,mlap}@inf.ed.ac.uk

Abstract

We summarize full-length movies by creating shorter videos containing their most informative scenes. We explore the hypothesis that a summary can be created by assembling scenes which are *turning points* (TPs), i.e., key events in a movie that describe its storyline. We propose a model that identifies TP scenes by building a sparse movie graph that represents relations between scenes and is constructed using multimodal information[1]. According to human judges, the summaries created by our approach are more informative and complete, and receive higher ratings, than the outputs of sequence-based models and general-purpose summarization algorithms. The induced graphs are interpretable, displaying different topology for different movie genres.

1. Opportunity
Introductory event that occurs after presentation of setting and background of main characters. (Juno discovers she is pregnant with a child fathered by her friend and longtime admirer.)
2. Change of Plans
Main goal of story is defined; action begins to increase. (Juno decides to give the baby up for adoption.)
3. Point of No Return
Event that pushes the main characters to fully commit to their goal. (Juno meets a couple, and agrees to a closed adoption.)
4. Major Setback
Event where everything falls apart, temporarily or permanently. (Juno watches the couple's marriage fall apart.)
5. Climax
Final event of the main story, moment of resolution and "biggest spoiler". (Juno gives birth and spouse from ex-couple claims newborn as single adoptive mother.)

Figure 1: Turning points (from the movie "Juno") and their definitions.

Introduction

Automatic summarization has received considerable attention due to its importance for downstream applications. Although current research has primarily focused on news articles (Grusky, Naaman, and Artzi 2018; Narayan, Cohen, and Lapata 2018; Liu and Lapata 2019), other application domains include meetings (Murray et al. 2007), lectures (Fujii, Kitaoka, and Nakagawa 2007), social media (Syed et al. 2018), scientific articles (Teufel and Moens 2002), and narratives ranging from short stories (Goyal, Riloff, and Daumé III 2010; Finlayson 2012) to books (Mihalcea and Ceylan 2007), and movies (Gorinski and Lapata 2015).

In this work, we aim at summarizing full-length movies by creating shorter video summaries encapsulating their most informative parts. Aside from enabling users to skim through movies quickly — Netflix alone has over 148 million subscribers worldwide, with more than 6000–7000 movies, series, and shows available — movie summarization is an ideal platform for real-world natural language understanding and the complex inferences associated with it. Movies are often based on elaborate stories, with non-linear structure and multiple characters, rendering the application of popular summarization approaches based on position biases, importance, and diversity problematic (Jung et al. 2019). Another key challenge in movie summarization lies in the scarcity of labeled data. For most movies there are no naturally occurring summaries (trailers aim to attract an audience to a film without revealing spoilers which a summary will contain), and manually creating these would be a major undertaking requiring substantial effort to collect, watch, preprocess, and annotate videos. As a result, the majority of available movie datasets contain at most a few hundred movies focusing on tasks like Question-Answering (QA) or the alignment between video clips and captions (Tapaswi et al. 2016; Xiong et al. 2019; Rohrbach et al. 2015) which are limited to video snippets rather than entire movies, or restricted to screenplays disregarding the video (Gorinski and Lapata 2015; Papalampidi et al. 2020).

Following previous work (Gorinski and Lapata 2015), we formalize movie summarization as the selection of a few important scenes from a movie. We further assume that important scenes display events which determine the progression of the movie's narrative and segment it into thematic sections. Screenwriting theory (Thompson 1999; Cutting 2016; Hauge 2017) reserves the term *turning points* (TPs) for events which have specific functionality inside a narrative and reveal its storyline. TPs are considered key for making successful movies whose stories are expected to consist of six basic stages, defined by five key turning points in the plot. An example of

[1]We make our data and code publicly available at https://github.com/ppapalampidi/GraphTP.

TPs and their definitions is given in Figure 1. Interestingly, TPs are assumed to be the same, no matter the movie genre, and occupy the same positions in the story (e.g., the Opportunity occurs after the first 10% of a 90-minute comedy or a three-hour epic).

We propose that automatic movie summarization can be reduced to turning point identification building on earlier work (Lehnert 1981; Lohnert, Black, and Reiser 1981; Mihalcea and Ceylan 2007) which claims that high level analysis is necessary for revealing concepts central to a story. Although there exist several theories of narrative structure (Cutting 2016), we argue that turning points are ideally suited to summarizing movies for at least three reasons. Firstly, they are intuitive, and can be identified by naive viewers (Papalampidi, Keller, and Lapata 2019), so there is hope the process can be automated. Secondly, TPs have specific definitions and expected positions which facilitate automatic identification especially in low resource settings by providing prior knowledge (semantic and positional). Thirdly, they provide data efficiency, since the summarization problem is re-formulated as a scene-level classification task and no additional resources are required for creating the movie summaries over and above those developed for identifying turning points.

We model TP identification (and by extension summarization) as a supervised classification task. However, we depart from previous approaches to movie analysis which mostly focus on interactions between *characters* (Do, Tran, and Tran 2018; Tran et al. 2017; Gorinski and Lapata 2015) and model connections between *events*. Moreover, we discard the simplifying assumption that a screenplay consists of a *sequence of scenes* (Gorinski and Lapata 2015; Papalampidi, Keller, and Lapata 2019; Papalampidi et al. 2020) and instead represent interactions between scenes as a *sparse graph*. Specifically, we view the screenplay of a movie as a graph whose nodes correspond to scenes (self-contained events) and edges denote relations between them which we compute based on their linguistic and audiovisual similarity. In contrast to previous work on general-purpose summarization that relies on fully connected graphs (Mihalcea and Tarau 2004; Zheng and Lapata 2019; Wang et al. 2020), we induce sparse graphs by selecting a subset of nodes as neighbors for a scene; the size of this subset is not set in advance but learnt as part of the network. Sparse graphs provide better contextualization for scenes and tend to be more informative, as different genres present different degrees of connectivity between important events. We rely on Graph Convolutional Networks (GCNs; Duvenaud et al. 2015; Kearnes et al. 2016; Kipf and Welling 2017) to encode relevant neighborhood information in the sparsified graph for every scene which in turn contributes to deciding whether it acts as a TP and should be included in the summary.

Our contributions can be summarized as follows: (a) we approach movie summarization directly via TP identification which we argue is a well-defined and possibly less subjective task; (b) we propose a TP identification model which relies on sparse graphs and is constructed based on multimodal information; (c) we find that the induced graphs are meaningful with differing graph topologies corresponding to different movie genres.

Related Work

The computational treatment of narratives has assumed various guises in the literature (Mani 2012; Richards, Finlayson, and Winston 2009). Previous work has attempted to analyze stories by examining the sequence of events in them (Schank and Abelson 1975; Chambers and Jurafsky 2009), plot units (McIntyre and Lapata 2010; Goyal, Riloff, and Daumé III 2010) and their structure (Lehnert 1981; Rumelhart 1980), or the interactions of characters in the narrative (Black and Wilensky 1979; Propp 1968; Valls-Vargas, Zhu, and Ontanon 2014; Srivastava, Chaturvedi, and Mitchell 2016).

The summarization of narratives has received less attention, possibly due to the lack of annotated data for modeling and evaluation. Nevertheless, Kazantseva and Szpakowicz (2010) summarize short stories as a browsing aids to help users decide whether a story is interesting to read. Other work (Mihalcea and Ceylan 2007; Gorinski and Lapata 2015; Tsoneva, Barbieri, and Weda 2007) focuses on long-form narratives such as books or movies and adopts primarily unsupervised, graph-based methods. More recently, Papalampidi, Keller, and Lapata (2019) released TRIPOD, a dataset containing screenplays and TP annotations and showed that TPs can be automatically identified in movie narratives. In follow-on work, Papalampidi et al. (2020) further demonstrate that TPs provide useful information when summarizing episodes from the TV series CSI. In this work, we consider scenes as the basic summarization units, and reduce the scene selection task to a TP identification problem.

Work on video understanding has also looked at movies. Existing datasets (Tapaswi et al. 2016; Rohrbach et al. 2015) do not contain more than a few hundred movies and focus mostly on *isolated* video clips rather than *entire* narratives. For example, Tapaswi, Bauml, and Stiefelhagen (2015) align movie scenes to book chapters, while Xiong et al. (2019) align movie segments to descriptions using a graph-based approach. Rohrbach et al. (2015) introduce a dataset where video clips from movies are aligned to text descriptions in order to address video captioning. Tapaswi, Bäuml, and Stiefelhagen (2015) introduce a Question-Answering (QA) dataset based on movies, although the questions are again restricted to isolated video clips. Fremann, Cohen, and Lapata (2018) analyze CSI episodes with the aim of modeling how viewers identify the perpetrator.

Our work is closest to Papalampidi, Keller, and Lapata (2019) in that we also develop a model for identifying turning points in movies. While they focus solely on textual analysis, we consider additional modalities such as audio and video. Moreover, we model screenplays more globally by representing them as graphs and inferring relationships between scenes. Our graphs are interpretable and differentially represent the morphology of different genres. Beyond improving TP prediction, we further argue that narrative structure can be *directly* used to create video summaries for movies of any genre. Previous work (Papalampidi et al. 2020) treats TPs as latent representations with the aim of enhancing a supervised summarization task. We do not assume goldstandard video summaries are available, we claim that scenes which contain TPs can yield good enough proxy summaries.

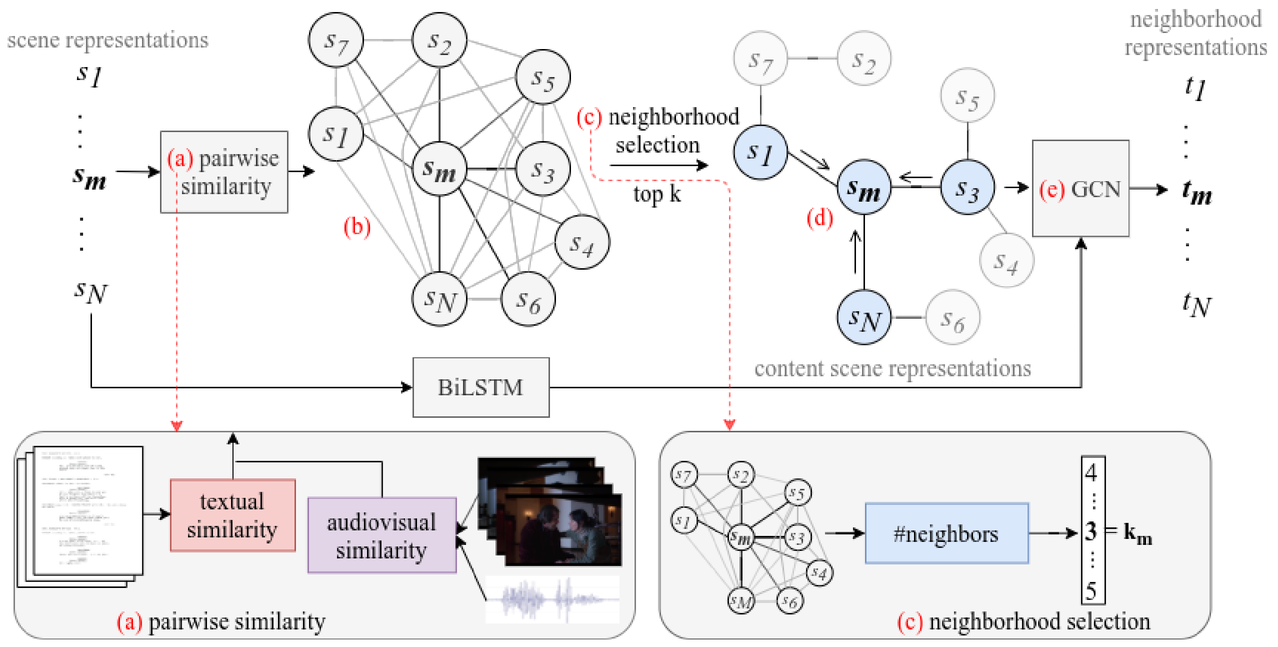

Figure 2: We construct a fully-connected graph based on pairwise textual and audiovisual similarity between scenes. The graph is sparsified (by automatically selecting the k nearest neighbors per scene) and together with contextualized scene representations is fed to a one-layer GCN.

Problem Formulation

Let $\mathcal{D}$ denote a screenplay consisting of a sequence of scenes $\mathcal{D} = \{s_1, s_2, \ldots, s_n\}$. We aim at selecting a smaller subset $\mathcal{D}' = \{s_i, \ldots, s_k\}$ consisting of the most *informative* scenes describing the movie's storyline. Hence, our objective is to assign a binary label y_i to each scene s_i denoting whether it is part of the summary.

Furthermore, we hypothesize that we can construct an informative summary by identifying TPs directly. As we explained earlier, screenwriting theory (Hauge 2017) postulates that most movie narratives are delineated by five key events called turning points (see Figure 1). Hence, we reformulate the summarization problem as follows: for each scene $s_i \in \mathcal{D}$ we assign a binary label y_{it} denoting whether it represents turning point t. Specifically, we calculate probabilities $p(y_{it}|s_i, \mathcal{D}, \theta)$ quantifying the extent to which s_i acts as the t^{th} TP, where $t \in [1, 5]$ (and θ are model parameters). During inference, we compose a summary by selecting l consecutive scenes that lie on the peak of the posterior distribution $\text{argmax}_{i=1}^{N}\, p(y_{it}|s_i, \mathcal{D}, \theta)$ for each TP.

A Turning Point Graph Model

Graph Construction

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a directed screenplay graph with nodes $\mathcal{V}$ and edges $\mathcal{E}$. $\mathcal{G}$ consists of N nodes, each corresponding to a scene (N varies with screenplay size; some screenplays have many short scenes, while others only a few long ones). We further represent $\mathcal{G}$ by an adjacency matrix $\mathcal{A} \in \mathcal{R}^{N\times N}$ where entry a_{ij} denotes the weight of the edge from node i to node j. We initially construct a dense complete graph $\mathcal{G}$ with edge weights representing the probability p_{ij} of scene i being a neighbor of scene j (see Figure 2(b)). We estimate p_{ij} as:

$$p_{ij} = \frac{\exp(e_{ij}/\tau)}{\sum_{t=1}^{T} \exp(e_{tj}/\tau)} \qquad (1)$$

where e_{ij} denotes the similarity between scenes s_i and s_j (explained in the next section), T are TPs (see Figure 1), and τ is a temperature parameter.

Similarity Computation There are various ways to compute the similarity e_{ij} between two scenes. In addition to linguistic information based on the text of the screenplay, we wish to take advantage of other modalities, such as audio and video. Audiovisual cues might be relatively superficial; simply on account of two scenes sounding or seeming alike, it might not be possible to induce which events are being described and their relations. Nevertheless, we expect audiovisual information to contribute to the similarity computation by helping distinguish scenes which refer to the same substory or event, e.g., because they have the same background, the same characters, or similar noises. We thus express e_{ij} as a composite term based mostly on textual information but also modulated by audiovisual cues:

$$e_{ij} = u_{ij}\left(\tanh(W_i v_i + b_i)^\intercal \tanh(W_j v_j + b_j)\right) + b_{ij} \qquad (2)$$

where W_i and W_j are weight matrices, v_i and v_j are *textual* vectors representing the content of scenes s_i and s_j, and u_{ij}

expresses the *audiovisual* similarity between s_i and s_j. If u_{ij} is high, the similarity between two scenes will be accentuated, but if it is low when textual similarity is high, its influence will be modest (see Figure 2(a)).

It is relatively straightforward to obtain textual representations for scenes. The latter contain mostly dialogue (lines the actors speak) as well as descriptions explaining what the camera sees. We first calculate representations for the sentences included in a scene via a pre-trained transformer-based sentence encoder (Cer et al. 2018b). We then obtain contextualized sentence representations using a BiLSTM equipped with an attention mechanism. A scene is represented as the weighted sum of the representations of its sentences.

We also assume that a scene corresponds to a sequence of audio segments extracted from the movie and a sequence of frames sampled (with a fixed sampling frequency) from the video. We first non-linearly project the features of each modality to a lower dimension and obtain scene-level representations (as the attention-weighted average of the segments/frames in each scene). The two modalities are combined into a joint representation using late fusion (Fremann, Cohen, and Lapata 2018; Papasarantopoulos et al. 2019). The audiovisual similarity u_{ij} (applied in Eq. (2)) between scenes s_i and s_j is the dot product of their fused representations.

Graph Sparsification Next, we sparsify graph $\mathcal{G}$ (or equivalently, matrix $\mathcal{A}$) by considering only k neighbors per scene (see Figure 2(d)). Compared to fully connected graphs, sparse representations are computationally more efficient and also have shown better classification accuracy (Ozaki et al. 2011; Zhu 2005). Moreover, we hypothesize that sparse graphs are crucial for our turning point identification task. We anticipate the screenplay graph to capture high-level differences and similarities between movies which would be difficult to discern when each scene is connected to every other scene.

The most common way to obtain a sparse graph is to construct a k-NN graph by introducing a threshold on the number of nearest neighbors k (Szummer and Jaakkola 2002; Goldberg and Zhu 2006; Niu, Ji, and Tan 2005). Specifically, we create sparse graph $\mathcal{G}'$ by selecting the set of neighbors $\mathcal{P}_i$ for each scene s_i as follows:

$$\mathcal{P}_i = \operatorname{argmax}_{j \in [1,N], |\mathcal{P}_i|=k} \, p_{ij} \tag{3}$$

where p_{ij} is calculated as in Eq. (1). After removing for each node the neighbors not included in the set $\mathcal{P}_i$, the new graph $\mathcal{G}'$ contains edges $|\mathcal{E}'| \ll |\mathcal{E}|$ which are unweighted.

Instead of a priori deciding on a fixed number of neighbors k for all scenes, which may cause false neighborhood assumptions, we treat k as a parameter to be learned as part of the network which computes $p(y_{it}|s_i, \mathcal{D})$, the probability of a scene being a TP. Figure 2(c) illustrates this neighborhood selection module. All connections of $s_i \in \mathcal{G}$ serve as input to a non-linear fully-connected layer which outputs a probability distribution z_i over a pre-defined set of neighborhood sizes $[1, C]$. We then select $k_i = \operatorname{argmax}_{t \in [1,C]} z_{it}$ as the neighborhood size for scene s_i in the sparse graph $\mathcal{G}'$.

When deciding on the neighborhood size k_i and the set of neighbors $\mathcal{P}_i$ for scene s_i, we perform discrete choices, which are not differentiable. We address these discontinuities in our model by utilizing the Straight-Through Estimator (Bengio, Léonard, and Courville 2013). During the backward pass we compute the gradients with the Gumbel-softmax reparametrization trick (Maddison, Mnih, and Teh 2017; Jang, Gu, and Poole 2017). To better approximate the argmax selections (for k and $\mathcal{P}$) during backpropagation, we also add a low temperature parameter $\tau = 0.1$ (Hinton, Vinyals, and Dean 2015) in the softmax function, shown in Eq. (1).

Graph Convolutional Networks

We rely on graph convolutional networks (GCNs; Duvenaud et al. 2015; Kearnes et al. 2016; Kipf and Welling 2017) to induce embeddings representing graph nodes. Our GCN operates over the sparsified graph $\mathcal{G}'$ and computes a representation for the current scene s_i based on the representation of its neighbors. We only encode information from the scene's *immediate* neighbors and thus consider one layer of convolution.[2] Moreover, in accordance with Kipf and Welling (2017), we add a self-loop to all scenes in $\mathcal{G}'$. This means that the representation of scene s_i itself affects the neighborhood representation t_i:

$$t_i = f\left(\frac{1}{|P_i \cup \{s_i\}|} \sum_{j \in P_i \cup \{s_i\}} (W_g c_j + b)\right) \tag{4}$$

where $f(.)$ is a non-linear activation function (i.e., ReLU), and vectors c represent the *content* of a scene (in relation to the overall screenplay and its relative position).

We encode the screenplay as a sequence $v_1, v_2, ..., v_N$ of textual scene representations with a BiLSTM network and obtain contextualized representations c by concatenating the hidden layers of the forward $\overrightarrow{h}$ and backward $\overleftarrow{h}$ LSTM ($c = [\overrightarrow{h}; \overleftarrow{h}]$). In other words, graph convolutions are performed on top of LSTM states (Marcheggiani and Titov 2017). The one-layer GCN only considers information about a scene's immediate neighbors, while contextualized scene representations c capture longer-range relations between scenes. Figure 2(e) illustrates our GCN and the computation of the neighborhood representations t.

Finally, we concatenate the neighborhood representation t_i and the content representation c_i to obtain an encoding for each scene s_i: $[c_i; t_i]$. This vector is fed to a single neuron that outputs probabilities $p(y_{it}|s_i, \mathcal{D})$.

Model Training

Our description so far has assumed that TP labels are available for screenplay scenes. However, in practice, such data cannot be easily sourced (due to the time consuming nature of watching movies, reading screenplays, and identifying TP locations). The only TP related dataset we are aware of is TRIPOD (Papalampidi, Keller, and Lapata 2019) which contains TP labels for sentences (not scenes) contained within movie synopses (not screenplays). For this reason, we first train a teacher model which takes as input synopses marked with

[2]Performance deteriorates when stacking GCN layers.

gold-standard TP sentences and the corresponding screenplays D and outputs the probability $q(y_{it}|s_i, D)$ for scene s_i to convey the meaning of the t^{th} TP sentence, where $t \in [1, 5]$.

We use the model proposed in Papalampidi, Keller, and Lapata (2019) as teacher to obtain probability distribution $q(y_t|D)$ over screenplay D. The TP-specific posterior distributions produced by the teacher model are used to train our model which only takes scenes as input. Similarly to knowledge distillation settings (Ba and Caruana 2014; Hinton, Vinyals, and Dean 2015) we utilize the KL divergence loss between the teacher posterior distributions $q(y_t|D)$ and the ones computed by our model $p(y_t|D)$:

$$\mathcal{O}_t = \mathcal{D}_{KL}\left(p(y_t|D) \| q(y_t|D)\right), t \in [1, T] \qquad (5)$$

where T is the number of TPs. We further add a second objective to the loss function in order to control adjacency matrix S and hence the latent graph $\mathcal{G}'$. Intuitively, we want to assign higher probabilities for scenes to be neighbors in $\mathcal{G}'$ if they are also temporally close in the screenplay. For this reason, we add a focal regularization term $\mathcal{F}$ to the loss function. Specifically, we assume a Gaussian distribution g_i over the screenplay centered around the current scene index i and try to keep the probability distribution in matrix S that corresponds to candidate neighbors for scene s_i close to the prior distribution: $\mathcal{F}_i = \mathcal{D}_{KL}\left(p_i \| g_i\right)$.[3] The loss function now becomes:

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\mathcal{O}_t + \lambda\frac{1}{N}\sum_{i=1}^{N}\mathcal{F}_i \qquad (6)$$

where λ is a hyperparameter.

Experimental Setup

Multimodal TRIPOD We performed experiments on the TRIPOD dataset [4] (Papalampidi, Keller, and Lapata 2019) originally used for analyzing the narrative structure of movies. We augmented this dataset by collecting gold-standard annotations for 23 new movies which we added to the test set. The resulting dataset contains 17,150 scenes from 122 movies, 38 of which have gold-standard scene-level annotations and were used for evaluation purposes. We also collected the videos and subtitles for the TRIPOD movies. Table 1 presents the dataset statistics.

Data Preprocessing We used the Universal Sentence Encoder (USE; Cer et al. 2018a) to obtain sentence-level representations. Following previous work (Tapaswi, Bäuml, and Stiefelhagen 2015), subtitles (and their timestamps on the movie video) were aligned to the dialogue parts of the screenplay using Dynamic Time Wrapping (DTW; Myers and Rabiner 1981). Subsequently, we obtained alignments of screenplay scenes to video segments. Finally, we segmented the video into scenes and extracted audiovisual features.

For the visual modality, we first sampled one out of every 50 frames within each scene. However, the length of a scene can vary from a few seconds to several minutes. For this reason, in cases where the number of sampled frames became too big for memory, we lowered the sampling frequency to one frame per 150. We employed ResNeXt-101 (Xie et al. 2016) pre-trained for object recognition on ImageNet (Deng et al. 2009) to extract a visual representation per frame. Similarly, for the audio modality, we used YAMNet pre-trained on the AudioSet-YouTube corpus (Gemmeke et al. 2017) for classifying audio segments into 521 audio classes (e.g., tools, music, explosion); for each audio segment contained in the scene, we extracted features from the penultimate layer.

	Train	**Test**
movies	84	38
scenes	11,320	5,830
TP scenes	1,260 (SS)	340 (GS)
vocabulary	37.8k	28.3k
	per movie	
scenes	133.0 (61.1)	153.4 (54.0)
sentences	3.0k (0.9)	2.9k (0.6)
tokens	23.0k (6.6)	21.5k (4.0)
video length (secs)	6.8k (1.1)	6.9k (1.3)
video frames	4.2k (3.0)	3.5k (1.1)
audio segments	12.0k (10.3)	9.7k (3.1)
	per scene	
sentences	22.2 (31.5)	19.0 (24.9)
tokens	173.0 (235.0)	139.9 (177.5)
sentence tokens	7.8 (6.0)	7.4 (6.0)
video length (secs)	88.1 (152.5)	81.6 (114.8)
video frames	29.2 (37.3)	23.0 (26.3)
audio segments	82.8 (133.6)	62.9 (94.0)

Table 1: Statistics of the augmented TRIPOD dataset; means are shown with standard deviation in brackets. SS: silver-standard labels based on Papalampidi, Keller, and Lapata (2019), GS: gold-standard labels.

Implementation Details Following (Papalampidi, Keller, and Lapata 2019), we select $l = 3$ consecutive scenes to represent each TP in the summary. Moreover, we set the maximum size of neighbors C that can be selected for a scene in graph $\mathcal{G}'$ to 6, since we want to create a sparse and interpretable graph. Experiments with fixed-sized neighborhoods also showed that performance dropped when considering neighborhoods over 6 scenes. For training our model we set the hyperparameter λ in Eq. (6) to 10. We used the Adam algorithm (Kingma and Ba 2014) for optimizing our networks. We chose an LSTM with 64 neurons for encoding scenes in the screenplay and an identical one for contextualizing them. We also added a dropout of 0.2. Our models were developed in PyTorch (Paszke et al. 2019) and PyTorch geometric (Fey and Lenssen 2019). For analyzing the movie graphs we used NetworkX (Hagberg, Swart, and S Chult 2008).

Results

Our experiments were designed to answer three questions: (1) Is the proposed graph-based model better at identifying TPs compared to less structure-aware variants? (2) To what extent are graphs and multimodal information helpful? and

[3] We disregard τ (see Eq. (1)) while recalculating probabilities p_{ij}, since we want to directly regulate the e_{ij} values.

[4] https://github.com/ppapalampidi/TRIPOD

	TA ↑	**PA** ↑	**D** ↓
Random (evenly distributed)	4.82	6.95	12.35
Theory position	4.41	6.32	11.03
Distribution position	5.59	7.37	10.74
TEXTRANK	6.18	10.00	17.77
+ audiovisual	6.18	10.00	18.90
SCENESUM	4.41	7.89	16.86
+ audiovisual	6.76	11.05	18.93
TAM	7.94	9.47	**9.42**
+ audiovisual	7.36	10.00	10.01
GRAPHTP	6.76	10.00	9.62
+ audiovisual	**9.12**	**12.63**	9.77

Table 2: Five-fold crossvalidation. Total Agreement (TA), Partial Agreement (PA), and mean distance D.

(3) Are the summaries produced by automatically identified TPs meaningful?

Which Model Identifies TPs Best Table 2 addresses our first question. We perform 5-fold cross-validation over 38 gold-standard movies to obtain a test-development split and evaluate model performance in terms of three metrics: Total Agreement (TA), i.e., the percentage of TP scenes that are correctly identified, Partial Agreement (PA), i.e., the percentage of TP events for which at least one gold-standard scene is identified, and Distance (D), i.e., the minimum distance in number of scenes between the predicted and gold-standard set of scenes for a given TP, normalized by the screenplay length (see Appendix for a more detailed definition of the evaluation metrics). We consider TA and PA as our main evaluation metrics as they measure the percentage of exact TP matches. However, apart from identifying important events, when producing a summary it is also important to display events from all parts of the movie in order to accurately describe its storyline. For this reason, we also employ the distance D metric which quantifies how well distributed the identified TP events are in the movie. Hence, a disproprionately large D suggests that the model fails to even predict the correct sections of a movie where TPs might be located let alone the TPs themselves.

The first block in the table compares our graph-based TP identification model (henceforth GRAPHTP) against the following baselines: a random selection of a sequence of three scenes from five evenly segmented sections of the movie (reported mean of five runs); the selection of a sequence of three scenes that lie on the expected position of each TP event according to screenwriting theory (Hauge 2017); and the selection of a sequence of three scenes based on the position of gold-standard TPs in the synopses of the TRIPOD training set. The second block includes the performance of two unsupervised summarization models: TEXTRANK (Mihalcea and Tarau 2004) with neural input representations (Zheng and Lapata 2019)[5] and SCENESUM (Papalampidi et al. 2020; Gorinski and Lapata 2015), a variant of TEXTRANK that takes the characters participating in each scene into account.

[5]We also experimented with directed TEXTRANK (Zheng and Lapata 2019), but these results were poor and are omitted for the sake of brevity.

	TA ↑	**PA** ↑	**D** ↓
Fully connected graph	5.00	7.37	9.73
Content	5.59	7.37	9.98
Neighborhood & position	**9.71**	**13.16**	10.98
GRAPHTP	6.76	10.00	**9.62**
+ vision	6.18	7.89	9.84
+ audio	7.06	8.95	10.38
+ audiovisual	9.12	12.63	9.77

Table 3: GRAPHTP variants. Total Agreement (TA), Partial Agreement (PA), and mean distance D.

	SCENESUM	TAM	GRAPHTP	Gold
TP1	28	28	**64**	66
TP2	36	54	**64**	62
TP3	38	18	**44**	54
TP4	26	34	**52**	56
TP5	8	16	**24**	48
Mean	27	30	**50**	57
Rating	2.63	2.68	**3.02**	3.58

Table 4: Human evaluation; proportion of TPs found in video summaries (shown as percentages) and average ratings attributed to each system (ratings vary from 1 to 5, with 5 being best). All pairwise differences are significant ($p < 0.05$, using a χ^2 test).

Finally, we report results for the Topic-Aware Model (TAM; Papalampidi et al. 2020); TAM is sequence-based supervised model which employs a sliding context window and computes the similarity between sequential contexts. We discuss implementation details for comparison systems in the Appendix. We also report the performance of a multimodal variant for all comparison systems (+audiovisual). For the unsupervised models, we add scene-level features as extra weights to the pairwise similarity calculation between scenes similarly to GRAPHTP. For TAM, we add audiovisual information via early fusion; we concatenate the scene-level vectors from all modalities (i.e., text, vision, audio).

The unsupervised summarization models (TEXTRANK, SCENESUM) have competitive performance in terms of TA and PA, but significantly higher average distance D. This suggests that they do not select events from all parts of a story but favor specific sections. For the supervised models (TAM and GRAPHTP), the average D is in general lower, which means that they are able to adapt to the positional bias and select events from all parts of a movie. Moreover, both models seem to benefit from multimodal information (see TA and PA metrics). Finally, GRAPHTP seems to perform best, by correctly identifying a higher number of gold-standard TP events (based on both TA and PA metrics), whereas D is comparable for TAM and GRAPHTP.

Which Information Matters Table 3 answers our second question by presenting an ablation study on GRAPHTP. We observe that the performance of a similar model which uses a fully-connected graph drops across metrics. This is also the case when we do not take into account a graph or any other form of interaction between scenes (i.e., only content).

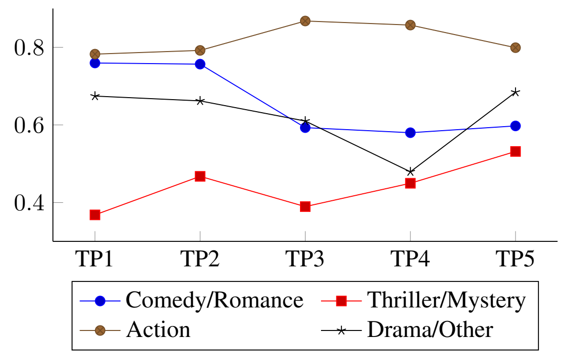

Figure 3: Average node connectivity per TP across movie genres (GRAPHTP (+audiovisual), test set.)

We also test the model's performance when we remove the content representation and keep only the neighborhood interactions together with a vector that simply encodes the position of a scene in the screenplay (as a one-hot vector). This model may not be able to adapt to the positional bias as well (higher D), but is able to predict TPs with higher accuracy (high TA and PA). Finally, we find that audio and visual information boost model performance in combination but not individually.

How Good Are the Summaries We now answer our last question by evaluating the video summaries produced by our model. We conducted a human evaluation experiment using the videos created for 10 movies of the test set. We produced summaries based on the gold scene-level annotations (Gold), SCENESUM, TAM, and GRAPHTP (see Appendix). For all systems we used model variants which consider audiovisual information except for SCENESUM. Inclusion of audiovisual information for this model yields overly long summaries (in the excess of 30 minutes) for most movies. All other systems produce 15 minutes long summaries on average.

Our study was conducted on Amazon Mechanical Turk (AMT). Crowdworkers first read a short summary of the movie (i.e., an abridged version of the Wikipedia plot synopsis). Subsequently, they were asked to watch a video summary and answer five questions each pertaining to a specific TP event (described in the textual summary). AMT workers answered with 'Yes' if they were certain it was present in the video, 'No' if the event was absent, and 'Unsure' otherwise. Finally, we asked AMT workers to provide an overall rating from 1 to 5, with 5 being the most informative summary. Workers were asked to take into account the questions answered previously, but also consider the overall quality of the summary (i.e., how compressed it was, whether it contained redundant events, and the overall information provided). We asked 5 different workers to evaluate each movie summary. Instructions and example questions are in the Appendix.

Table 4 shows the proportion of 'Yes' answers (per TP and overall mean) and the average system rating.[6] Perhaps unsurprisingly gold summaries are the most informative. Some

[6]We omit 'Unsure' from Table 4, since it only accounts for 4.1% of the answers.

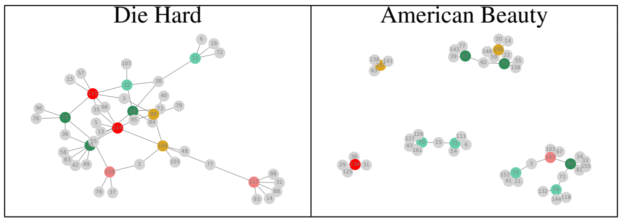

Figure 4: Examples of graphs produced by GRAPHTP (+audiovisual) for two movies from the test set. Nodes (in color) are scenes which act as TPs and their immediate neighbors (in gray).

key events might still be absent due to errors in the automatic alignment between the screenplay scenes and the video. GRAPHTP is the second best system overall (and across TPs), while SCENESUM and TAM have similar ratings. GRAPHTP manages to create more informative and diverse summaries, presenting important events from all parts of the story.

What Do the Graphs Mean We further analyzed the graphs induced by our model, in particular their connectivity. Figure 3 shows the average node connectivity per TP (i.e., minimum number of nodes that need to be removed to separate the remaining nodes into isolated subgraphs) for the movies in the test set. For this analysis, graphs were pruned to nodes which act as TPs and their immediate neighbors and movies were grouped in four broad genre categories (comedy/romance, thriller/mystery, action and drama/other). We find that thrillers and mysteries correspond to more disconnected graphs followed by dramas, while comedies, romance and especially action movies display more connected graphs. This is intuitive, since comedies and action movies tend to follow predictable storylines, while thrillers often contain surprising events which break screenwriting conventions. Moreover, for comedies and dramas the introductory events (i.e., first two TPs) are the central ones in the graph and connectivity decreases as the story unfolds and unexpected events take place. We see the opposite trend for thrillers and action movies. Initial events present lower connectivity, while the last ones are now central when crucial information is revealed justifying earlier actions (e.g., see the last two TPs which correspond to 'major setback' and 'climax'). A similar picture emerges when visualizing the graphs (see Figure 4). "Die Hard", a conventional action movie, has a clear storyline and seems more connected, while "American Beauty", a drama with several flashbacks, contains several disconnected subgraphs (see Appendix for more illustrations).

Conclusions

In this paper we demonstrate that TP identification can be used directly for summarizing movies. We propose GRAPHTP, a model that operates over sparse graphs relying on multimodal information. Summaries created by GRAPHTP are preferred by humans in contrast to general summarization algorithms and sequence-based models. In the future, we will explore ways to further exploit the graph structure and definitions of TPs in order to produce personalized summaries.

References

Ba, J.; and Caruana, R. 2014. Do Deep Nets Really Need to be Deep? In *Proceedings of the Advances in NeurIPS*.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432* .

Black, J. B.; and Wilensky, R. 1979. An evaluation of story grammars. *Cognitive science* 3(3): 213–229.

Cer, D.; Yang, Y.; Kong, S.-y.; Hua, N.; Limtiaco, N.; John, R. S.; Constant, N.; Guajardo-Cespedes, M.; Yuan, S.; Tar, C.; et al. 2018a. Universal sentence encoder. *arXiv preprint arXiv:1803.11175* .

Cer, D.; Yang, Y.; Kong, S.-y.; Hua, N.; Limtiaco, N.; St. John, R.; Constant, N.; Guajardo-Cespedes, M.; Yuan, S.; Tar, C.; Strope, B.; and Kurzweil, R. 2018b. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on EMNLP: System Demonstrations*.

Chambers, N.; and Jurafsky, D. 2009. Unsupervised Learning of Narrative Schemas and their Participants. In *Proceedings of the 47th Annual Meeting of ACL and the 4th IJCNLP*.

Cutting, J. E. 2016. Narrative theory and the dynamics of popular movies. *Psychonomic bulletin & review* 23(6).

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on CVPR*, 248–255. Ieee.

Do, T. T. H.; Tran, Q. H. B.; and Tran, Q. D. 2018. Movie indexing and summarization using social network techniques. *Vietnam Journal of Computer Science* 5(2): 157–164.

Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in NeurIPS*, 2224–2232.

Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Finlayson, M. A. 2012. *Learning Narrative Structure from Annotated Folktales*. Ph.D. thesis, MIT.

Frermann, L.; Cohen, S. B.; and Lapata, M. 2018. Whodunnit? Crime Drama as a Case for Natural Language Understanding. *TACL* 6.

Fujii, Y.; Kitaoka, N.; and Nakagawa, S. 2007. Automatic extraction of cue phrases for important sentences in lecture speech and automatic lecture speech summarization. In *INTERSPEECH*.

Gemmeke, J. F.; Ellis, D. P.; Freedman, D.; Jansen, A.; Lawrence, W.; Moore, R. C.; Plakal, M.; and Ritter, M. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE ICASSP*, 776–780. IEEE.

Goldberg, A.; and Zhu, X. 2006. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for NLP*.

Gorinski, P. J.; and Lapata, M. 2015. Movie Script Summarization as Graph-based Scene Extraction. In *Proceedings of the 2015 Conference of NAACL-HLT*.

Goyal, A.; Riloff, E.; and Daumé III, H. 2010. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on EMNLP*, 77–86.

Grusky, M.; Naaman, M.; and Artzi, Y. 2018. NEWSROOM: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of the 16th Annual Conference of NAACL-HLT*.

Hagberg, A.; Swart, P.; and S Chult, D. 2008. Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Hauge, M. 2017. *Storytelling Made Easy: Persuade and Transform Your Audiences, Buyers, and Clients – Simply, Quickly, and Profitably*. Indie Books International.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Jang, E.; Gu, S.; and Poole, B. 2017. Categorical Reparametrization with Gumble-Softmax. In *ICLR*.

Jung, T.; Kang, D.; Mentch, L.; and Hovy, E. 2019. Earlier Isn't Always Better: Sub-aspect Analysis on Corpus and System Biases in Summarization. *arXiv preprint arXiv:1908.11723* .

Kazantseva, A.; and Szpakowicz, S. 2010. Summarizing Short Stories. *Computational Linguistics* 36(1).

Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; and Riley, P. 2016. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design* 30(8): 595–608.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

Lehnert, W. G. 1981. Plot Units and Narrative Summarization. *Cognitive Science* 5(4): 293–331.

Liu, Y.; and Lapata, M. 2019. Hierarchical Transformers for Multi-Document Summarization. In *Proceedings of the 57th Annual Meeting of ACL*.

Lohnert, W. G.; Black, J. B.; and Reiser, B. J. 1981. Summarizing narratives. In *Proceedings of the 7th international joint conference on Artificial intelligence-Volume 1*, 184–189.

Maddison, C. J.; Mnih, A.; and Teh, Y. W. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *5th ICLR*. OpenReview.net.

Mani, I. 2012. *Computational Modeling of Narative*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool Publishers.

Marcheggiani, D.; and Titov, I. 2017. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In *Proceedings of the 2017 Conference on EMNLP*.

McIntyre, N.; and Lapata, M. 2010. Plot Induction and Evolutionary Search for Story Generation. In *Proceedings of the 48th Annual Meeting of ACL*.

Mihalcea, R.; and Ceylan, H. 2007. Explorations in Automatic Book Summarization. In *Proceedings of the 2007 Joint Conference on EMNLP and CoNLL*.

Mihalcea, R.; and Tarau, P. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference of EMNLP*.

Murray, G.; Hsueh, P.-Y.; Tucker, S.; Kilgour, J.; Carletta, J.; Moore, J. D.; and Renals, S. 2007. Automatic segmentation and summarization of meeting speech. In *Proceedings of NAACL-HLT*, 9–10.

Myers, C. S.; and Rabiner, L. R. 1981. A comparative study of several dynamic time-warping algorithms for connected-word recognition. *Bell System Technical Journal* 60(7).

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Proceedings of the 2018 Conference on EMNLP*.

Niu, Z.-Y.; Ji, D.-H.; and Tan, C. L. 2005. Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning. In *Proceedings of the 43rd Annual Meeting of ACL*.

Ozaki, K.; Shimbo, M.; Komachi, M.; and Matsumoto, Y. 2011. Using the mutual k-nearest neighbor graphs for semi-supervised classification on natural language data. In *Proceedings of the fifteenth conference of CoNLL*, 154–162.

Papalampidi, P.; Keller, F.; Frermann, L.; and Lapata, M. 2020. Screenplay Summarization Using Latent Narrative Structure. In *Proceedings of the 58th Annual Meeting of ACL*.

Papalampidi, P.; Keller, F.; and Lapata, M. 2019. Movie Plot Analysis via Turning Point Identification. In *Proceedings of the 2019 Conference on EMNLP and the 9th IJCNLP*.

Papasarantopoulos, N.; Frermann, L.; Lapata, M.; and Cohen, S. B. 2019. Partners in Crime: Multi-view Sequential Inference for Movie Understanding. In *Proceedings of the 2019 Conference on EMNLP and the 9th IJCNLP*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in NeurIPS*, 8024–8035.

Propp, V. I. 1968. *Morphology of the Folktale*. University of Texas.

Richards, W.; Finlayson, M. A.; and Winston, P. H. 2009. Advancing Computational Models of Narrative. Technical Report 63:2009, MIT Computer Science and Atrificial Intelligence Laboratory.

Rohrbach, A.; Rohrbach, M.; Tandon, N.; and Schiele, B. 2015. A dataset for movie description. In *Proceedings of the IEEE conference on CVPR*.

Rumelhart, D. E. 1980. On evaluating story grammars. *Cognitive Science* 4(3): 313–316.

Schank, R. C.; and Abelson, R. P. 1975. Scripts, Plans, and Knowledge. In *Proceedings of the 4th International Joint Conference on Artificial Intelligence*, 151–157. Tblisi, USSR.

Srivastava, S.; Chaturvedi, S.; and Mitchell, T. 2016. Inferring Interpersonal Relations in Narrative Summaries. In *Proceedings of the 13th AAAI Conference*.

Syed, S.; Völske, M.; Potthast, M.; Lipka, N.; Stein, B.; and Schütze, H. 2018. Task Proposal: The TL; DR Challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, 318–321.

Szummer, M.; and Jaakkola, T. 2002. Information regularization with partially labeled data. *Advances in Neural Information processing systems* 15: 1049–1056.

Tapaswi, M.; Bäuml, M.; and Stiefelhagen, R. 2015. Aligning plot synopses to videos for story-based retrieval. *International Journal of Multimedia Information Retrieval* 4(1).

Tapaswi, M.; Bauml, M.; and Stiefelhagen, R. 2015. Book2movie: Aligning video scenes with book chapters. In *Proceedings of the IEEE Conference on CVPR*, 1827–1835.

Tapaswi, M.; Zhu, Y.; Stiefelhagen, R.; Torralba, A.; Urtasun, R.; and Fidler, S. 2016. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on CVPR*.

Teufel, S.; and Moens, M. 2002. Articles Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status. *Computational Linguistics* 28(4).

Thompson, K. 1999. *Storytelling in the new Hollywood: Understanding classical narrative technique*. Harvard University Press.

Tran, Q. D.; Hwang, D.; Lee, O.-J.; and Jung, J. E. 2017. Exploiting character networks for movie summarization. *Multimedia Tools and Applications* 76(8): 10357–10369.

Tsoneva, T.; Barbieri, M.; and Weda, H. 2007. Automated summarization of narrative video on a semantic level. In *ICSC*.

Valls-Vargas, J.; Zhu, J.; and Ontanon, S. 2014. Toward automatic role identification in unannotated folk tales. In *Proceedings of the 10th AAAI Conference*, 188–194.

Wang, D.; Liu, P.; Zheng, Y.; Qiu, X.; and Huang, X. 2020. Heterogeneous Graph Neural Networks for Extractive Document Summarization. In *Proceedings of the 58th Annual Meeting of ACL*.

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2016. Aggregated Residual Transformations for Deep Neural Networks. *arXiv preprint arXiv:1611.05431* .

Xiong, Y.; Huang, Q.; Guo, L.; Zhou, H.; Zhou, B.; and Lin, D. 2019. A Graph-Based Framework to Bridge Movies and Synopses. In *Proceedings of the IEEE ICCV*, 4592–4601.

Zheng, H.; and Lapata, M. 2019. Sentence Centrality Revisited for Unsupervised Summarization. *arXiv preprint arXiv:1906.03508* .

Zhu, X. 2005. *Semi-supervised Learning with Graphs*. Ph.D. thesis, Carnegie Mellon University.