Movie Plot Analysis via Turning Point Identification

Citation for published version:

Digital Object Identifier (DOI):
10.18653/v1/D19-1180

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Proceedings of the Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on NLP

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Movie Plot Analysis via Turning Point Identification

Pinelopi Papalampidi  Frank Keller  Mirella Lapata
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
p.papalampidi@sms.ed.ac.uk, {keller,mlap}@inf.ed.ac.uk

Abstract

According to screenwriting theory, turning points (e.g., change of plans, major setback, climax) are crucial narrative moments within a screenplay: they define the plot structure, determine its progression and thematic units (e.g., setup, complications, aftermath). We propose the task of turning point identification in movies as a means of analyzing their narrative structure. We argue that turning points and the segmentation they provide can facilitate processing long, complex narratives, such as screenplays, for summarization and question answering. We introduce a dataset consisting of screenplays and plot synopses annotated with turning points and present an end-to-end neural network model that identifies turning points in plot synopses and projects them onto scenes in screenplays. Our model outperforms strong baselines based on state-of-the-art sentence representations and the expected position of turning points.

1 Introduction

Computational literary analysis works at the intersection of natural language processing and literary studies, aiming to evaluate various theories of storytelling (e.g., by examining a collection of works within a single genre, by an author, or topic) and to develop tools which aid in searching, visualizing, or summarizing literary content.

Within natural language processing, computational literary analysis has mostly targeted works of fiction such as novels, plays, and screenplays. Examples include analyzing characters, their relationships, and emotional trajectories (Chaturvedi et al., 2017; Iyyer et al., 2016; Elsner, 2012), identifying enemies and allies (Nalisnick and Baird, 2013), villains or heroes (Bamman et al., 2014, 2013), measuring the memorability of quotes (Danescu-Niculescu-Mizil et al., 2012), characterizing gender representation in dialogue (Agarwal et al., 2015; Ramakrishna et al., 2015; Sap et al., 2017), identifying perpetrators in crime series (Frermann et al., 2018), summarizing screenplays (Gorinski and Lapata, 2018), and answering questions about long and complex narratives (Kočiský et al., 2018).

In this paper we are interested in the automatic analysis of narrative structure in screenplays. Narrative structure, also referred to as a storyline or plotline, describes the framework of how one tells a story and has its origins to Aristotle who defined the basic triangle-shaped plot structure representing the beginning (protasis), middle (epitasis), and end (catastrophe) of a story (Pavis, 1998). The German novelist and playwright Gustav Freytag modified Aristotle’s structure by transforming the triangle into a pyramid (Freytag, 1896). In his scheme, there are five acts (introduction, rising movement, climax, return, and catastrophe). Several variations of Freytag’s pyramid are used today in film analysis and screenwriting (Cutting, 2016).

In this work, we adopt a variant commonly employed by screenwriters as a practical guide for producing successful screenplays (Hague, 2017). According to this scheme, there are six stages (acts) in a film, namely the setup, the new situation, progress, complications and higher stakes, the final push, and the aftermath, separated by five turning points (TPs). TPs are narrative moments from which the plot goes in a different direction.

<table>
<thead>
<tr>
<th>Turning Point</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Opportunity</td>
<td>Introductory event that occurs after the presentation of the setting and the background of the main characters.</td>
</tr>
<tr>
<td>2. Change of Plans</td>
<td>Event where the main goal of the story is defined. From this point on, the action begins to increase.</td>
</tr>
<tr>
<td>3. Point of No Return</td>
<td>Event that pushes the main character(s) to fully commit to their goal.</td>
</tr>
<tr>
<td>4. Major Setback</td>
<td>Event where everything falls apart (temporarily or permanently).</td>
</tr>
<tr>
<td>5. Climax</td>
<td>Final event of the main story, moment of resolution and the “biggest spoiler”.</td>
</tr>
</tbody>
</table>

Table 1: Turning points and their definitions.
Recently divorced Meg Altman and her 11-year-old daughter Sarah have just purchased a four-story brownstone on New York City. The house’s previous owner installed an isolated room used to protect the house’s occupants from intruders. On the night the two move into the home, it is broken by Junior, the previous owner’s grandson; Burnham, an employee of the security company; and Raoul, a ski-mask-wearing gunman.

The three are after $3 million in bearer bonds, which are locked inside a floor safe in the panic room. As they begin the robbery, Meg wakes up and happens to see the intruders on the video monitors in the panic room. Before the three can reach them, Meg and Sarah run into the panic room and close the door behind them, only to find that the burglars have disabled the telephone.

็TTT

Intending to force them out of the room, Burnham introduces propane gas into the room’s air vents. Meg then taps into the main telephone line and gets through to her ex-husband Stephen, before the burglars cut them off. Stephen arrives at the home and is taken hostage by Burnham and Raoul, who severely beats him. To make matters worse, Sarah, who has diabetes, suffer a seizure.

Her glucagon syringe is in a refrigerator outside the panic room. After using an unconscious Stephen to trick Meg into momentarily leaving the panic room, Burnham enters it, finding Sarah motionless on the floor. After Burnham gives Sarah the injection, Sarah thanks him. Having earlier received a call from Stephen, two policemen arrive, which prompts Raoul to threaten Sarah’s life. Sensing the potential danger to her daughter, Meg lies to the officers and they leave.

Meanwhile, Burnham opens the safe and robs the $2 million in bearer bonds inside. As the robbers attempt to leave, using Sarah as a hostage, Meg hits Burn with a sledgehammer and Raoul flees. After a badly injured Stephen shoots at Raoul and misses, Raoul flips him over and punishes him. Burnham, upon hearing Sarah’s screams of pain, returns to the house and shoots Raoul dead, stating, “You’ll be okay now,” to Meg and her daughter before leaving.

The police, alerted by Meg’s suspicious behavior earlier, arrive in force and capture Burnham. Later, Meg and Sarah, having recovered from their harrowing experience, begin searching the newspaper for a new home. After a badly injured Stephen shoots at Raoul and misses, Raoul flips him over and punishes him. Burnham, upon hearing Sarah’s screams of pain, returns to the house and shoots Raoul dead, stating, “You’ll be okay now,” to Meg and her daughter before leaving.
tion. Finally, Kočiský et al. (2018) recently introduced a dataset consisting of question-answer pairs over 1,572 movie screenplays and books.

Previous approaches have focused on fine-grained story analysis, such as inducing character types (Bamman et al., 2013, 2014) or understanding relationships between characters (Iyyer et al., 2016; Chaturvedi et al., 2017). Various approaches have also attempted to analyze the goal and structure of narratives. Black and Wilensky (1979) evaluate the functionality of story grammars in story understanding, Elson and McKeown (2009) develop a platform for representing and reasoning over narratives, and Chambers and Jurafsky (2009) learn fine-grained chains of events.

In the context of movie summarization, Gorinski and Lapata (2018) automatically generate an overview of the movie’s genre, mood, and artistic style based on screenplay analysis. Gorinski and Lapata (2015) summarize full length screenplays by extracting an optimal chain of scenes via a graph-based approach centered around the characters of the movie. A similar approach has also been adopted by Vicol et al. (2018), who introduce the MovieGraphs dataset consisting of 51 movies and describe video clips with character-centered graphs. Other work creates animated story-boards using the action descriptions of screenplays (Ye and Baldwin, 2008), extracts social networks from screenplays (Agarwal et al., 2014a), or creates xkcd movie narrative charts (Agarwal et al., 2014b).

Our work also aims to analyze the narrative structure of movies, but we adopt a high-level approach. We advocate TP identification as a precursor to more fine-grained analysis that unveils character attributes and their relationships. Our approach identifies key narrative events and segments the screenplay accordingly; we argue that this type of preprocessing is useful for applications which might perform question answering and summarization over screenplays. Although our experiments focus solely on the textual modality, turning point analysis is also relevant for multimodal tasks such as trailer generation and video summarization.

3 The TRIPOD Dataset

The TRIPOD dataset contains 99 screenplays, accompanied with cast information (according to IMDb), and Wikipedia plot synopses annotated with turning points. The movies were selected from the Scriptbase dataset (Gorinski and Lapata, 2015) based on the following criteria: (a) maintaining a variation across different movie genres (e.g., action, romance, comedy, drama) and narrative types (e.g., flashbacks, time shifts); and (b) including screenplays that are faithful to the released movies and their synopses as much as possible. In Table 2, we present various statistics of the dataset.

Our motivation for obtaining TP annotations at the synopsis level (coarse-grained), instead of at the screenplay level (fine-grained) was twofold. Firstly, on account of being relatively short, synopses are easier to annotate than full-length screenplays, allowing us to scale the dataset in the future. Secondly, we would expect synopsis-level annotations to be more reliable and the degree of inter-annotator agreement higher; asking annotators to identify precisely where a turning point occurs might seem like looking for a needle in a haystack. An example of a synopsis with TP annotations is shown in Figure 1 for the movie “Panic Room”. Each TP is colored differently, and both the chain of key events (colored text) and resulting segmentation (§) are illustrated.

In an initial pilot study, the three authors acted as annotators for identifying TPs in movie synopses. They selected exactly one sentence per TP, under the assumption that all TPs are present. Based on the pilot, annotation instructions were devised and an annotation tool was created which allows to label synopses with TPs sentence-by-sentence. After piloting the annotation scheme on 30 movies, two new annotators were trained using our instructions and in a second study, they doubly annotated five movies. The remaining movies

<table>
<thead>
<tr>
<th>Movies</th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>movies</td>
<td>84</td>
<td>15</td>
</tr>
<tr>
<td>turning points</td>
<td>420</td>
<td>75</td>
</tr>
<tr>
<td>synopsis sentences</td>
<td>2,821</td>
<td>508</td>
</tr>
<tr>
<td>screenplay scenes</td>
<td>11,320</td>
<td>2,083</td>
</tr>
<tr>
<td>synopsis vocabulary</td>
<td>7.9k</td>
<td>2.8k</td>
</tr>
<tr>
<td>screenplay vocabulary</td>
<td>37.8k</td>
<td>16.8k</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Per Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td>tokens</td>
</tr>
<tr>
<td>sentences</td>
</tr>
<tr>
<td>sentence tokens</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Per Screenplay</th>
</tr>
</thead>
<tbody>
<tr>
<td>tokens</td>
</tr>
<tr>
<td>sentences</td>
</tr>
<tr>
<td>scenes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Per Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>tokens</td>
</tr>
<tr>
<td>sentences</td>
</tr>
<tr>
<td>sentence tokens</td>
</tr>
</tbody>
</table>

Table 2: Statistics of the TRIPOD dataset; all means are shown with standard deviation in brackets.
in the dataset were then single annotated by the new annotators.

We computed inter-annotator agreement using two different metrics: (a) total agreement (TA), i.e., the percentage of TPs that two annotators agree upon by selecting the exact same sentence; and (b) annotation distance, i.e., the distance \( d[p_i, tp_i] \) between two annotations for a given TP, normalized by synopsis length:

\[
d[p_i, tp_i] = \frac{1}{N} |p_i - tp_i|
\]  

where \( N \) is the number of synopsis sentences and \( tp_i \) are the indices of the sentences labeled with TP \( i \) by two annotators. The mean annotation distance \( D \) is then computed by averaging distances \( d[p_i, tp_i] \) across all annotated TPs.

The TA between the two annotators in our second study was 64.00% and the mean annotation distance was 4.30% (StDev 3.43%). The annotation distance per TP is presented in Table 5 (last line), where it is compared with the automatic TP identification results (to be explained later).

We also asked our annotators to annotate the screenplays (rather than synopses) for a subset of 15 movies. This subset serves as our goldstandard test set. Annotators were given synopses annotated with TPs and were instructed to indicate for each TP which scenes in the screenplay correspond to it. Six of the 15 movies were doubly annotated, so that we could measure agreement. Since annotators were allowed to choose a variable number of scenes for each TP, this changes slightly our agreement metrics.

Total Agreement (TA) now is the percentage of TP scenes the annotators agree on:

\[
TA = \frac{1}{T \cdot L} \sum_{i=1}^{T \cdot L} \frac{|S_i \cap G_i|}{|S_i \cup G_i|}
\]

where \( T, L \) are the TPs identified per annotator in a screenplay, and \( S_i \) and \( G_i \) are the indices of the scenes selected for TP \( i \) by the two annotators.

Partial Agreement (PA) is the percentage of TPs where there is an overlap of at least one scene:

\[
PA = \frac{1}{T \cdot L} \sum_{i=1}^{T \cdot L} [S_i \cap G_i \neq \emptyset]
\]

And annotation distance \( D \) becomes the mean of the distances\(^2\) \( d[S_i, G_i] \) between two annotators normalized by \( M \), the length of the screenplay:

\[
d[S_i, G_i] = \frac{1}{M} \min_{i \in S_i, g \in G_i} |s - g|
\]

The TA and PA between the two annotators were 35.48% and 56.67%, respectively. The mean annotation distance was 1.48% (StDev 2.93%). The TA shows that the annotators rarely indicate the same scenes, even if they are asked to annotate an event in the screenplay that is described by a specific synopsis sentence. However, they identify scenes which are in close proximity in the screenplay, as PA and annotation distance reveal. This analysis validates our assumption that annotating the synopses first limits the degree of overall disagreement.

4 Turning Point Prediction Models

In this work, we aim to detect text segments which act as TPs. We first identify which sentences in plot synopses are TPs (Section 4.1); next, we identify which scenes in screenplays act as TPs via projection of goldstandard TP labels (Section 4.2); finally, we build an end-to-end system which identifies TPs in screenplays based on predicted TP synopsis labels (Section 4.3).

All models we propose in this paper have the same basic structure; they take text segments \( i \) (sentences or scenes) as input and predict whether these act as TPs or not. Since the sequence, number, and labels of TPs are fixed (see Table 1), we treat TP identification as a binary classification problem (where 1 indicates that the text is a TP and 0 otherwise). Each segment is encoded into a multi-dimensional feature space \( x_i \) which serves as input to a fully-connected layer with a single neuron representing the probability that \( i \) acts as a TP. In the following, we describe several models which vary in the way input segments are encoded.

4.1 Identifying Turning Points in Synopses

Context-Aware Model (CAM) A simple baseline model would compute the semantic representation of each sentence in the synopsis using a pre-trained sentence encoder. However, classifying segments in isolation without considering the context in which they appear, might yield inferior semantic representations. We therefore obtain richer representations for sentences by modeling their surrounding context. We encode the synopsis with a Bidirectional Long Short-Term Memory (BiLSTM; Hochreiter and Schmidhuber 1997) network; and obtain contextualized representation \( c_p \).
Multi-view TAM

Synopsis

1

3

N

TP1-... TP5-
Synopsis
encoder
Synopsis
encoder
Synopsis
encoder
Synopsis
encoder
Synopsis
encoder

(a) Topic-Aware Model (TAM)

<table>
<thead>
<tr>
<th>Sentence representations</th>
<th>Synopsis encoder</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>xN</td>
<td></td>
</tr>
</tbody>
</table>

Context-aware sentence representations

Context interaction layer

<table>
<thead>
<tr>
<th>Left context</th>
<th>Sliding window</th>
<th>Right context</th>
</tr>
</thead>
<tbody>
<tr>
<td>h̅i_b</td>
<td></td>
<td>h̅i_c</td>
</tr>
<tr>
<td>cp_i</td>
<td></td>
<td>lc_i</td>
</tr>
<tr>
<td>rc_i</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Context interaction layer

<table>
<thead>
<tr>
<th>sentence context similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>b_i = cp_i ⊙ lc_i</td>
</tr>
<tr>
<td>c_i = |cp_i</td>
</tr>
<tr>
<td>u_i = max(|cp_i</td>
</tr>
</tbody>
</table>

The interaction representation of sentence cp_i with its left context is the concatenation of cp_i, fl_i, and the above similarity values (i.e., b_i, c_i, u_i):

f_l_i = [cp_i; lc_i; b_i; c_i; u_i]

The interaction representation fr_i for the right context is computed analogously. We obtain the final representation of sentence i via concatenating fl_i and fr_i: y_i = [fl_i; fr_i; cp_i].

TP-Specific Information

Another variation of our model is to use TP-specific encoders instead of a single one (see Figure 2b). In this case, we employ five different encoders for calculating five different representations for the current synopsis sentence x_i, each one with respect to a specific TP. These representations can be considered multiple views of the same sentence. We calculate the interaction of each view with the left and right context window, as previously, via the context interaction layer. Finally, we compute the sentence representation y_i by concatenating its individual context-enriched TP representations.

Entity-Specific Information

We also enrich our model with information about entities. We first apply co-reference resolution to the plot synopses using the Stanford CoreNLP toolkit (Manning et al., 2005).
2014) and substitute mentions of named entities whenever these are included in the IMDb cast list. We then obtain entity-specific sentence representations as follows. Our encoder uses a word embedding layer initialized with pre-trained entity embeddings and a BiLSTM for contextualizing word representations. We add an attention mechanism on top of the LSTM, which assigns a weight to each word representation. We compute the entity-specific representation $e_i$ for synopsis sentence $s_i$ via the context interaction layer. The similarity between sentence $t_p$ and scene $s_i$ is computed by the TP–scene interaction layer.

**4.2 Identifying Turning Points in Screenplays**

Identifying TPs in synopses serves as a testbed for validating some of the assumptions put forward in this work, namely that turning points mark narrative progression and can be identified automatically based on their lexical makeup. Nevertheless, we are mainly interested in the real-world scenario where TPs are detected in longer documents such as screenplays. Screenplays are naturally segmented into scenes, which often describe a self-contained event that takes place in one location, and revolves around a few characters. We therefore assume that scenes are suitable textual segments for signaling TPs in screenplays.

Unfortunately, we do not have any goldstandard information about TPs in screenplays. We provide distant supervision by constructing noisy labels based on goldstandard TP annotations in synopses (see the description below). Given sentences labeled as TPs in a synopsis, we identify scenes in the corresponding screenplay which are semantically similar to them. We formulate this task as a binary classification problem, where a sentence-scene pair is deemed either “relevant” or “irrelevant” for a given TP.

**Distant Supervision** Based on the screenwriting scheme of Hague (2017), TPs are expected to occur in specific parts of a screenplay (e.g., the Climax is likely to occur towards the end). We exploit this knowledge as a form of distant supervision. We estimate the mean position for each TP using the gold standard annotation of the plot synopses in our training set (normalized by the synopsis length). The results are shown in Table 3, together with the TP positions postulated by screenwriting theory. We observe that our estimates agree well with the theoretical predictions, but also that some TPs (e.g., TP2 and TP3) are more variable in their position than others (e.g., TP1 and TP5). This leads us to the following hypothesis: Each TP is situated within a specific window in a screenplay. Scenes that lie within the window are semantically related to the TP, whereas all other scenes are unrelated. In experiments we calculate a window $\mu \pm \sigma$ based on our data (see Table 3).

We compute scene representations based on the sequence of sentences that comprise it using a BiLSTM equipped with an attention mechanism (see Section 4.1). The final scene representation $s$ is the weighted sum of the representations of the scene sentences. Next, the TP–scene interaction layer enriches scene representations with similarity values with each marked TP synopsis sentence $t_p$ as shown in Equations (5)–(7).

We again augment the above-described base model with contextualized sentence and scene representations using a synopsis and screenplay encoder. The synopsis encoder is the same one used for our sentence-level TP prediction task (see Section 4.1). The screenplay encoder works in a sim-
ilar fashion over scene representations.

**Topic-Aware Model (TAM)** TAM enhances our screenplay encoder with information about topic boundaries. Specifically, we compute the representations of the left $l_{ci}$ and right $r_{ci}$ context window of the $i^{th}$ scene in the screenplay as described in Section 4.1. Next, we compute the final representation $z_i$ of scene $s_i$ by concatenating the representations of the context windows $l_{ci}$ and $r_{ci}$ and the current scene $s_{ci}$: $z_i = [l_{ci}; s_{ci}; r_{ci}]$. There is no need to compute the similarity between scenes and context windows here as we now have gold-standard TP representations in the synopsis and employ the TP–scene interaction layer for the computation of the similarity between TPs and enriched scene representations $z_i$. Hence, we directly calculate in this layer a scene-level feature vector that encodes information about the scene, its similarity to TP sentences, and whether these function as boundaries between topics in the screenplay.

**Entity-Specific information** We can also employ an entity-specific encoder (see Section 4.1) for the representing the synopsis and scene sentences. Again, generic and entity-specific representations are combined via concatenation.

### 4.3 End-to-end TP Identification

Our ultimate goal is to identify TPs in screenplays without assuming any goldstandard information about their position in the synopsis. We address this with an end-to-end model which first predicts the sentences that act as TPs in the synopsis (e.g., TAM in Section 4.1) and then feeds these predictions to a model which identifies the corresponding TP scenes (e.g., TAM in Section 4.2).

### 5 Experimental Setup

**Training** We used the Universal Sentence Encoder (USE; Cer et al. 2018) as a pre-trained sentence encoder for all models and tasks; its performance was superior to BERT (Devlin et al., 2018) and other related pre-trained encoders (for more details, see the Appendix). Since the binary labels in both prediction tasks are imbalanced, we apply class weights to the loss function of our models. We weight each class by its inverse frequency in the training set (for more implementation details, see the Appendix).

**Inference** During inference in our first task (i.e., identification of TPs in synopses), we select one sentence per TP. Specifically, we want to track the five sentences with the highest posterior probability of being TPs and sequentially assign them TP labels based on their position. However, it is possible to have a cluster of neighboring sentences with high probability, even though they all belong to the same TP. We therefore constrain the sentence selection for each TP within the window of its expected position, as calculated in the distribution baseline (Section 4.2).

For models which predict TPs in screenplays, we obtain a probability distribution over all scenes in a screenplay indicating how relevant each is to the TPs of the corresponding plot synopsis. We find the peak of each distribution and select a neighborhood of scenes around this peak as TP-relevant ones. Based on the goldstandard annotation, each TP corresponds to 1.77 relevant scenes on average (StDev 1.23). We therefore consider a neighborhood of three relevant scenes per TP.

### 6 Results

**TP Identification in Synopses** Table 4a reports our results on the development set (we extracted 20 movies from the original training set) which aim at comparing various model instantiations for the TP identification task. Specifically, we report the performance of a baseline model which is nei-
ther context-aware nor utilizes topic boundary information against CAM and TAM. We also show two variants of TAM enhanced with TP-specific encoders (+ TP views) and entity-specific information (+ entities). Model performance is measured using the evaluation metrics of Total Agreement (TA) and annotation distance (D), normalized by synopsis length (equation (1)).

The baseline model presents the lowest performance among all variants which suggests that state-of-the-art sentence representations on their own are not suitable for our task. Indeed, when contextualizing the synopsis sentences via a BiLSTM layer we observe an absolute increase of 4.00% in terms of TA. Moreover, the addition of a context interaction layer (see TAM row in Table 4a) yields an absolute TA improvement of 4.00% compared to CAM. Combining different TP views further improves by 3.00%, reaching a TA of 39.00%, and reducing D to 6.52%.

Table 4b shows our results on the test set. We compare TAM, our best performing model against two strong baselines. The first one selects sentences that lie on the expected positions of TPs according to screenwriting theory; while the second one selects sentences that lie on the peaks of the empirical TP distributions in the training set (Section 4.2). As we can see, TAM (+ TP views) achieves a TA of 38.57% compared to 22.00% for the distribution baseline. And although entity-specific information does not have much impact on the development set, it yields a 2.76% improvement on the test set. A detailed break down of results per TP is given in Table 5. Interestingly, our model resembles human behavior (see row Human agreement): TPs 1, 4, and 5 are easiest to distinguish, whereas TPs 2 and 3 are hardest and frequently placed at different points in the synopsis.

We also conducted a human evaluation experiment on Amazon Mechanical Turk (AMT). AMT workers were presented with a synopsis and “highlights”, i.e., five sentences corresponding to TPs. We obtained highlights from goldstandard annotations, the distribution baseline, and TAM (+ TP views). AMT workers were asked to read the synopsis and rank the highlights from best to worst according to the following criteria: (1) the quality of the plotline that they form; (2) whether they include the most important events and plot twists of the movie; and (3) whether they provide some description of the events in the beginning and end of the movie. In Figure 4 we show, proportionally, how often our participants ranked each model 1st, 2nd, and so on. Perhaps unsurprisingly, gold-standard TPs were considered best (and ranked 1st 42% of the time). TAM is ranked best 30% of the time, followed by the distribution baseline which was only ranked first 26% of the time. Overall, the average ranking positions for the goldstandard, TAM, and the baseline are 1.87, 1.98, and 2.16, respectively. Human evaluation therefore validates that our model outperforms the position-based baselines.

### TP Identification in Screenplays

Our results are summarized in Table 6. For this task, we performed five-fold cross validation over our original goldstandard set to obtain a test-development split (recall we do not have goldstandard annotations for training). We report Total Agreement (TA), Partial Agreement (PA), and annotation distance D, normalized by screenplay length (Equations (2)–(4)).

Aside from the theory and distribution-based baselines, we also experimented with a common segmentation approaches such as TextTiling (Hearst, 1997) perform poorly on our task and we do not report them due to space constraints.

<table>
<thead>
<tr>
<th>Model</th>
<th>TA (%)</th>
<th>PA (%)</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory baseline</td>
<td>8.66</td>
<td>10.67</td>
<td>10.45 (9.14)</td>
</tr>
<tr>
<td>Distribution baseline</td>
<td>6.67</td>
<td>9.33</td>
<td>10.84 (8.94)</td>
</tr>
<tr>
<td>t<em>f</em>idf similarity</td>
<td>0.74</td>
<td>1.33</td>
<td>53.07 (31.83)</td>
</tr>
<tr>
<td>t<em>f</em>idf + distribution</td>
<td>4.44</td>
<td>6.67</td>
<td>13.33 (11.51)</td>
</tr>
<tr>
<td>CAM</td>
<td>11.11</td>
<td>16.00</td>
<td>10.23 (11.23)</td>
</tr>
<tr>
<td>TAM + entities</td>
<td>14.18</td>
<td>17.33</td>
<td>12.77 (12.61)</td>
</tr>
<tr>
<td>TAM End2end</td>
<td>10.63</td>
<td>13.33</td>
<td>8.94 (9.39)</td>
</tr>
<tr>
<td>TAM + entities</td>
<td>10.63</td>
<td>13.33</td>
<td>10.15 (10.56)</td>
</tr>
<tr>
<td>TAM End2end</td>
<td>7.87</td>
<td>9.33</td>
<td>10.16 (10.74)</td>
</tr>
<tr>
<td>Human agreement</td>
<td>35.48</td>
<td>56.67</td>
<td>1.48 (2.93)</td>
</tr>
</tbody>
</table>
mon IR baseline which considers TP synopsis sentences as queries and retrieves a neighborhood of semantically similar scenes from the screenplay using tf*idf similarity. Specifically, we compute the maximum tf*idf similarity for all sentences included in the respective scene. We empirically observed that tf*idf’s behavior can be erratic selecting scenes in completely different sections of the screenplay, and therefore constrain it by selecting scenes only within the windows determined by the position distributions ($\mu \pm \sigma$) for each TP. As far as our own models are concerned, we report results with goldstandard TP labels for CAM and TAM on their own and enriched with entity information. We also built and end-to-end system based on TP predictions from TAM.

As can be seen in Table 6, tf*idf approaches perform worse than position-related baselines. Overall, similar vocabulary across scenes and mentions of the same entities throughout the screenplay make tf*idf approaches insufficient for our tasks. The best performing model is TAM confirming our hypothesis that TPs are not just isolated key events, but also mark boundaries between thematic units and, therefore, segmentation-inspired approaches can be beneficial for the task. Results for entities are somewhat mixed; for CAM, the entity-specific information improves $TA$ and $PA$ but increases $D$, while it does not seem to make much difference for TAM. The performance of the end-to-end TAM model drops slightly compared to the same model using goldstandard TP annotations. However, it still remains competitive against the baselines, indicating that tracking TPs in screenplays fully automatically is feasible.

In Figure 5, we visualize the posterior distribution of various models over the scenes of the screenplay for the movie “Juno”. The first panel shows the distribution baseline alongside goldstandard TP scenes (vertical lines). We observe that the distribution baseline provides a good approximation of relevant TP positions (which validates its use in the construction of noisy labels, Section 4.2), even though it is not always accurate. For example, TPs 1 and 3 lie outside the expected window in “Juno”.

The second panel presents the TP predictions according to tf*idf similarity. We observe that scenes located in entirely different parts of the screenplay present high similarity scores with respect to a given TP due to vocabulary uniformity and mentions of the same entities throughout the screenplay. In the next panel we present the predictions of TAM. Adding synopsis and screenplay encoders yields smoother distributions increasing the probability of selecting TP scenes inside distinct regions of the screenplay, with sharper peaks and higher confidence.

7 Conclusions

We proposed the task of turning point identification in screenplays as a means of analyzing their narrative structure. We demonstrated that automatically identifying a sequence of key events and segmenting the screenplay into thematic units is feasible via an end-to-end neural network model. In future work, we will investigate the usefulness of TPs for summarization and question answering. We will also scale the TRIPOD dataset and move to a multi-modal setting where TPs are identified directly in video data.

Acknowledgments

We thank the anonymous reviewers for their feedback. We gratefully acknowledge the support of the European Research Council (Lapata; award 681760, “Translating Multiple Modalities into Text”) and of the Leverhulme Trust (Keller; award IAF-2017-019).
References


embeddings of words and entities from wikipedia. arXiv preprint 1812.06280.


A Model Details

Synopsis Encoder In all tasks we use a synopsis encoder in order to contextualize the sentences in the synopsis. We employ an LSTM network as the synopsis encoder which produces sentence representations $h_1, h_2, \ldots, h_T$, where $h_i$ is the hidden state at time-step $i$, summarizing all the information of the synopsis up to the $i$-th sentence. We use a Bidirectional LSTM (BiLSTM) in order to get sentence representations that summarize the information from both directions. A BiLSTM consists of a forward LSTM $\tilde{f}$ that reads the synopsis from $p_1$ to $p_N$ and a backward LSTM $\tilde{b}$ that reads it from $p_N$ to $p_1$. We obtain the final representation $cp_i$ for a given synopsis sentence $p_i$ by concatenating the representations from both directions, $cp_i = h_i = [\tilde{h}_i; \tilde{b}_i]$, $h_i \in \mathbb{R}^{2S}$, where $S$ denotes the size of each LSTM.

Entity-Specific Encoder This encoder is used to evaluate the contribution of entity-specific information to the performance of our models. We use a word embedding layer to project words $w_1, w_2, \ldots, w_T$ of the $i$-th synopsis sentence $p_i$ to a continuous vector space $\mathbb{R}^E$, where $E$ the size of the embedding layer. This layer is initialized with pre-trained entity embeddings. Next, we use a BiLSTM as described in the case of the synopsis encoder. On top of the LSTM, we add an attention mechanism, which assigns a weight $a_i$ to each word representation $h_i$. We compute the entity-specific representation $pe_i$ of the $i$-th plot sentence as the weighted sum of word representations:

$$e_j = \tanh(W_h h_j + b_h), \quad e_j \in [-1, 1]$$  

$$a_j = \frac{\exp(e_j)}{\sum_{l=1}^{T} \exp(e_l)}, \quad \sum_{j=1}^{T} a_j = 1$$  

$$pe_i = \sum_{j=1}^{T} a_j h_j, \quad e \in \mathbb{R}^{2S}$$

where $W_h$ and $b_h$ are the attention layer’s weights.

B Implementation Details

Pre-trained Sentence Encoder The performance of our models depends on the initial sentence representations. We experimented with using the large BERT model (Devlin et al., 2018) and the Universal Sentence Encoder (USE) (Cer et al., 2018) as pre-trained sentence encoders in all tasks. Intuitively, we expect USE to be more suitable, since it was trained in textual similarity tasks which are more relevant to ours. Experiments on the development set confirmed our intuition. Specifically, on the screenplay TP prediction task, annotation distance $D$ dropped from 17.00% to 10.04% when employing USE instead of the BERT embeddings in the CAM version of our architecture.

Hyper-parameters We used the Adam algorithm (Kingma and Ba, 2014) for optimizing our networks. After experimentation, we chose an LSTM with 32 neurons (64 for the BiLSTM) for the synopsis encoder in the first task and one with 64 neurons for the encoder in the second task. For the context interaction layer, the window $l$ was set to two sentences for the first task and 20% of the screenplay length for the second task. For the entity encoder, an embedding layer of size 300 was initialized with the Wikipedia2Vec pre-trained word embeddings (Yamada et al., 2018) and remained frozen during training. The LSTM of the encoder had 32 and 64 neurons for the first and second tasks, respectively. Finally, we also added a dropout of 0.2. For developing our models we used PyTorch (Paszke et al., 2017).

Data Augmentation We used multiple annotations for training for movies where these were available and considered reliable. The reasons for this are twofold. Firstly, this allowed us to take into account the subjective nature of the task during training; and secondly, it increased the size of our dataset, which contains a limited number of movies. Specifically, we added triplicate annotations for 17 movies and duplicate annotations for 5 movies.

C Example Output: TP Identification in Synopses

As mentioned in Section 6, we also conducted a human evaluation experiment, where highlights were extracted by combining the five sentences labeled as TPs the synopsis. In Tables 7, 8, and 9,
we present the highlights presented to the AMT workers for the movies "Juno", "Panic Room", and "The Shining", respectively. For each movie we show the goldstandard annotations alongside with the predicted TPs for TAM (+ TP views) and the distribution baseline, which is the strongest performing baseline with respect to the automatic evaluation results.

Overall, we observe that goldstandard highlights describe the plotline of the movie, contain a first introductory sentence, some major and intense events, and a last sentence that describes the ending of the story.

The distribution baseline is able to predict a few goldstandard TPs by only considering the relative position of the sentences in the synopsis. This observation validates the screenwriting theory: TPs, or more generally important events that determine the progression of the plot, are consistently distributed in specific parts of a movie. However, when the distribution baseline cannot predict the exact TP sentence, it might select one that describes irrelevant events of minor importance (e.g., TP4 for "Panic Room" is a detail about a secondary character instead of a major setback and highly intense event in the movie).

Finally, our own model seems to be able to predict some goldstandard TP sentences, as demonstrated during the automatic evaluation. However, we also observe here that even when it does not select the goldstandard TPs, the predicted ones describe important events in the movie that have some desired characteristics. In particular, for the movie "Juno" the climax (TP5) is the moment of resolution, where Vanessa decides to adopt the baby after all the setbacks and obstacles. Even though our model does not predict this sentence, it does select one that reveals information about the ending of the movie. An other such example is the movie "Panic Room", where the point of no return (TP3) is not correctly predicted, but the selected sentence refers to the same event.
In the movie "Panic Room", on the night the two move into the home, it is broken into by Junior, the previous owner's grandson; Burnham, an employee of the residence's security company; and Raoul, a ski mask-wearing gunman recruited by Junior.

Before the three can reach them, Meg and Sarah run into the panic room and close the door behind them, only to find that the burglars have disabled the telephone.

To make matters worse, Sarah, who has diabetes, suffers a seizure.

Sensing the potential danger to her daughter, Meg lies to the officers and they leave.

After a badly injured Stephen shoots at Raoul and misses, Raoul disables him and prepares to kill Meg with the sledgehammer, but Burnham, upon hearing Sarah's screams of pain, returns to the house and shoots Raoul dead, stating, "You'll be okay now", to Meg and her daughter before leaving.

---

Table 8: Highlights for the movie "Panic Room": goldstandard annotations and the predicted TPs for TAM (+ TP views) and distribution baseline.

---

In the movie "The Shining", Jack's wife, Wendy, tells a visiting doctor that Danny has an imaginary friend named Tony, and that Jack has given up drinking because he had hurt Danny's arm following a binge.

Hallorann tells Danny that the hotel itself has a "shine" to it along with many memories, not all of which are good.

When Wendy sees this in the bedroom mirror, the letters spell out "MURDER".

Jack investigates Room 237, where he encounters the ghost of a dead woman, but tells Wendy he saw nothing.

He kills Hallorann in the lobby and pursues Danny into the hedge maze.

---

Table 9: Highlights for the movie "The Shining": goldstandard annotations and the predicted TPs TAM (+ TP views) and distribution baseline.