Revista Romana de Interactiune Om-Calculator 9 (3) 2016, 233-254 

# Aggregating textual and video data from movies

Alexandru Hulea, Traian Rebedea

University Politehnica of Bucharest
Splaiul Independentei nr. 313, sector 6, 060042, Bucuresti
*E-mail: alexandru.hulea@gmail.com, traian.rebedea@cs.pub.ro*

**Abstract.** In this paper, we present an automatically annotated corpus[2] based on movie screenplays (script) and subtitles. We extract the relevant textual information from movie screenplays and subtitles using a regular expression approach. Then, we synchronize screenplays with subtitles using a matching algorithm, thus bounding each sentence from a script between two temporal limits. We also developed an application using the corpus to test our approach and to show practical situations where this corpus is useful. The application employs topic detection and it involves searching for a specified topic in the movie text and marking the topic as non-existent, episodic or primary topic for the analyzed text. The major problem we faced while working on this system was the unexpected structure of the screenplay sheets as this kind of files are not entirely written using a standardized format which can be easily parsed and structured automatically. Some types of errors can be overcome with regular expressions, but there are other errors that need a machine learning approach to be surpassed.

**Keywords**: Video-text annotation, Corpus creation, Video understanding, Topic detection, Information retrieval.

## 1. Introduction

Processing data from movies, either textual, audio or video became lately an interesting field mostly because of the hardware improvement and development of machine learning techniques for processing large volumes of data, especially deep learning (Zou, Zhu, Ng, and Kai, 2012). One of the reasons why this field wasn't so interesting until recently could also be the lack of standard corpora, as movie data (screenplays, subtitles) do not have a completely standardized structure.

Modern movies can roughly be defined as having two main textual

[2] The corpus presented in this paper is avaiable online at the following address: https://github.com/HuleaAlexandru/MovieTextCorpus.

components (a screenplay/script and a subtitle), an audio component and a video component. While subtitles are synchronized and marked with timestamps from the movie, they contain only the dialogue between characters and do not provide any details about the action of the movie. Thus we decided to extract more textual information from the screenplay and to give temporal boundaries to every sentence in a screenplay.

While a subtitle is organized in dialogue turns, each turn being marked with a timestamp, a screenplay is organized in scenes and transitions which mark the jump from a scene to another. A scene begins by specifying the place where the scene takes place, sometimes giving some extra temporal information, like the year, the month or the day when a scene takes place and the time of the day (e.g. day, night, sunset). A scene also contains a descriptive part where the characters participating in that scene are mentioned, the action is recounted and the location where the scene takes place is described. It also mentions the dialogue between characters and sometimes the script specifies the writer's directions about how the scene should be played.

In this paper, we present a method for extracting relevant data from screenplays by splitting the text in sentences and assigning a type to each sentence. After studying previous work in this domain (Turetsky and Dimitrova, 2004, Agarwal et al., 2014), we concluded that each sentence from a screenplay can be assigned to one of the following five main types: scene boundary, scene description, character name, dialogue, and metadata – each type having one or several subtypes. After we assign a type to each sentence from the screenplay we used the sentences marked as dialogue to synchronize the subtitle with the screenplay. The algorithm used for synchronization uses a Greedy strategy and it is one of the main contributions of this paper. Once the corpus construction is finished, we use the dataset for a topic detection application which uses a semantic network to create a taxonomy for a chosen topic and then we search the words from the taxonomy to identify the topic in the text.

In our preliminary research we discovered several useful applications developed based on an annotated and synchronized movie corpus and our goal is to provide such a reliable corpus. The most relevant of these applications are: an action recognition project (Laptev, Marszałek, Schmid, and Rozenfeld, 2008), a system which improves the quality of the audio description and screenplays for the blind people (Rohrbach, Rohrbach,

Tandon, and Schiele, 2015) and a movie summarizer application (Tsoneva, Barbieri, and Weda, 2007).

The paper is structured as follows. Section 2 presents the most important works in movie screenplay structuring and building corpora of video-text aligned data from movies. In Section 3 we detail the corpus development process, focusing on structuring screenplay text files and then aligning screenplays and subtitles. Next, Section 4 introduces the topic detection in movies application developed using the created corpus, while Section 5 offers details about the accuracy of the proposed methods and algorithms used for creating the corpus. The paper ends with conclusions and proposes several improvements.

## 2. Related work

The first steps in processing textual data from movies were by Turetsky and Dimitrova (2004). They present the screenplays organized in scenes and each scene beginning with a "slug line" which contains a marker INT / EXT which specifies if the scene is taking place inside or outside. The slug line also specifies some spatial and temporal boundaries. They observed that each scene begins like this and continues with the description of the scene and action, along with the presentation of the characters which participate in the scene. Finally, the scene contains the dialogue between characters, where it is specified which character is uttering each turn and these are highlighted the keywords "V.O", "O.S" which means that the speaker is off-screen. There are identified action directions for actors and transitions by using keywords like "FADE IN", "DISSOLVE TO", "CUT TO".

They parsed the screenplay based on these observations using a regular expression approach. Then they employed a dynamic programming strategy to synchronize the subtitle with the screenplay and by doing that, they exposed a lot of possible applications that can be developed having a screenplay annotated with temporal boundaries.

In the paper are exposed some problems like the semi-regular format for screenplays meaning that the screenplays don't respect the same format and thus can't be easily parsed. Another highlighted problem is represented by the scenes added, deleted or modified when filming the movie, which makes subtitles sometimes contain different turns from the original movie script.

This problem can be easily seen in the synchronization part where there are sentences in screenplay without an adequate correspondent in the subtitle. These problems along with proposed solutions will be analyzed in the next sections.

A more recent paper published in this field was Agarwal et al. (2014). Their work builds on the ideas presented by Turesky and Dimitrova (2004). The project exposes a classification of screenplay lines in five classes: scene boundary, scene description, character name, dialogue, and metadata. They also presented two strategies used to assign a class to each line from screenplay: a regular expression approach and a machine learning approach.

In the paper it is defined a screenplay as well-structured if the top three unique indentation sums up more than 90% of the lines. Knowing that a screenplay is well-structured, a class can be assigned using only the indentation. They observed that the indentation for the character name class is the greatest and the indentation for the dialogue class is greater than the indentation for scene boundary and scene description classes. Because the scene boundary and scene description classes have the same indentation, they can only be identified based on the keywords which appear in the scene boundary lines, keywords like INT / EXT. For the rest of the lines they assigned the metadata class. To check the sanity of the assignments they introduced the idea of structural constraints. They said that a scene description must be contained between a scene boundary line and a character name line, the dialogues should be between character name lines and all character names must be within two scene boundaries. We find this idea very useful not only to check the sanity of the assignments, but also to set some structural constraints before assign a class to a line.

In a screenplay which is not well-structured, the regular expression approach firstly identifies, based on keywords, the scene boundary lines (INT / EXT), the character name lines (V.O, O.S), and metadata lines (CUT TO, DISSOLVE TO, FADE IN). After using the keywords, the character name class is set to lines with no more than three words, all in uppercase. Having set the scene boundary and character name lines, they can also set the dialogue and the scene description lines based on the structural constraints previously mentioned. The lines which are between a scene boundary line and a character line will be tagged as a scene description line and those between two character name lines or between a character name line and a scene boundary line will be tagged as a dialogue line. This way

the classification doesn't need a well-structured screenplay but the precision is a bit lower comparing with a well-structured screenplay.

Then the authors observed some limitations which couldn't be overcome with this regular expression approach, so they decided that a machine learning approach will be more suitable. The main limitations of the regular expression approach presented in the paper, limitations observed in our own research, are: the missing of the specific keywords for scene boundary lines (INT/EXT) and the scene boundary and character name lines not being always capitalized.

Laptev (2008) and Rohrbach et al. (2015) demonstrate the need of an annotated corpus based on textual data from movies. Both papers present two practical applications based on this such a corpus: an action recognition from movies system and an application which raises the quality of the audio description and screenplays for blind people. These projects use the information from the synchronized screenplays together with the video component of the movie to extract important information. Besides extracting particular information, we see in these papers the corpus being used as a provider of data for several computer vision applcations. An example presented by Laptev (2008) is to use the textual data to identify actions from movies and then to validate them with computer vision techniques. These two papers show the need of a reliable corpus which provides information about both the text and video of a movie which can later be the input for other practical applications in natural language processing and computer vision (or a combination of both).

## ➢ 3. Corpus creation

The entire process of creating the corpus is organized in three main parts: data collection, annotation and synchronization. For every movie, the process starts with a subtitle file and a screenplay file and ends with both files annotated and synchronized with timestamps.

### ➢ 3.1 Data collection

The first part of creating the corpus is the information gathering. We keep a file which contains the links for the screenplay and subtitle files and the

name of the movie.

We collected the subtitle files from the website Open Subtitles (http://www.opensubtitles.org) which provides an API using the XML-RPC protocol to communicate with other programs. Using this protocol we used HTTP requests to search for a specific subtitle file based on the movie name and the link from the initial file.

The screenplay files were found on the International Movie Script Database (http://www.imsdb.com). To download a file, only a single HTTP GET request was needed.

We chose 80 movies to create the initial database, but one can add a movie to the database only by writing all the required data in an input file: the name of the movie and two links to the corresponding screenplay file and subtitle file from the aforementioned sites.

After downloading the subtitle and screenplay files, we removed formatting tags, in this case HTML, CSS and JavaScript tags, and obtained two text files with only the textual content of the subtitle and movie script.

### ➢ 3.2 Annotation process

The process of annotation starts from the ideas presented in Section 2, but we also developed original methods, as well as new particularities, based on the observed performance of the other approaches. These will be further detailed in this section.

For the subtitle file, the annotation process is performed by merely splitting the text in sentences and assigning each sentence two temporal boundaries (the time for the current turn in the subtitle and the time of the subsequent turn), a very easy task considering the format of a subtitle file.

For the screenplay file, the annotation is troublesome, mostly because of the semi-regular format of these files. We chose to split the text into lines and assign to each line one of the following five classes: scene boundary, scene description, character name, dialogue, and metadata. We further split the metadata class in three subtypes: transition between scenes, author direction, and undefined. The scene boundary lines are the first lines in a scene and contain a specific marker (INT / EXT) to specify if the scene is taking place indoors or outside, the place where the scene takes place and the time of the day (day, night). The scene description lines recount the action, describe the scene and introduce the characters which participate at

the scene. The character name lines provide the list of characters who are about to speak, the dialogue lines represent the speech of each character and the metadata lines are all the remaining lines. The metadata lines can be transitions between scenes, author directions which express the way a character should play the scene and the undefined type represents the lines with no relevant information like a scene number or the name of the movie. After assigning a class to each line in the script, we create blocks from consecutive lines that have the same class and then split the blocks into sentences, assigning to each sentence the class of the block. Figure 1 shows an example of an annotated scene from the script of the movie "Memento" which was created using the method proposed in this section.

Because the screenplay files don't have a regular format we decided to annotate them using two diffent strategies: one using only structural constraints and keywords and another using the indentation as a constraint as well if the file is well indented.

A screenplay file is considered well indented if the first five most common indentation sums up more than 95% of lines. We observed that the lines from each class have the same indentation level for a well indented screenplay, while sometimes scene boundary class and scene description class having the same indentation.

```
{SCENE_BOUNDARY}: 7 EXT. DISCOUNT INN CAR PARK - DAY <<COLOUR SEQUENCE>>
{SCENE_DESCRIPTION}: Teddy starts for a GREY SEDAN.
{SCENE_DESCRIPTION}: Leonard pauses behind him.
{CHARACTER_NAME}: LEONARD
{DIALOGUE}: My car.
{SCENE_DESCRIPTION}: Teddy glances back in surprise.
{CHARACTER_NAME}: TEDDY
{DIALOGUE}: This is your car.
{CHARACTER_NAME}: LEONARD
{META_DATA}: (shakes head)
{DIALOGUE}: You're in a playful mood.
{SCENE_DESCRIPTION}: Leonard holds up a Polaroid of a late model JAGUAR.
{CHARACTER_NAME}: LEONARD (cont'd)
{DIALOGUE}: Shouldn't make fun of somebody's handicap.
{SCENE_DESCRIPTION}: Teddy smiles and heads for the BRAND-NEW JAGUAR parked
several cars further down.
{CHARACTER_NAME}: TEDDY
{DIALOGUE}: Just trying to have a little fun.
{SCENE_BOUNDARY}: 8 INT. CAR - DAY <<COLOUR SEQUENCE>>
{SCENE_DESCRIPTION}: Leonard drives, Teddy admires the new car' s interior,
reaching down around the seats, exploring the car with his hands.
{CHARACTER_NAME}: TEDDY
{DIALOGUE}: Roll your window up, will ya?
{SCENE_DESCRIPTION}: Leonard hits his window button.
{SCENE_DESCRIPTION}: A few fragments of safety glass rise out of the door,
remnants of a broken window.
{CHARACTER_NAME}: LEONARD
{DIALOGUE}: It's broken.
{SCENE_DESCRIPTION}: Teddy looks, curious.
{CHARACTER_NAME}: TEDDY
{DIALOGUE}: I can get that fixed for you.
{SCENE_DESCRIPTION}: Leonard shrugs.
{CHARACTER_NAME}: TEDDY (cont'd)
{DIALOGUE}: So where are we going, Sherlock?
{META_DATA}: (CONTINUED)
{SCENE_DESCRIPTION}: MEMENTO Pink Revisions - 9/7/99 6A.
```

Figure 1. Annotated scene from the movie “Memento”

We used three techniques to assign a class to a line: based on keywords and markers, using structural constraints and using indentation constraints. We identified specific keywords for some classes: INT, EXT for scene boundary, V.O, O.S, CONT’D for character name, FADE IN, DISSOLVE TO, CUT TO for metadata, transition type. A line enclosed by brackets is a metadata line and if it is following a character name line or a dialogue line that line has the author direction type.

The structural constraints tell us that after a scene boundary line a scene description line will be next or after a character name line a dialogue line or a metadata line, author direction type will follow. The lines before the first

scene boundary can be only metadata or scene description lines. We identified the dialogue lines as the block which follows a character name line, but the block can contain author direction lines which can be identified as previously explained. The scene description class was assigned to the lines between scene boundary lines and character name lines, between two blocks of dialogue and between a dialogue block and a scene boundary line. Finally, the character name lines and the scene boundary lines are always capitalized and the character name lines usually have less than four words.

The indentation constraints only restrict us to assign a class to a line that has a different indentation level than the class indentation level.

The novelty of this approach, besides using indentation as a constraint and some structural constraints, consists in the idea of blocks. We call a block multiple lines of the same type, generally enclosed by empty lines. The blocks usually have the dialogue or scene description type, but there are metadata or scene boundary blocks as well. We identified some special cases where, between a character name line or an author direction line and a dialogue block are no empty lines. A similar case we discover between some scene boundary lines and scene description lines. These cases are not a problem though, because we identify from the start, the scene boundary lines, the character name lines and the author direction lines and we use the blocks to separate the dialogue lines from the scene description lines.

To summarize, the scene boundary lines can be identified only by searching for the capitalized lines which contains INT/EXT keywords. We discovered the character name lines using the specific keywords and the less than four words rule and the metadata lines based on keywords and markers. After assigning these three classes to the specific lines, we can find the dialogue lines based on character name lines and the scene description lines based on all the other classes. The main steps gathered in a pseudocode are:

- Set metadata class, transition type based on keywords;
- Set metadata class, author direction and undefined type based on parenthesis;
- Set scene boundary class based on keywords;
- Set character name class based on keywords;
- Set character name class based on less than four words rule;

- Set dialogue class for each block which follows a character name line;
- Set scene description class based on structural constraints.

The strategy that involves indentation constraints is a little bit different from the previous one. It discovers the indentation level of each class and then uses it as an additional constraint. The main steps are:

- Set metadata class, transition type based on keywords;
- Set metadata class, author direction and undefined type based on parenthesis;
- Determine the indentation level of scene boundary class based on the capitalized lines which contain the scene boundary keywords;
- Set scene boundary class based on keywords and indentation;
- Determine the indentation level of character name class based on the capitalized lines which contain the character name keywords or respect the less than four words rule;
- Set character name class based on indentation and keywords or less than four words rule;
- Determine the indentation level of dialogue class based on the first lines without class which follow character name lines;
- Set dialogue class for the lines without class which follow character name lines and have the proper indentation level;
- Determine the indentation level of scene description class based on the first lines without class which follow scene boundary lines;
- Set scene description class for the lines without class which are between scene boundary lines and character name lines or between two dialogue blocks or between a dialogue block and a scene boundary line and have the proper indentation level;
- Set the class for the lines without class based only on indentation;
- Set the class based on the class block which contains the line.

After this annotation step, we ran sanity checks based on the structural constraints and some movies failed the test because screenplays often contains errors like omitting one's dialogue turns or the character name of the speaker. We also used Stanford CoreNLP (Manning et al., 2014) to tokenize and annotate each word with its part of speech which is useful for the topic detection application described in Section 4.

### ➢ 3.3 Synchronization between subtitle and screenplay

Once the screenplay sentences have been split into the five main classes, our goal is to assign for as many dialogue sentences as possible a relevant turn from the corresponding subtitle file. The only restriction for the assignments is that having two dialogue turns A1 and A2 and two subtitle turns B1 and B2 and A1 appears before A2, B1 appears before B2, we are not allowed to assign A1 to B2 and A2 to B1, both at the same time. We call this an inconsistency and we illustrate this case in Figure 2.

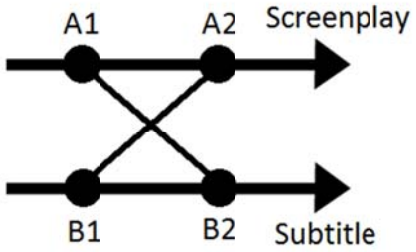



Figure 2. A typical example of inconsistency between screenplay and subtitle turns

As previously mentioned, dialogue turn - subtitle turn pairs have to be relevant, meaning that they should contain similar or identical texts. The relevance is determined using heuristics and the assignments can be adapted to be more or less relevant. To find the relevant pairs we used Elasticsearch (Gormley and Tong, 2014), a fast and optimized full text search distributed server. It also has several different algorithms to determine the relevance for a query in the indexed documents. Thus, we indexed the subtitle turns in Elasticsearch and then searched each dialogue turn.

The specific Elasticsearch methods used for ranking the search results are: match phrase, match and common. The *match phrase* ranking is the

most restrictive one as it returns a match only if more consecutive words from the searched text are found in an indexed text. The *match* search method returns all texts which have at least one common word with the query and the results are ranked based on the number of matched words and whether the words have the same order in the query and the indexed document. The *common* search uses a tf-idf English analyser (Manning and Schütze, 2009) which offers more relevance to the sentences which contain rare words from the query.

We then developed a Greedy algorithm to assign as many dialogue turns from a script to a subtitle turn. The ideal case would be if all dialogue turns from the screenplay would appear in the subtitle and all subtitle turns could be found in the dialogue turns. As concluded in the related work section, this case is very rare because the subtitle and the screenplay don't always contain the same scenes. In the subtitle we see scenes that don't exist in the screenplay, modified scenes or some scenes that are in the screenplay and were eliminated in the movie and the subtitle. We also observed that there are some turns in the subtitle and the screenplay, with the same meaning, but written with different words. This is also the reason why we cannot find for all dialogue turns an adequate match in the subtitle.

The proposed Greedy algorithm can be summarized as follows: while there is a relevant match between a dialogue and a subtitle turn, find the most relevant pair which doesn't break the consistency constraint and keep that match. An optimized way to ensure that the consistency constraint won't be broken is to assign each turn two boundaries where it can be searched. For example, if we have three dialogue turns A1, A2, A3 which appear in this order in a screenplay and B1, B2 – 2 subtitle turns in this order in the subtitle and the set of existing matches A1–B1 and A3–B2, we can search for a match for A2 only between B1 and B2 as illustrated in Figure 3.

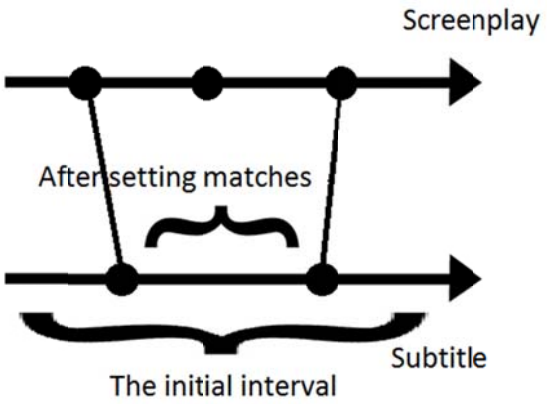



Figure 3. The bounding Greedy search method

The most relevant pair is determined based on the score function of each ranking/search algorithm. Because the proposed algorithm is time consuming, we chose to keep at each step only the matches with a relevance above a threshold. At each step, we lower the threshold to accept matches less and less relevant until reaching a minimum threshold.

While allowing more matches at each step determines the algorithm to run faster, this approach can produce inconsistencies between two matches chosen at the same step. We developed four strategies to eliminate these inconsistencies. As we cannot keep both matches that produce an inconsistency, we have to choose if we keep one match or none. Four strategies are used in this situation:

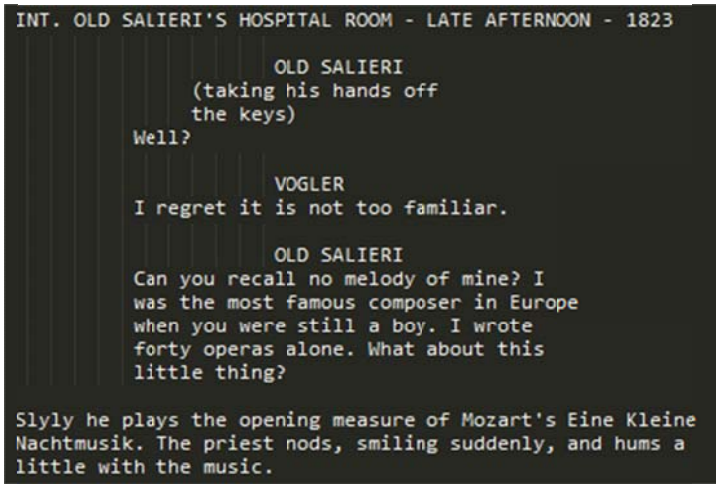
- Keep none of the matches because they have similar relevance scores and we can't decide which match is the correct one.
- Keep the match with a higher relevance score.
- Keep the match which determines less inconsistencies with the other matches in the current step and, if equal, keep none.
- Keep the match which determines less inconsistencies with the other matches in this step and, if equal, keep the most relevant one.

After we have assigned timestamps for some of dialogue turns in the

script, we are able to assign temporal boundaries to each turn. Of course, the higher the matching percentage between dialogue and subtitle turns is, more precise temporal boundaries we will obtain for each part of the script. For each sentence B from the screenplay, without assigned temporal boundaries, we search for the first preceding sentence with temporal data – A with boundaries *A-start* and *A-end*, and for the first subsequent sentence with boundaries on the right – C between *C-start* and *C-end*. Thus, for sentence B we will set the temporal boundaries between *A-start* and *C-end*.

An example that shows how the system works is presented in the following figures. Figure 4 contains the screenplay for the movie "Amadeus", while Figure 5 shows how the text is annotated following the steps presented in this section. Figure 6 presents the synchronization annotation of a sample from the movie script of "Memento" together with its corresponding subtitles. In this figure we also highlight the matches relevant enough to be detected by our program. The used search method was a combination of *match phrase*, *match* and *common* search.

After synchronization, we run a test and keep only the movies that have more than 33% of the dialogue turn from the screenplay matched with a corresponding subtitle turn. We considered that a lower percentage affects the precision of the temporal boundaries for too many sentences. This means that the subtitle and the screenplay are too different as previously detailed. The scenes from the screenplay added, deleted or modified when filming the movie and thus also in the subtitle file have a major impact on this result. The differences between two sentences from the subtitle and the screenplay, with the same meaning but written with in a different manner also degrade the quality of the synchronization.



```
INT. OLD SALIERI'S HOSPITAL ROOM - LATE AFTERNOON - 1823

                    OLD SALIERI
               (taking his hands off
               the keys)
          Well?

                    VOGLER
          I regret it is not too familiar.

                    OLD SALIERI
          Can you recall no melody of mine? I
          was the most famous composer in Europe
          when you were still a boy. I wrote
          forty operas alone. What about this
          little thing?

Slyly he plays the opening measure of Mozart's Eine Kleine
Nachtmusik. The priest nods, smiling suddenly, and hums a
little with the music.
```

Figure 4. Original script for the movie *Amadeus*

```
{SCENE_BOUNDARY}: INT. OLD SALIERI'S HOSPITAL ROOM - LATE AFTERNOON - 1823
{CHARACTER_NAME}: OLD SALIERI
{META_DATA}: (taking his hands off the keys)
{DIALOGUE}: Well?
{CHARACTER_NAME}: VOGLER
{DIALOGUE}: I regret it is not too familiar.
{CHARACTER_NAME}: OLD SALIERI
{DIALOGUE}: Can you recall no melody of mine?
{DIALOGUE}: I was the most famous composer in Europe when you were still a
boy.
{DIALOGUE}: I wrote forty operas alone.
{DIALOGUE}: What about this little thing?
{SCENE_DESCRIPTION}: Slyly he plays the opening measure of Mozart's Eine
Kleine Nachtmusik.
{SCENE_DESCRIPTION}: The priest nods, smiling suddenly, and hums a little
with the music.
```

Figure 5. Extractd annotation of the movie *Amadeus*

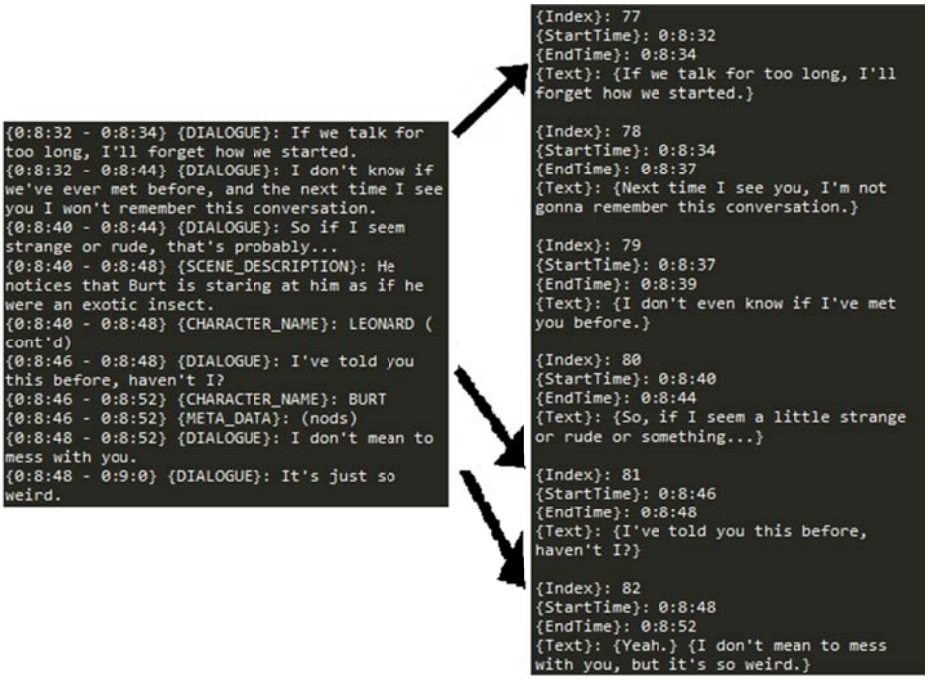



Figure 6. Example of subtitle-script synchronization for the movie *Memento*

## 4. Topic detection in movies

In order to test the created corpus, we developed an application which splits the screenplay of a movie in scenes based on the scene boundary lines and then searches in each scene the words from a taxonomy that define a topic. Each word that describes a topic has a relevance score as some words describe better the current topic. At the end, we check if the sum of the score of all words defining a topic and appearing in the current scene is greater than a threshold to determine if a topic is relevant or not for each scene. We count the number of scenes and if the number of scenes where the topic is found is lower than 3% of the total number of scenes we will mark the topic as non relevant for the movie. If the number of scenes where the topic is found is greater than 10% of the total number of scenes the topic is considered a primary topic for the movie, otherwise the topic is marked as episodic.

WordNet (Miller, 1995, Fellbaum, 1998) is a linguistic semantic network, a tool which keeps pairs of words – called synsets – semantically related, based on specific relationship between the two words. The main relationships are hyponymy and hypernymy, these relationships have associated a higher relevance too when defining a topic. A word X is a hyponym of a word Y if X is more specific than Y. The hypernymy relationship is the opposite relationship. A word X is a hypernym of a word Y if X is more generic than Y. An example for these two relationships is the pair "weapon" – "gun", "gun" is a hyponym for "weapon" and "weapon" is a hypernym for "gun". We used some other relationships with a lower relevance score like part – whole relationships (e.g. "wing" – "plane"). Figure 7 presents the relationships between the first synset of the word "car" and other synsets extracted from WordNet.

A topic is defined starting from a seed word and then we use WordNet relationships to find other words semantically related with the initial one. Because we start with some initial keywords and their meaning, we create a taxonomy by adding some other words obtained by using the relationships already exposed. For each word added in the taxonomy that we create, we also keep its part of speech. When we search for a word in a scene, we compare both the word and its part of speech.

In Clifton, Cooley and Rennie (1999) the search is somehow similar, meaning that they only look for the words that have the primary meaning the same with a keyword. This way the meaning of the words in the taxonomy is related with the topic. The paper says that 80% of the words are used with their primary meaning in texts. Using the part of speech constraint, we look for a word with a specific part of speech, fact that raises the precision above 80%.

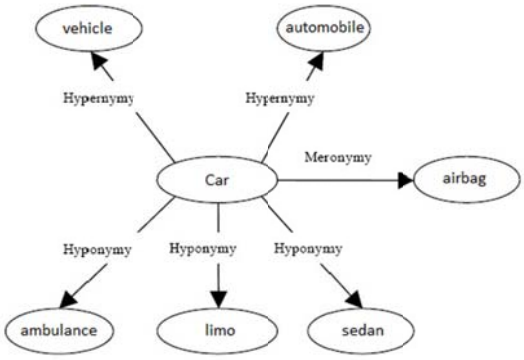



Figure 7. Example of relations from the WordNet ontology

## 5. Results and analysis

We downloaded 80 movies in the data collection step, but only 55 of them passed all the tests presented in Section 3. Four movies didn't meet the structural constraints because they omitted dialogue blocks, scenes or character names, four movies didn't have the markers INT / EXT in the scene boundary lines and couldn't be correctly identified and the rest had synchronization problems. We considered that a movie has synchronization problems between subtitle and screenplay if less than 33% of the dialogue turns from the screenplay have a match in the subtitle.

We manually checked the classification of sentences in the five classes before mentioned and we got 98% precision for the movie "Memento", a well indented screenplay and 95% precision for the movie "American Beauty" a not so well indented movie screenplay. We conclude that the well indented movies tend to be more precise because the indentation constraint usually leads to the right classification.

We tried several search/ranking algorithms from Elasticsearch using the same method to avoid inconsistencies, keeping the match with less inconsistencies and if equal keeping the most relevant one (method 4 from Section 3). We ran a test on five search methods for the movie "Forrest Gump" and the results are presented in Figure 8. We conclude that *match phrase* is the most restrictive search and it provides the most relevant

matches, while the *match* search and the *common* search offer less relevant matches as well. We tried some combination too and we conclude that the best search algorithm is the *match phrase + match search + common search* combination because we set the most relevant matches first, and doing so, we narrow the search interval, so the less restrictive searches that follow won't make many mistakes.

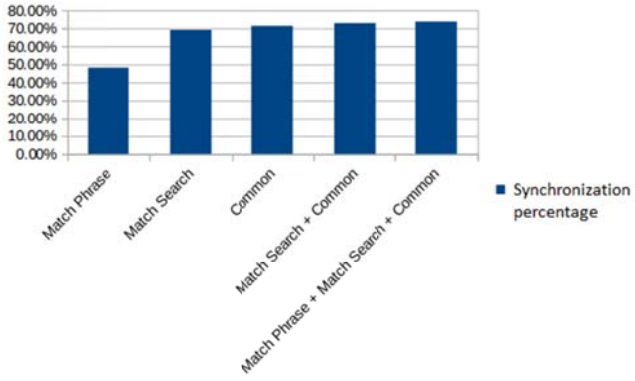



Figure 8. Performance for the diferent ranking/search algorithms

We also ran some tests on the methods used to avoid inconsistencies for the movie "Memento" and the results are depicted in Figure 9.

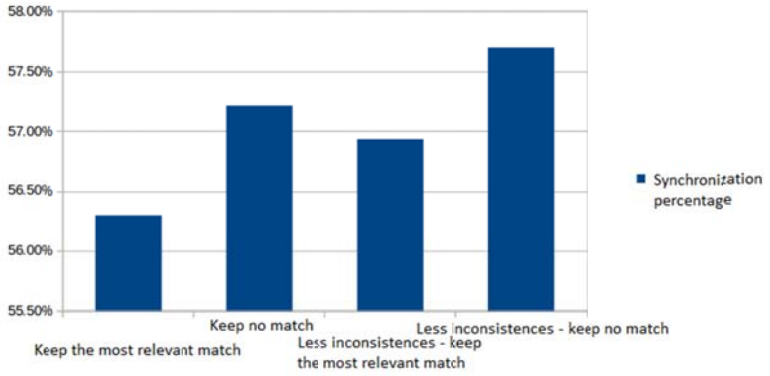



Figure 9. Performance for the methods that avoid inconsistencies

We didn't obtain very large differences between the methods mainly because the number of inconsistencies is around 30-40 in the whole process of synchronization. While we analyzed the charts, we observed that the methods which don't keep any match are better than the methods that keep the match with a higher relevance score. An explanation for this result could be that choosing a match from an inconsistency will narrow the search interval for the next steps and that match could be an incorrect one, so when we get an inconsistency it is better not to choose any of the matches.

For the movies that pass all the tests, using the best search algorithm (the combination of *match phrase*, *match* search and *common* search) and the best method to avoid inconsistencies, we got the distribution of the synchronization accuracy presented in Table 1.

Table 1. Synchronization percentage distribution

| Synchronization range (%) | Number of movie screenplays |
|---|---|
| 30 – 40 | 9 |
| 40 – 50 | 13 |
| 50 – 60 | 12 |
| 60 – 70 | 10 |
| 70 – 80 | 4 |
| 80 – 90 | 7 |

For the topic detection application we ran a test for the topic "crime" and we started with the words "murder" as noun and verb, "gun" and "violent". After using the WordNet ontology we obtained some other relevant words like "weapon", "shoot down", "kill", "barrel", "hostile", "trigger", "homicide" with a high score of relevance, but some irrelevant words too like "instrument" with a lower score. There are about 5% - 10% cases where the detection of the topic went wrong. The main problem that was identified is that the taxonomy extracted from WordNet doesn't contain the words related with this topic ("crime"). Another problem is that the topic didn't exist in the scene, but there were a lot of irrelevant words that existed in the taxonomy and even if these words have a lower relevance score, if we find a lot of them, the total relevance score will be higher than the established threshold. An example of a scene where the crime topic is identified can be seen in the Figure 10. We identified the "crime" topic as a primary topic in the movie "Memento" because we find it in 29 out of 186 scenes.

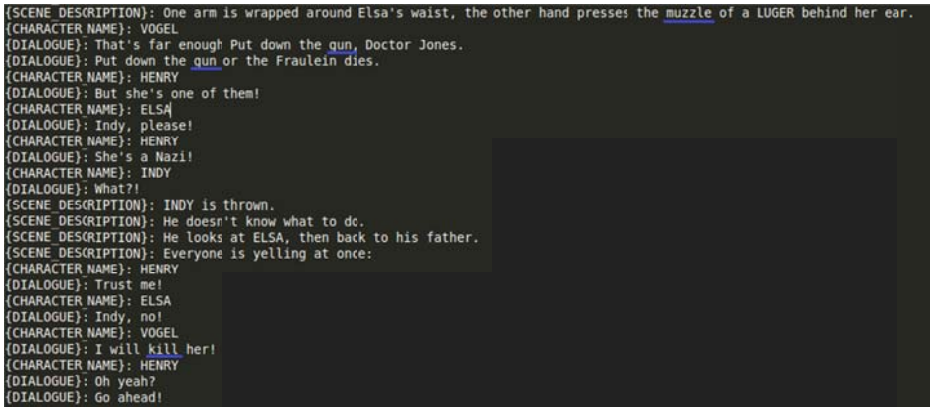



Figure 10. Topic identification example

## 6. Conclusions and future work

In this paper we presented the process of annotation and synchronization of subtitle and screenplay files for movies. At the end of this process, we obtained a corpus with textual information from movies which is also partially timestamped and aligned with the video of each movie. We have also used the developed corpus for a topic detection application that uses the classification of the screenplay sentences in five classes and the part of speech of each word.

While working on this project, we have discovered some problems as about 30% of the initial set of movie screenplays didn't pass all the tests for the synchronization to be considered successful. The main problem we identified is the bad synchronization between texts in the subtitle and screenplay files for the same movie. A good way to solve this problem would be to use an updated or better aligned/formatted screenplay. More, using the modified version of the screenplay after which the movie was shot would also be useful as there are lots of screenplay adaptation made by the producer or director of a movie. However, for most movies with this kind of problem we were not able to find the updated version of the screenplay online – in many cases, this version is not even updated or recorded during the filming process. Another problem that was also identified in previous work on this topic is the problem related to the structure of movie scripts, the missing markers from the scene boundary sentences and the character

name lines not being capitalized. This problem could be solved using a machine learning approach.

In the future we shall try a machine learning approach for solving these structural problems to improve the precision of classification of sentences in the script into the various classes presented in this paper. We think of using this corpus in some other applications with a more practical purpose such as automatic captioning of movies for people with disabilities.

## References

Agarwal, A., Balasubramanian, S., Zheng, J., & Dash, S. (2014). Parsing Screenplays for Extracting Social Networks from Movies. *EACL*, (pp. 50-58).

Clifton, C., Cooley, R., & Rennie, J. (1999). TopCat: Data Mining for Topic Identification in a Text Corpus. *the 3rd European Conference of Principles and Practice of Knowledge Discovery in Databases*, (pp. 20-23).

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database* . Cambridge: MIT Press.

Gormley, C., & Tong, Z. (2014). *Elasticsearch - The Definitive Guide*. Retrieved from https://www.elastic.co/guide/en/elasticsearch/guide/current/index.html on 15.09.2016.

Laptev, I., Marszałek, M., Schmid, C., & Rozenfeld, B. (2008). Learning realistic human actions from movies. *Conference on Computer Vision and Pattern Recognition.* Anchorage.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing* (Vol. 999). Cambridge: MIT Press.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & David, M. (2014). The Stanford Core NLP Natural Language Processing Toolkit. *the 52th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, (pp. 55-60).

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM, 38*(11), 39-41.

Rohrbach, A., Rohrbach, M., Tandon, N., & Schiele, B. (2015). A Dataset for Movie Description. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Tsoneva, T., Barbieri, M., & Weda, H. (2007). Automated summarization of narrative video on a semantic level. *Proceedings of the International Conference on Semantic Computing.*

Turetsky, R., & Dimitrova, N. (2004). Screenplay alignment for closed-system speaker identification and analysis of feature films. *Proc. ICME'04. 2004 IEEE,* 1659-1662.

Zou, W. Y., Zhu, S., Ng, A. Y., Kai, Y. (2012). Deep learning of invariant features via simulated fixations in video. *Advances in neural information processing systems*, (pp. 3212-3220).