University of Edinburgh

Text Technologies for Data Science

Film Genre Classification using Subtitles

Emanuilov, Stiliyan - s1430428
Ivanov, Simeon - s1346621
Maza Dominguez, Serafín - s1449315
Rust, Sam - s1315613
Stergiou, Konstantinos - s1412880

January 19, 2018

Abstract

We attempted to create a collection of freely available, English language subtitles corresponding to films and use them to train a classifier that predicts the genres of a film given the subtitles. The data was collected by scraping OpenSubtitles and IMDb, and the classification was done via an SVM classifier using a bag-of-words representation. Due to the limitations of the source website for the subtitles the collected data was a fraction of what was initially planned, which resulted in a suboptimal performance of the classifier.

1 Introduction

The "genre" of a film is often subjective and while certain films may be perfect representation of a given genre, it is rarely so. What is horror to one may be comedy to another which may cause a considerable amount of debate between both critics and fans. It is not a secret that, other than the visual content of a film, the dialogue provides the most contextual information. In addition, it is substantially easier to manipulate, thus it is a perfect choice for a machine learning algorithm that will attempt to answer the question 'Can we successfully classify a film's genre using subtitles?'.

The purpose of this document is to provide a detailed overview of our group project for Text Technologies for Data Science. We had two main objectives for this project. The first was to collect a set of English film subtitles with genre, while the second was to perform classification task onto film genres using the films' subtitles. A presentation of previous work and relevant literature on the topic is given in Section 2. The data collection, processing, and the classification task are explained in Sections 3, 4, and 5 respectively. And finally, a conclusion of our work is given in Section 6.

All the code we talk about throughout this report can be found on our GitHub repository [Stergiou u. a. (2017)].

2 Literature

Initially, we did some research into the feasibility of classifying the genre of a film based on it's subtitles. A great deal of work has gone into classifying the genre of songs or webpages based on their text content [Sadovsky und Chen (2006), Santini (2007)]. Despite this, there was little work to be found on classifying a movie's genre using only it's subtitles. Some of the existing work done by other university group projects had found some success, albeit on an extremely limited dataset. Given 399 scripts and 22 different genre labels (with each movie having multiple genres), a Stanford group achieved a 50% accuracy using a Maximum Entropy Markov Model [Blackstock und M.Spitz (2008)]. This group did choose to classify a successfully prediction of any of the genres of the film as a successful classification, so their statistics are inflated compared to other studies which do not make this assumption. Another similar project by a student group at Bilkent University managed to gather more data, with over 9000 subtitles split over just 6 Genres. For this project, the decision was made to only use a single genre for each film. Additionally, each of the 6 genre categories had roughly 1000 subtitles, allowing for larger training, test, and validation sets than the previous example. Their analysis of their collected data provides some interesting insights on differences inherent to differing movie genres. For example, comedy films averaged 69 words per minute whereas horror films only averaged 43. Running a kNN classifier over the generated words per minute and dialogue per minute feature sets, they managed to tune to an accuracy of 32% . Additionally, this group collected hearing impaired subtitles which encode additional, non-speech sounds. They again ran a kNN classifier on this data to use as an additional feature for classification. Their final classifier combined the above classifiers with BOW on the text from each film, and managed to achieve 78% accuracy at correctly classifying one of the genres [Yildiz u. a. (2016)].

3 Data Collection

We divided the data collection task into two main subtasks: collecting a list of movies with their title, genre and IMDb ID; and downloading the actual subtitles.

3.1 Title, Genre, and IMDb ID Collection

The collection consists of 4500 English subtitles for a variety of films released between 1972 and 2017. We scraped close to 20000 titles and their corresponding genres from IMDb (Internet Movie Database) [Needham (1996)]. This was done using Scrapy [Ltd. (2008)], an open source framework for extracting data from websites. The first half of the titles were acquired from the IMDb list called "All U.S. Released Movies: 1972-2016" and we collected the second half by scraping results for films in English from IMDb's advanced search. Four elements were essential to extracting the film data using the Scrapy framework. The two obvious items were the films' title and genre. While the title was not needed for the classification process, it helped with keeping the data human readable for debugging purposes. The third one was the pagination link that leads the film crawling spider to the next page of films. The last one was the films' id in IMDb which was used both to differentiate them and to search for subtitles. The Scrapy API allows us to access the aforementioned elements using either a CSS or an XPath expression. We opted to use CSS expressions as they are more intuitive when parsing HTML but they can easily be rewritten to XPath should the situation demand so. The expressions for the elements are not hard to derive with some HTML knowledge. However, it is important to note that different lists in IMDb may have vastly different designs and thus different HTML. We also noted that IMDb often update and revise their website which means that the scripts for collecting film titles may have to be updated accordingly if they were to be used in the future.

3.2 Subtitle Collection

The subtitles for the scraped films were downloaded from OpenSubtitles [OpenSubtitles.org (2008)]. We used a python implementation of the OpenSubtitles API called `python-opensubtitles` [@agonzalezro (2015)], which allowed us to request information about a film by using its IMDb ID. With the help of the urllib module in python we automatised the downloading task.

However, the website imposes limitations on the amount of subtitles a user can download. This is the main reason the collected subtitles are noticeably less than the collected film titles. In an attempt to work around this limitation we created several accounts, as well as purchasing one VIP account, and we looped through them to partially alleviate some of the download limitations but it was still not enough to get us the needed amount of subtitles.

4 Data Processing

Initially, the collection consisted of 6500 subtitles. However, we discovered that a lot of the subtitles we downloaded were mistakenly labelled as English. Because of that, a module was implemented to filter out foreign subtitles using LangDetect [@Mimino666 (2014)], we used 85% confidence level). Moreover, there were subtitles that had unreadable encodings and needed to be removed. All in all, we filtered out around 2000 subtitles, which left us with 4529. In order to do classification, the 4529 subtitles were randomly shuffled and divided into sets of 3529 for training, 500 for development and 500 for testing.

4.1 TREC Collection

As a side outcome of this project, we also decided to create a collection of documents with the XML format usually seen in the Text Retrieval Conference (TREC) [NIST (1992)]. With the help of the `readSubtitles` module we created, and a few lines of code in a script, we automated the generation of TREC format collections of subtitles contained in a folder. This collection uses the IMDb ID as the `<DOCNO>` tag, the `<TITLE>` tag for the title of the film,the `<GENRE>` tag for the genre(s) of the film, and finally, in the position of the `<TEXT>` tag, all the parsed phrases found in the subtitles file. A link to this collection can be found in the GitHub repository we used for this project [Stergiou u. a. (2017)], inside the `subtitles-module` folder.

5 Classification Task

In order to classify the subtitles of each movie, we used the bag of words approach. For this purpose, all the words from the training set were extracted into a feature file. Stop words were removed from the file and the rest was lowercased and stemmed using the Porter Stemmer [Porter (2017)]. That resulted in a total of 12706 feature word tokens that were used in a bag of words representation for each of the training, development and testing set.

SVM was chosen as a classifier, since it works well for text classification with limited amount of data. We used the scikit learn machine learning python library that has available implementations of different SVM classifiers. In previous works, the classification task focused on single-genre classification. However, realistically, movies can have multiple genres, so we need to be able to do multi-label classification. Scikit's OneVsRestClassifier allows us do multi-label classification by doing one-label classification (using a classifier passed as a parameter) for each class and then combining them do predict multiple classes. After testing several different classifiers, skicit's SVC single-label classifier was found to work best with the OneVsRestClassifier.

The base classifier uses a simple bag-of-words representation by counting only whether a feature word appears or not. We used the whole 3529 training examples that had in total 25 possible genre labels. Different metrics were used to measure the performance of our classifier on the development set. Since it is a multi-label classification, accuracy is a harsh metric, because for a prediction to be accurate all the labels need to be predicted correctly. Instead, we focus more on the average precision, recall, f1-score and hamming score. The model's parameters were tuned on the development set and the results on the testing set are shown in Table 1. We can see from the results that the base model has a poor performance in terms of recall, hamming score and f1-score. The only metric it performs slightly better in is precision, achieving 0.50.

These results can be explained by our limited data. Table 2 shows the distribution of the training set subtitles in terms of the 25 genres. As shown, some of the genres have a really low subtitle count in the training set and that classifier simply does not have enough data to make a better prediction. Thus, we decided to reduce the classes in half, focusing on the more popular movie genres. This left us with the following 12 labels: Action, Adventure, Biography, Comedy, Crime, Drama, Fantasy, Horror, Mystery, Romance, Sci-Fi, Thriller. Our training, development and testing data was filtered out to not include movies that do not have any of the 12 genres. This left us with 3504 training samples, 495 development samples and 498 testing examples. Re-running the feature extraction on the new training set gave us 12654 features which were used to generate an updated feature data for the classifier. The base model was then tuned on the 12 labels with the new data and the results are reported in Table 1 as well. The performance, however, only increased slightly, not giving a significant improvement over the 25-genre case.

Another model was created that used a bag-of-words approach with the counts of each word rather than just a value for appearance. Simply counting the words however has one issue. It favours longer subtitles and common words. Instead, the word counts were normalized by converting them to a TFIDF (term frequency–inverse document frequency) count. The model was tuned and run on the same 12-label data. The results shown in Table 1, however, show exactly the same results as the simple bag-of-words words which only counts whether a word appears or not.

Since trying to improve the classifier with further feature engineering did not give us any significant improvements, we concluded that the data is simply too limited to make a good base multi-label classifier.

6 Conclusion

Based on this research, we noticed that while classifying a single genre for a film has been successfully demonstrated, multi-label classification on film genres was not something which had previously been attempted.

Model	Accuracy	Precision	Recall	F1-score	Hamming Score
Base - 25 labels	6,60%	0.50	0.21	0.30	0.24
Base - 12 labels	9.69%	0.51	0.23	0.32	0.27
TFIDF - 12 labels	9.69%	0.51	0.23	0.32	0.27

Table 1: Testing set results.

Genre	Training set count	Genre	Training set count
Action	3840	Adult	1
Adventure	3018	Animation	754
Biography	1512	Comedy	7171
Crime	3410	Documentary	534
Drama	10167	Family	1300
Fantasy	1362	Film-Noir	48
History	706	Horror	2429
Music	643	Musical	274
Mystery	1553	News	5
Romance	3417	Sci-Fi	1392
Short	16	Sport	484
Thriller	3184	War	415
Western	251		

Table 2: Training subtitles genre counts.

On the Data collection front, OpenSubtitles has proved to be a not-so-reliable database for this kind of task, as explained in Section 3.2. We struggled to download all the subtitles due to the download limitations even with multiple accounts, one of them being VIP; and then in the processing stage we realised that around 2000 subtitles (See Section 4) were mistagged as English. On the other hand, OpenSubtitles is a community-based database, which relies on its users for the integrity of their data: therefore we acknowledge this is something we should have investigated better beforehand.

With the help of automatic language detection code and some manual work we filtered out all this noisy data and created a TREC-like collection (Section 4.1) and proceeded to the following stage: Classification.

Classifying the movie subtitles into multiple genres proved to be difficult. While, the chosen classifier and feature representation are usually effective for text classification, they proved to be ineffective for a multi-label classification with such limited amount of data. The OneVsRestClassifier with an SVC-SVM classifier can be a good starting point for movie genre prediction, however, since we have many labels, much more data has to be collected before attempting to build on it. For a 25-label classifier, we need at least 30 thousand subtitles with a good balance among the classes. Something that wasn't present in our data.

There are some open ends in this project that allow further working on this idea, for instance, finding new features that would improve our results, such as using captions for hearing-impaired people, or speaking frequencies. Moreover, we believe that a bigger dataset would have significantly improved our classification, but due to time and resource constraints we could not expand this dataset any further.

References

[@agonzalezro 2015] @agonzalezro: *python-opensubtitles, A Python Wrapper for the OpenSubtitles API.* 2015. – URL `https://github.com/agonzalezro/python-opensubtitles`

[Blackstock und M.Spitz 2008] Blackstock, A. ; M.Spitz: Classifying Movie Scripts by Genre with a MEMM using NLP-Based Features. (2008). – URL `https://nlp.stanford.edu/courses/cs224n/2008/reports/06.pdf`

[Ltd. 2008] Ltd., Scrapinghub: *Scrapy, A Fast and Powerful Scraping and Web Crawling Framework.* 2008. – URL `https://scrapy.org`

[@Mimino666 2014] @Mimino666: *langdetect, Port of Google's language-detection library to Python.* 2014. – URL `https://github.com/Mimino666/langdetect`

[Needham 1996] Needham, Col: *IMDb Homepage.* 1996. – URL `http://www.imdb.com/`

[NIST 1992] NIST: *Text REtrieval Conference (TREC), to Encourage Research in Information Retrieval from Large Text Collections.* 1992. – URL `http://trec.nist.gov/`

[OpenSubtitles.org 2008] OpenSubtitles.org: *OpenSubtitles XMLRPC API.* 2008. – URL `http://trac.opensubtitles.org/projects/opensubtitles/wiki/XMLRPC`

[Porter 2017] Porter, Martin: *Porter Stemmer for Perl.* 2017. – URL `http://www.inf.ed.ac.uk/teaching/courses/tts/labs/2017/Porter.pm`

[Sadovsky und Chen 2006] Sadovsky, A. ; Chen, X.: Song genre and artist classification via supervised learning from lyrics. In: *Previous CS22N Final Project* (2006)

[Santini 2007] Santini, M.: Automatic Identification of Genre in Web Pages. In: *University of Brighton* (2007)

[Stergiou u. a. 2017] Stergiou, K. ; Rust, S. ; Domínguez, S. M. ; Ivanov, S. ; Emanuilov, S.: *Repository containing code for this project.* 2017. – URL `https://github.com/xsrust/ttds_group`

[Yildiz u. a. 2016] Yildiz, M. ; Akdogan, E. ; Akgul, O. ; Erdogan, A. ; Gurlek, M.: Movie Genre Detection based on Subtitle Analysis Methods. (2016). – URL `\url{https://github.com/mesutgurlek/Movie-Category-Classification-from-Subtitles/blob/master/Group4\_Final\_Report.pdf}`