# Classification of Movie Genres Through Movie Subtitles

Ali Burak ERDOGAN
Comp. Eng. Dept.
Hacettepe University
Ankara, Turkey
erdoganaliburak@gmail.com

***Abstract*— This study aims to perform classification of movies into movie genres, utilizing the subtitles. The first phase of study reached 63% rate of accuracy at classification of subtitles into 8 movie genres. Our experiments show that Logistic Regression algoirthm scores the best with TF-IDF unigram text models.**

***Keywords—text classification, subtitles, movie classification***

## I. INTRODUCTION

The task of text classification is highly in demand in our era due to the vast number of textual data present in the digital world. Many of the examples of this task focus on binary classification such as spam filtering, or sentiment analysis. Our project focuses on a supervised learning task for classification of movies making use of the subtitles. We have collected more than 9,000 movie subtitles belonging to 8 different genres successfully and performed various state-of-the-art machine learning algorithms.

## II. RELATED WORK

Text classification has been extensively studied and researched over many years. This task requires some basic steps:

- Data collection: retrieving textual data either manually or via automated programs from Web or other media.
- Text Preprocessing: Tokenizing text, stemming words, removal of stopwords and any other useless texts
- Feature Extraction and Selection: Different approaches such as n-grams or unigrams, TF-IDF or Bag-of-Words etc.
- Model Training: Applying machine learning algorithms to build a model

For the feature selection, numerous techniques have been applied in order to have dimensionality reduction due to the high amount of vocabulary in texts. Studies show that TF-IDF with thresholding can be used for computation efficiency and successful representation of data. [1]

Model training step contains numerous alternative algorithm options. Each can be applied and then be compared in terms of efficiency. For example, SVM provides a binary classification capability, however SVM can also be utilized for multiclass classification by using one-vs-rest approach. Naive Bayes algorithm follows a probabilistic approach based on the Bayes theorem for computation of probability of an item belonging to each class.[2]

Since this project's aim focuses on categorizing movies, we can list some studies made for classification of videos benefiting from subtitles. In a study named VIRUS [3], simultaneous analysis of video, audio and subtitle content is performed for video classification. In another study, an unsupervised approach is used for video classification using WordNet [6] lexical database.

## III. METHODS

### *A. Data Collection*

More than 9,000 subtitles from 8 movie genres as equally distributed in number (Action, Comedy, Crime, Horror, Musical, Romance, War, Western) have been collected by using OpenSubtitles.org API [4] and ScraPy [5] library for automated download of a big amount of subtitles. The distribution of subtitles is shown in figure below:

```
pprint.pprint(collections.Counter(fullgenre))

Counter({'Horror': 986,
         'Action': 986,
         'War': 986,
         'Comedy': 986,
         'Romance': 986,
         'Crime': 986,
         'Musical': 986,
         'Western': 985})
```

Figure 1. Equally distributed sets of movie subtitles from various genres

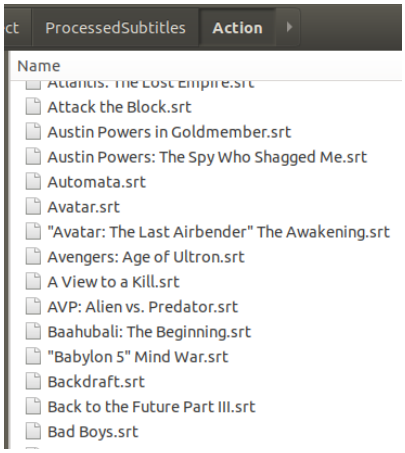


Figure 2. Example view of subtitles data from Action genre

*B. Text Preprocessing*

Preprocessing subtitle texts is not different than other types of texts except for filtering some subtitle format (.srt) related tags and marks. As can be seen in Figure 3, other than the dialogues, each row includes dialog id, time identifier and some HTML tags for markdown text. Also, there are sometimes sound descriptions enclosed in parentheses for hearing impaired people. During our text preprocessing step, we filter all those out, including time information since we disregard the order of dialogues as well. Next, we apply Porter's stemming algorithm to individual words since words of "loving" and "loves" do not make any difference for our purpose. Next, we filter some common stop words such as "the", "him", "in", "but" by making use of NLTK's stopword list for English. The final output of a preprocessing step can be seen in Figure 4.

```
1
00:00:41,333 --> 00:00:43,586
<i>(rowdy voices and blows landed)</i>

2
00:00:44,002 --> 00:00:46,130
MAN: Come on, put some effort in!

3
00:00:46,838 --> 00:00:48,181
Come on, son!

4
00:00:48,257 --> 00:00:50,305
Right in the kisser!

5
00:00:50,467 --> 00:00:52,515
Keep moving.
```

Figure 3. Sample subtitle before text preprocessing

```
man
come
put
effort
come
son
right
kisser
keep
move
that
son
```

Figure 4. Sample subtitle after text preprocessing

*C. Feature Extraction*

*1) Bag of Words*

Bag-of-words is one of the simplest methods for representing textual data. This approach holds a dictionary of words present in all documents, and stores the occurrences of terms in a given document.

*2) TF-IDF*

Tf-Idf is another approach for computation of text features in documents. [7] Basically, it's the product of two metrics: term-frequency (tf) and inverse-term-frequency (idf). Tf score is defined as the frequency of a term t in a document d. Idf score of term t is computed as the rareness of it among all of documents D.

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D)$$

The product of those two metrics yields the tf-idf score, and it's a very strong metric for emphasizing important terms with high scores, and decreasing the importance of common words appearing in all classes of texts, for example common verbs like "go", "take".

*3) n-gram Model*

Two approaches we discussed above do not take word orders into account by default. However, it is possible to try to increase the effectiveness of our models by using n-gram models. For example, instead of computing separate word counts, we could parse the text groups of consecutive two-words, described as a bigram model (2-gram). An example set of bigrams from our preprocessed subtitle data can be seen in Figure 5. As for our dataset consisting of 7887 subtitles, the shape of the matrix for our model is (7887 x 626244) for bigram, and (7887 x 44339) for 1-gram model.

```
vectorizer.vocabulary_.keys()
```

```
'love richard', 'true test', 'like samurai', 'realiz potenti', 'sir t
raffic', 'make flight', 'im broadway', 'andand im', 'make curtain',
'contract need', 'know liter', 'want album', 'progress think', 'la ok
ay', 'okay delay', 'tonight perform', 'mean rachel', 'hold um', 'poss
ibl um', 'spend coupl', 'sidney ive', 'email say', 'tomorrow screw',
'return kind', 'okay sudden', 'heart realiz', 'thing freez', 'use bit
ch', 'screw mean', 'person held', 'huge talent', 'russel crow', 'john
ni carson', 'matter aw', 'santana oh', 'gonna final', 'final alon',
'time couch', 'say june', 'june know', 'know introduc', 'friend ric
h', 'famou way', 'text like', 'hate sweetheart', 'mean ruin', 'green
realli', 'uncertain term', 'finish lie', 'betray im', 'child star',
'star need', 'know mistak', 'broadway legend', 'exist hell', 'great a
udit', 'mean clearli', 'project said', 'want develop', 'oh figur', 'gc
ter', 'celebr friend', 'penlight', 'doj', 'saling', 'epicent', 'unsub
stanti', 'tox', 'criteria', 'alberto', 'ahmet', 'oneminut', 'capitan
o', 'sherwood', 'alitalia', 'real blow', 'open told', 'told bank', 'b
uy 200', 'meet agre', 'ear ye', 'like overnight', 'berlin polic', 'co
me ella', 'odd guy', 'guy foreign', 'tommi close', 'want wit', 'strai
```

Figure 5. 2-gram model of a sample subtitle after vectorization

## IV. CLASSIFICATION EXPERIMENTS

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

### *A. Selecting the feature model*

We used 4 different feature sets and 4 different learning algorithms during our experiments. In order to compare efficiency of n-gram models and make a final decision between Bag-of-Words and TF-IDF approach, we run all learning algorithms with those 4 different feature models. As we can see in Figure 6, **TF-IDF with 1-gram model is the most successful method** in terms of high accuracy. The detailed outputs of those models are listed extensively in Appendix. Another conclusion is that TF-IDF is a more successful method when compared to Bag-of-Words in the majority of cases. This is most probably because TF-IDF takes the inverse-document-frequency into account, which means important words are given more highlighted and resulting in a better classification performance. Another interesting result is that 2-gram models did not increase the accuracy of classification in a significant matter.

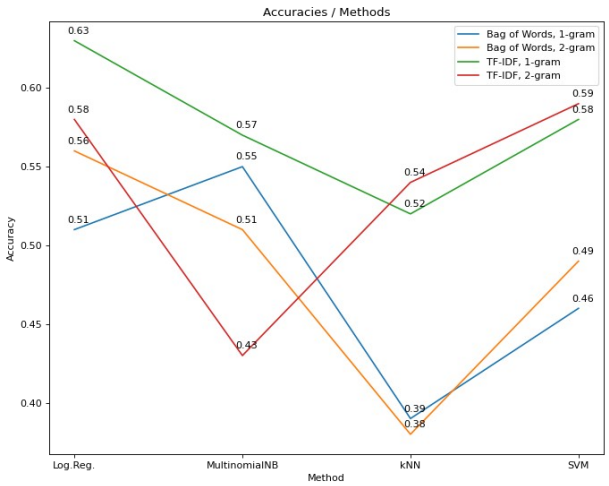


Figure 6. Comparison between feature sets and learning algorithms in terms of accuracy

### *B. Effectiveness of learning algorithms*

Figure 6 shows that Logistic Regression mostly yielded the most accurate classification performance. Since we have 8 classes, the random classification score is 12.5%. Logistic regression with TF-IDF 1-gram model gave 63% accuracy. The second best algorithm is SVM, specifically Linear Support Vector Classifier, with an accuracy score of 58%.

### *C. Classification performance of movie genres*

Another aspect of our experimental results is the unequal distribution of performance of different genres. As can be seen in Figure 7, F1-scores which is the harmonic mean of precision and recall scores, are quite high in some genres such as Horror, Musical, War and Western and quite low in some other genres such as Romance, Action, Comedy, Crime.

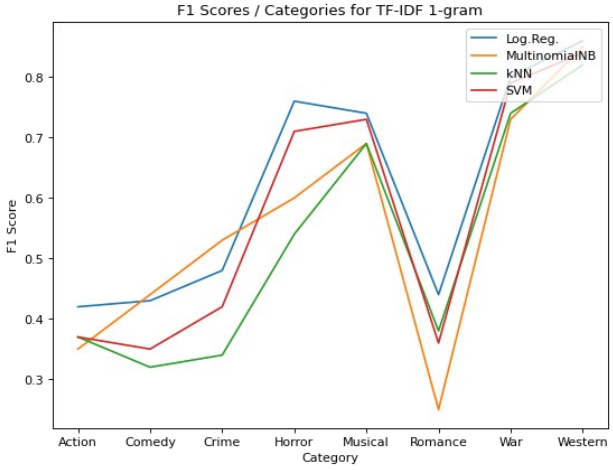


Figure 7. Classification F1-scores of each movie genre

## V. Evaluation

To further investigate this situation, we can check the confusion matrix of Logistic Regression with the TF-IDF 1-gram model in Figure 8 since we selected it as the best performing model. Confusion matrices of other models are available in the Appendix.

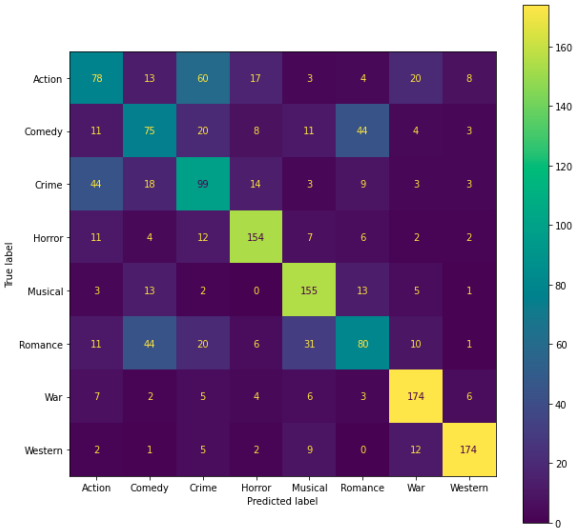


Figure 8. Confusion matrix of Logistic Regression model with TF-IDF 1-gram feature set

Confusion matrix allows us to understand which classes are confused during classification. As we can see Action and Crime genres are the most confused classes. It can be expected due to the common subset of actional words in these two genres. The second most confused classes are Comedy and Romance. We can explain this with the presence of emotional words in those two classes. We further analyzed the presence of common important words among movie genres and generated a heatmap. First, we collected the top-10 words from each movie according to its TF-IDF score. Then, we collected those words into separate lists belonging to specific genres. Then, we compared each genre to another in terms of common words. Figure 9 shows that intersection sets of words are at the highest level in Action-Crime, and Romance-Comedy pairs. Also, we see that the smallest set of common word counts belong to genres Horror, Musical, Western and War. This heatmap has a complete correlation with the F1-score table presented in Figure 7. We conclude that, as the number of common set of words increase in two movie genres, the ability to differentiate them during classification becomes more difficult, hence the F1-score decreases.

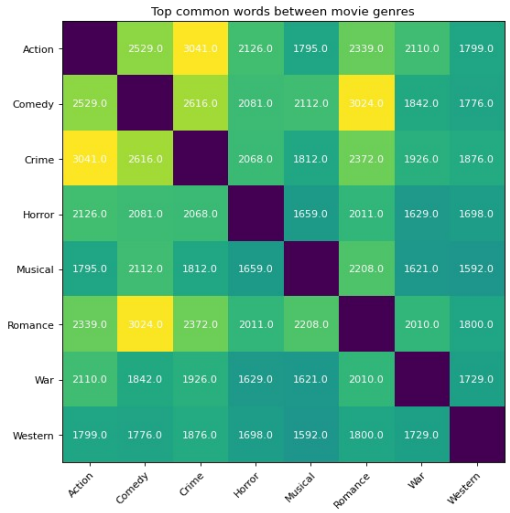


Figure 9. Intersection of most important word lists between movie genres

## VI. Conclusion

We performed the essential parts of text classification on our dataset, and reached 63% of accuracy as the best for classification into 8 categories. We also saw that using n-grams does not increase the overall success rate in a significant manner. In order to make progress on our project, we plan to apply dimensionality reduction techniques such as PCA, and compare the results.

## References


[1] Y. Yang, and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In Proceedings of the Fourteenth International Conference on Machine Learning.. D. H. Fisher, Ed. Morgan Kaufmann Publishers, San Francisco, CA (1997) 412- 420

[2] Katsiouli P., Tsetsos V., Hadjiefthymiades S. 2007. Semantic video classification based on subtitles and domain terminologies Workshop on Knowledge Acquisition from Multimedia Content.

[3] Langlois, Thibault & Chambel, Teresa & Oliveira, Eva & Carvalho, Paula & Marques, Gonçalo & Falcão, André. (2010). VIRUS: Video information retrieval using subtitles. Proceedings of the 14th International Academic MindTrek Conference: Envisioning Future Media Environments, MindTrek 2010. 197-200. 10.1145/1930488.1930530.

[4] OpenSubtitles API. https://opensubtitles.stoplight.io/docs/opensubtitles-api/open_api.json

[5] ScraPy. https://scrapy.org/

[6] Fellbaum, Christiane (2005). WordNet and wordnets. In: Brown, Keith et al. (eds.), Encyclopedia of Language and Linguistics, Second Edition, Oxford: Elsevier, 665-670.

[7] SPARCK JONES, K. (1972), "A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL", Journal of Documentation, Vol. 28 No. 1, pp. 11-21. https://doi.org/10.1108/eb026526