ALLIANCE UNIVERSITY

*Private University established in Karnataka State by Act No.34 of year 2010*
*Recognized by the University Grants Commission (UGC), New Delhi*


# Alliance College Of Engineering and Design

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING- COMPUTER SCIENCE AND ENGINEERING**

**Batch: 2019-2023**

# Movie Genre Classification Using Plot Summary

**By:**

**Name : Payal**

**Registration NO : 19030141CSE062**

**Semester : VI**

# CS-603 MACHINE LEARNING PROJECT

ALLIANCE UNIVERSITY

*Private University established in Karnataka State by Act No.34 of year 2010*
*Recognized by the University Grants Commission (UGC), New Delhi*

*Celebrating* 25 SILVER JUBILEE *years of Alliance Education*

# CERTIFICATE

This is to certify that, **Payal**, **Registration No.: 1903141CSE062** of Batch 2019-2023 has completed the report titled "Movie Genre Classification Using Plot Summary", for the partial fulfilment of the Course: Machine Learning in Semester VI of the Bachelor of Technology in Computer Science.

**INTERNAL GUIDE:**
**Dr. Shekhar R.**

**HEAD OF THE DEPARTMENT:**
**Dr. Abraham George**

ALLIANCE UNIVERSITY

*Private University established in Karnataka State by Act No.34 of year 2010*
*Recognized by the University Grants Commission (UGC), New Delhi*

*Celebrating 25 SILVER JUBILEE years of Alliance Education*

# DECLARATION

This is to declare that the report titled "Movie Genre Classification Using Plot Summary" has been made for the partial fulfilment of the Course Bachelor of Technology in Computer Science and Engineering VI semester, under the guidance of **Dr. Shekhar R.**

I confirm that this report truly represents my work undertaken as a part of my project work. This work is not a replication of work done previously by any other person. I also confirm that the contents of the report and the views contained therein have been discussed and deliberated with the faculty guide.

| NAME | REG NO | SIGNATURE |
|---|---|---|
| PAYAL | 19030141CSE062 |  |

# ABSTRACT

This project explores several Machine Learning methods to predict movie genres based on plot summaries. Naive Bayes, Word2Vec+XGBoost, and Recurrent Neural Networks are used for text classification, while K-binary transformation, rank method, and probabilistic classification with learned probability threshold are employed for the multi-label problem involved in the genre tagging task. Experiments with more than 250,000 movies show that employing the Gated Recurrent Units (GRU) neural networks for the probabilistic classification with a learned probability threshold approach achieves the best result on the test set. The model attains a Jaccard Index of 50.0%, an F-score of 0.56, and a hit rate of 80.5%.

# TABLE OF CONTENTS

# CHAPTER-1

# INTRODUCTION

Movie plot summaries reflect the genre of the movies such as action, drama, horror, etc., such that people can easily capture the genre information of the movies from their plot summaries. Especially, several sentences in the plot summaries are high representative of the genre of the movie. People usually read the plot summaries of movies before watching them get an idea about the movie. Therefore, plot summaries are written in such a way that they convey the genre information to the people. For example, if the plot mentions humorous obstacles that must be overcome before lovers eventually come together, the movie is likely to be a romantic comedy. In this regard, there is a hidden representation of genre information in the movie plot summaries. In this study, we aim to learn this hidden representation. In other words, our purpose is to classify the genres of the movies from their plot summaries using Bi-LSTM by considering genre information represented by each individual sentence. With this method, representations of plot summaries can be used for movie recommendations. In addition to that, it can be inferred whether a plot summary actually reflects the genre of the movie it belongs to. Therefore, this method can be beneficial during the preparation of movie plots. In the literature, there exists a number of studies that perform movie genre classification using a variety of sources including visual, audio, and textual features from trailers, posters, and texts. Among the studies that employ visual and/or audio features, visual features are utilized including average shot length, colour variance, motion content, and lighting key, from movie previews to predict movie genres. visual features from videos are employed including temporal and spatial ones to classify genres using hierarchical SVM. Movie trailers are implemented using a bag-of-visual-words model with shot classes as vocabularies and utilized for genre classification. Moreover, both visual and audio features are extracted from movie trailers using a meta-heuristic optimization algorithm and performed genre classification. low-level audio and visual features are combined including signal energy, the fundamental frequency for audio; colour, and texture-based features for a visual representation to conduct multimodal genre classification. Low-level visual features are employed based on colours and edges obtained from movie posters, then used to classify posters into genres. Furthermore, convolutional neural networks (CNNs) are used based on architectures to perform movie genre classification from movie trailers instead of using hand-crafted features. In addition to efforts employing visual and audio features, several studies used textual sources including plots and synopses for movie genre classification. A vector space model is utilized to represent synopses and used this representation as input for SVM.

textual features from social tags are extracted via social websites. Then, they applied probabilistic latent semantic analysis (PLSA) to incorporate textual, visual, and audio features for genre classification. Furthermore, Arevalo et al. [13] proposed a gated unit for multi-modal classification task and they performed movie genre classification using poster and plot information with a basic recurrent neural network (RNNs) to represent plot information. Similarly, a column network is proposed for collective classification and they evaluated this network on movie genre classification tasks using plot summaries. They represented plot summaries as Bag-of-Words (BoW) vector of 1.000 most frequent words. The aforementioned studies using TABLE I DISTRIBUTION OF THE SAMPLES FOR EACH GENRE Thriller Horror Comedy Drama # of Full Plots 1590 1590 1590 1590 # of Sentences 5421 5437 5480 5940 plots or synopses either did not benefit from the power of deep learning for sequence modeling of textual data or they obtained textual representations using basic RNNs. Moreover, these studies performed document-level use of text for genre classification.

Supervised text classification is a mature tool that has achieved great success in a wide range of applications such as sentiment analysis and topic classification. When applying to movies, most of the previous work has been focused on predicting movie reviews or revenue, and little research was done to predict movie genres. Movie genres are still tagged through a manual process in which users send their suggestions to the email address of The Internet Movie Database (IMDB). As a plot summary conveys much information about a movie, I explore in this project different machine learning methods to classify movie genres using synopsis. I first perform the experiment with Naive Bayes using bag-of-word features. Next, I make use of the pre-trained word2vec embeddings to turn plot summaries into vectors, which are then used as inputs for an XGBoost classifier. Finally, I train a Gated Recurrent Unit (GRU) neural network for the genre tagging task.

# WORK DONE

A) Data Collection and Pre-processing

In a pre-processing step, we first obtained movie names from MovieLens1 datasets. We further collected necessary information of the movies including full plot summaries (input) and genres (ground-truth) through OMDb API2 using corresponding movie names as inputs. Within the scope of this study, we selected four types of genres, namely Thriller, Horror, Comedy and Drama for movie genre classification. Since the number of the movies in the database vary for each genre, we randomly sampled movies for each of them uniformly. However, the total number of sentences in the plot summaries changes for each genre as the plots may include different number of sentences. Accordingly, in the document-level classification task (using whole plots as inputs), we have uniformly sampled the data based on their genres. On the other hand, the data for the sentence-level classification (using sentences as inputs), is unbalanced for the training. As a result, we obtained a total of 6.360 movies and 22.278 sentences for the genre classification task, respectively. The Table I shows the distribution of the number of the movies and the total number of sentences for each genre in the dataset. Before training a model for classification, we conducted a pre-processing step. We first converted all texts in the plots to lowercase. Next, we eliminated all punctuation marks except the ones that separate the sentences. Additionally, we eliminated the stop-words. We also divided plot summaries into sentences for the sentence-level classification task. We performed all tasks in pre-processing step using NLTK3 .

B) Text Representation

The purpose of this step is to represent semantic and syntactic relationship among the words, which improves the performance where the training data is limited. After preprocessing step, each input (full plot for document-level and sentence for sentence-level) is represented using continuous vector representation. In order to do that, the pre-trained word vectors that are proposed by [15] are used. The word vectors are obtained as a result of training on Wikipedia. These vectors are in dimension of 300 and they were obtained using the skip-gram model. Therefore, the relationships between words and their

context are modeled beforehand. As a result, the row input is converted to continuous vector representation and then fed into the network. Note that, for any word in the plots that does not have a corresponding word vector in the dictionary, a random word vector in dimension of 300 was generated.

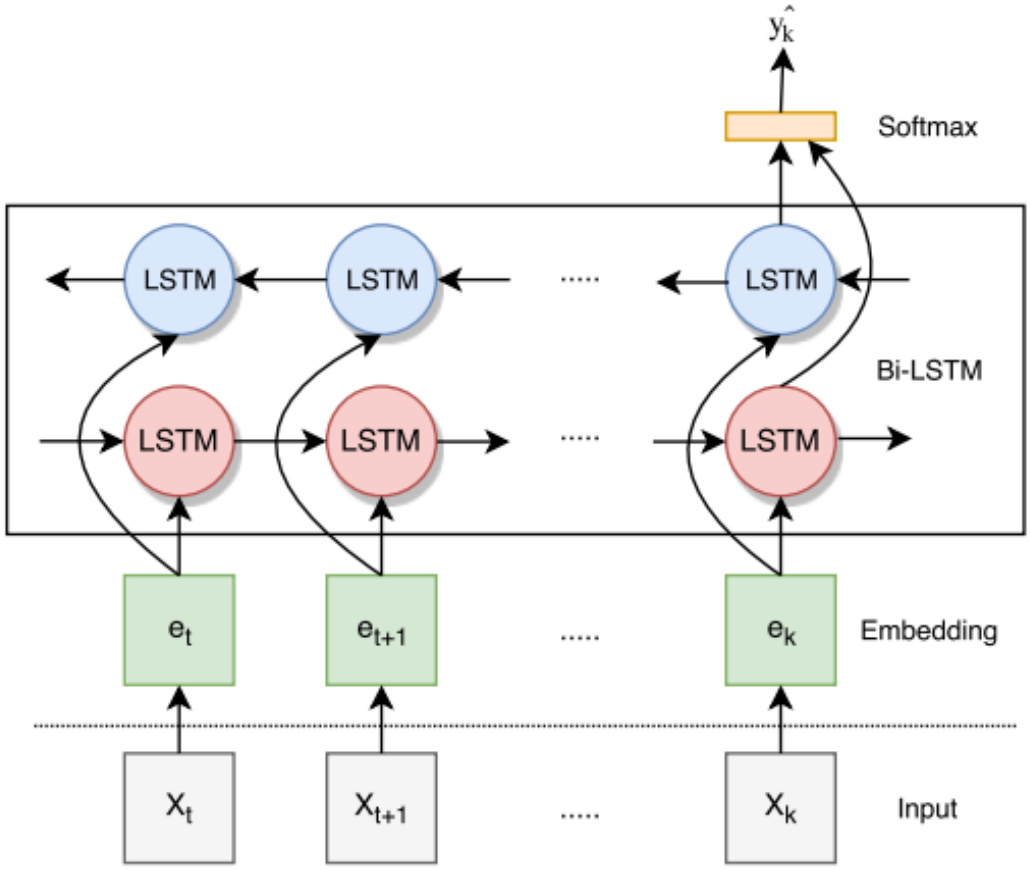



Fig. 1. Bi-LSTM Network Architecture for Movie Genre Classification

C) Model

The LSTM model [16] is an RNN architecture, which is capable of learning complex dependencies across time. LSTM RNNs address the vanishing gradient problem of basic RNNs by employing gating functions together with the state dynamics. In this study, we use Bi-LSTM network. It is composed of two LSTM neural networks, a forward LSTM to model the preceding contexts, and a backward LSTM to model the following contexts respectively. The architecture used in the study is given in Fig. 1. Note that, each plot summary of a movie is divided into sentences and the genre of corresponding movie is assigned to each sentence. During training, each input (sentence) is represented as the words it includes and continuous word representations are obtained using [15]. It is useful

when the limited data is used since semantic and syntactic relationship among the words are captured. We name this representation in the architecture in Fig. 1 as embedding layer. Then, the word representations are fed into the Bi-LSTM network. Practically, a linear projection layer is put between Bi-LSTM and softmax layers. Finally, a softmax layer, which is stacked on the top of the Bi-LSTM, takes the learned representations of the last output of Bi-LSTM as the input and returns the classification probabilities for each movie genre. In order to train the model, we minimize the negative log-likelihood of the estimation error, where the loss function is given in Eq. 1 below.

$$L(\theta) = -\frac{1}{C}\sum_{i=1}^{C} y_i \log(\hat{y}_i), \qquad (1)$$

where C is the number of the target classes, y is the onehot representation of the ground truth, and yˆ is the estimated probability distribution assigned to the genres by the model.

D) Classification

Since we divide the plot summaries into sentences, we estimate the class labels for each of them separately during test time. However, we need to assign a single class label for any given plot summary. Therefore, we fuse the decisions of the model for each sentence to obtain a final class label for the corresponding plot summary. For that purpose, we use majority voting to obtain the final decision by considering the posterior probabilities of genres assigned to the sentences. If the majority voting outputs a single label, then the plot summary is assigned to that label. On the other hand, if more than one genre have the maximum number of votes, we assign the genre to the plot summary, whose average class posterior probability is the maximum among others. The steps of the genre label assignment process are given in Algorithm 1.

**Algorithm 1** Assignment of genre label to a plot summary during prediction.

**Require:** $G_p \in \mathbb{R}^{n \times c}$, genre posterior probability vectors of all sentences of plot summary $p$, where $n$ is the number of sentences of $p$ and $c$ is the number of target classes.

**Require:** $G_{p,s} \in \mathbb{R}^{c}$, genre posterior probability vector of sentence $s$ of plot summary $p$.

**Ensure:** $\hat{y}_p$, estimated genre label for plot summary $p$.

$G_p^{avg} \leftarrow$ getAverageProbabilities($G_p$)
label_count $\leftarrow$ zeros($c$)
**for** $s \in p$ **do**
    estimated $\leftarrow$ index($\max(G_{p,s})$)
    label_count(estimated) $\leftarrow$ label_count(estimated) $+ 1$
**end for**
candidate_labels $\leftarrow$ indices($\max$(label_count))
**if** candidate_labels.length() $> 1$ **then**
    $\hat{y}_p \leftarrow$ findLabelOfHighestProb($G_p^{avg}$, candidate_labels)
**else**
    $\hat{y}_p \leftarrow$ candidate_labels.first()
**end if**

E) Evaluation Measures

Since the movie genre classification task is a multi-class classification problem, we used two versions for the averages of the f-score (f1), which are micro and macro. The former computes the f-score using all estimations at once. On the other hand, the latter computes the f-score for each genre. Since the movie genre classification task is a multi-class classification problem, we used two versions for the averages of the f-score (f1), which are micro and macro. The former computes the f-score using all estimations at once. On the other hand, the latter computes the f-score for each genre.

$$p^{micro} = \frac{\sum_{i=1}^{C} tp_i}{\sum_{i=1}^{C} tp_i + \sum_{i=1}^{C} fp_i} \quad (2a)$$

$$r^{micro} = \frac{\sum_{i=1}^{C} tp_i}{\sum_{i=1}^{C} tp_i + \sum_{i=1}^{C} fn_i} \quad (2b)$$

$$p^{macro} = \frac{1}{C} \sum_{i=1}^{C} \frac{tp_i}{tp_i + fp_i} \quad (3a)$$

$$r^{macro} = \frac{1}{C} \sum_{i=1}^{C} \frac{tp_i}{tp_i + fn_i} \quad (3b)$$

where C is the number of target labels, p is the precision, r is the recall, tpi , f pi and fni stand for the number of true positives, false positives and false negatives for the i th target label, respectively. For both micro and macro measures, we compute the f-score as f1 = 2×p×r p+r . Note that, since we perform our experiments on a single dataset, micro-precision, micro recall and micro f-score values are all equal and they represent the accuracy of the classifier. Accordingly, we only present the micro f-score for the micro results.

# CHAPTER-2

# ANALYSIS

## Hyperparameter selection

There is no hyperparameter to tune for the Naive Bayes models. Due to limited time for experiment and priority, the default setting for XGBoost is used. For the GRU network, the hyperparameters include the number of layers, the number of units in each layer, the recurrent dropout keep rate [23] and the learning rate for Adam optimizer [18]. Different values for each hyperparameter are considered as shown in Table 1 To tune the hyperparameters, I divide the dataset into the train, validation and test sets according to the 70/10/20 proportion. The hyperparameters that yield the highest SoftMax loss or rank loss in the validation set are selected. The optimal settings are 2 for the number of layers, 128 for the number of units in each layers, 0.8 for the dropout keep rate and 0.01 for the learning rate.

| Hyperparameters | Values |
|---|---|
| Number of layers | [1, 2, 3] |
| Number of units | [128, 256] |
| Dropout keep rate | [0.5, 0.6, 0.7, 0.8, 0.9] |
| Learning rate | [0.0002, 0.0005, 0.001, 0.003, 0.005, 0.008, 0.01, 0.02, 0.05] |

Table 1: Different values considered for hyperparameter selection.

## Evaluation metrics

A weak metric is the “hit rate”, proportion of examples where the model predicts at least one correct genre. However, tagging all movies with all genres can yield a hit rate of 100%. To punish for incorrect predictions, I adopt the Jaccard index, which is defined as the number of correctly predicted labels divided by the union of predicted and true labels, $|T \cap P|$ $|T \cup P|$ . To further compare performance of different methods, I calculate the confusion matrix,

accuracy, precision, recall, and F-score for each genre as well as for all of the test data. These metrics are defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

$$Fscore = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{14}$$

where TP, FP, TN and FN are true positive, false positive, true negative and false negative respectively.

Baseline

I use two approaches to estimate the baseline. The first approach is a simple heuristic that predict the top 1 (drama), top 2 (drama and comedy) or top 3 (drama, comedy and action) most popular genres for all movies, resulting Jaccard indices of 29.2%, 28.6% and 22.8% and hit rates of 45.3%, 66.4% and 71.3% respectively. So, tagging all movies to drama and comedy is a reasonable baseline. The second approach is to build 20 binary Naive Bayes model for each genre, using casefolding and the simple whitespace tokenizer. This method achieves an Jaccard index of 36.3%, which is a significant improvement over the heuristic baseline. The hit rate, however, is only 55.5%. Additionally, the F-score in this approach is 0.44.

Experiment results

Table 2 summarizes experiment results of different methods discussed in Section 4. Multinomial GRU achieves the highest Jaccard index of 50.0% and F-score of 0.56. Binary-based models such as Binary GRU and Binary Naive Bayes attain high precision (67.4% and 60.7% respectively), while rank-based models such as Multinomial Naive Bayes, rank GRU and Multinomial GRU

obtain high recall (56.3%, 56.1%, and 51.3% respectively). This result is plausible as binary-based models make decision for each genre independently, and return positive prediction only when it is very confident. Meanwhile, rank-based models make decisions based on relative rank values (or conditional probabilities) among different genres, thus output positive prediction more often. This observation is confirmed by Figure 2.

| Methods | Hit rate | Jaccard | F-score | Precision | Recall |
|---|---|---|---|---|---|
| Binary Naive Bayes | 74.2% | 42.6% | 0.52 | 60.7% | 45.0% |
| Multinomial Naive Bayes | **82.9%** | 46.6% | 0.53 | 50.7% | **56.3%** |
| XGBoost | 74.6% | 42.9% | 0.49 | 55.3% | 44.4% |
| Binary GRU | 70.6% | 46.5% | 0.53 | **67.4%** | 44.0% |
| Rank GRU | 82.7% | 48.5% | 0.56 | 56.6% | 56.1% |
| Multinomial GRU | 80.5% | **50.0%** | **0.56** | 61.0% | 51.3% |

Table 2: Summary of experiment results for different methods.

which shows that Multinomial Naive Bayes (in orange) and Rank GRU (in light blue) predict on average more genres per movie. As a result, these models get the highest hit rates, while Binary GRU and Binary Naive Bayes get the lowest. Overall, Multinomial GRU and Rank GRU achieve better balance among different metrics. Figure 1 shows F-score by genre of each model. It can be seen that GRU models (the last 3 columns in gold, light blue, and green) consistently attain higher F-score than Naive Bayes models (the first 2 columns in blue and orange) and XGBoost (the middle column in gray). XGBoost consistently under-performs across genres. Tuning hyperparameters might improve this model a little bit, but this result strongly suggests that taking average of embeddings vector of all words in a plot summary might be too lossy and does not yield a good representation. The Naive Bayes models are competitive with GRU models on the most popular genres such as Drama (46% of train examples), Comedy (27.9%), Thriller (11.2%), Romance (11.0%) and Action (9.7%), but under-performs on less popular genres. So, the GRU models generalize to less popular genres better. One reason for this behavior is that

Naive Bayes takes the bag-of-word and conditional independence assumption, and the data is not highly skewed. As a result, Naive Bayes models are biased towards the most popular genres. Figure 2 shows the size distribution of the predicted genre sets by different methods. One problem of the binary-based models is that many movies are tagged with no genre. Binary Naive Bayes (in blue) and Binary GRU (in gold) respectively predict empty set for 15% and 9% of test examples. The size distributions of the genre sets predicted by the rank-based models are closer to the true distribution.

# CHAPTER-3

# DESIGN AND IMPLEMENTATION

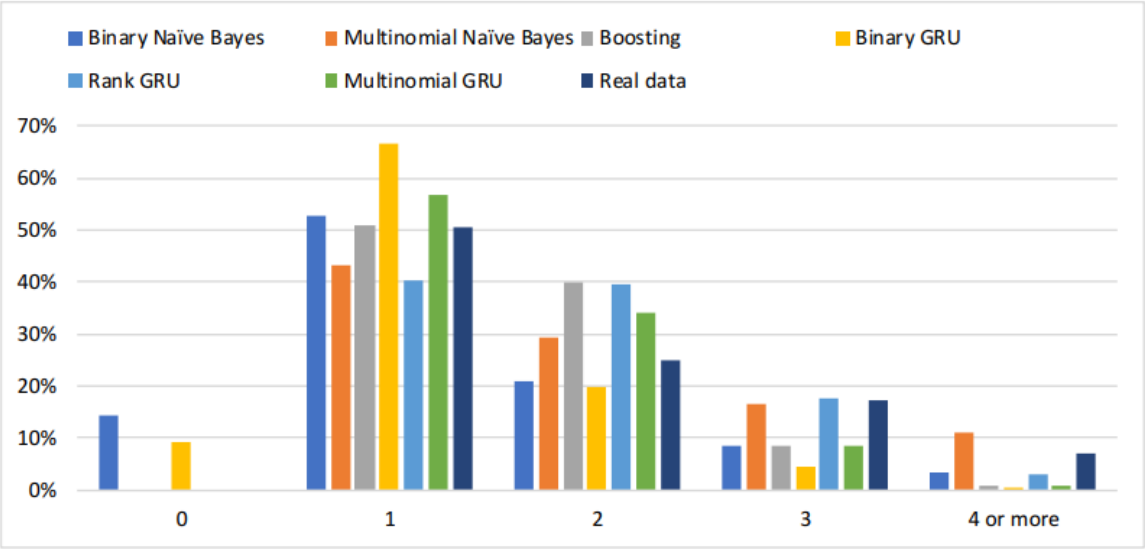



Figure 2: Size distribution of the predicted genre sets.

| | Comedy | Drama | Sport | War | Western |
|---|---|---|---|---|---|
| Precision | 54.1% | 60.2% | 80.0% | 65.3% | 91.1% |
| Recall | 75.3% | 85.1% | 35.5% | 27.1% | 47.5% |
| F-score | 0.63 | 0.70 | 0.49 | 0.38 | 0.62 |

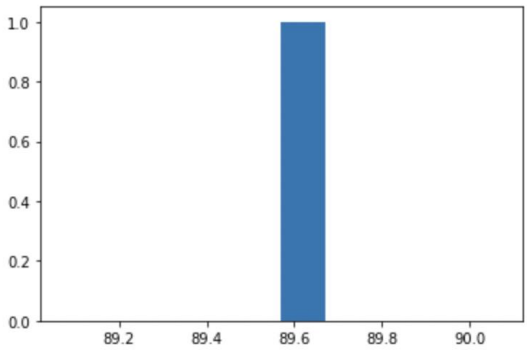
Table 3: The Multinomial Naive Bayes model's precision, recall and F-score for some genres.

**Naive Bayes Models**

As discussed, I consider two approaches to apply the Naive Bayes to the movie genre tagging task. The first approach is to build 20 binary Naive Bayes models for each genre. This model, when preprocessed with the whitespace tokenizer and casefolding, is used as the baseline. To better process the data, I apply casefolding, remove all stop words, and use the NLTK word tokenizer. As a result, this approach achieves a Jaccard index of 42.6%, which is a significant improvement over the whitespace-tokenizer version's 36.3% score. Similarly, F-score improves from 0.44 to 0.52. Using the same word normalization procedure, Multinomial Naive Bayes attains a Jaccard index of 46.6% and F score of 0.53. Figure 1 further compares F-score by genre on the test data.

Multinomial Naive Bayes (second columns in orange) attains much higher F-score than Binary Naive Bayes (first columns in blue) for less popular genres. The common between the two models is that they tend to have high recall for very popular genres such as drama or comedy, and high precision but low recall for less popular genres such as sport, war and western as illustrated in Table 3. This result is plausible given the skewed dataset, which makes the Naive Bayes models to predict the popular genres more often, and to predict the less popular genres only when it has strong belief.

**Multinomial GRU**



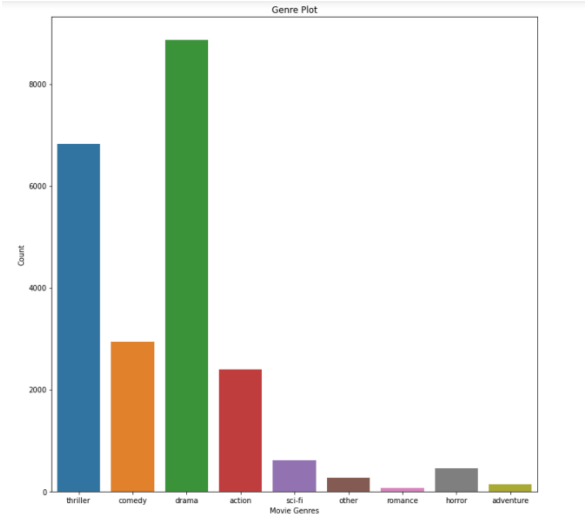

This subsection discusses further the Multinomial GRU model. Tab. 4 shows Multinomial GRU's predictions on some random test examples. These few examples illustrate that predicting movie genres is a challenging task even for human. Plot summaries do not convey all the information about a movie, and are sometimes ambiguous. The first example is about a guy being misunderstood by his girlfriend, who becomes upset and runs away. The correct genre is romance but the model predicts drama, which is quite plausible. In the second example, the model correctly predicts western, but incorrectly predicts action, perhaps due to "chases and catches" or "robber". In the third example, the predicted genres are drama and thriller while the true genres are drama and comedy. However, the plot with details such as "stalking", "murder" and "more

people to kill" suggests thriller more than comedy. It should be noted that the Jaccard index is only 33.3% in this case. In the fourth 8 example, there is no sign of musical unless one knows that Frank Sinatra is singer, while biography seems a reasonable choice. In the fifth example, the model correctly predicts comedy and family, but fails to include fantasy, mystery, romance, and sci-fi. Of these missed genres, it is possible to detect mystery or even fantasy from the plot summary, but there is no clue to romance or sci-fi. As the only sign of comedy in the plot summary is "many comical situations", I changed it into "many unexpected situations" and the model predicts family and horror instead. The probabilities of family, comedy and horror are changed from 33%, 27% and 9% to 29%, 13% and 21%, respectively. The sixth and seventh example shows the model's bias towards the most popular genres such as drama or comedy. In the sixth example, the model assigns a probability of 27% to drama, 24% to horror, 15% to thriller and 14% to mystery and selects only drama and horror, although there is no information in the plot summary suggesting drama. Similarly in the seventh example, the model assigns a probability of 46% to romance, 25% to comedy and 11% to drama and predict romance and comedy. The correct genre is romance only. The model is also sensitive to some words in a plot. In the last example, the model assigns a probability of 22% to sci-fi, 15% to animation, 11% to action, 11% to fantasy, and 9% to horror, and it predicts sci-fi and animation. The model is obviously not biased towards animation because only 6.2% of train examples belong to the animation genre. It turned out that the culprit is "miniaturized". When removing this word from the plot, the model assigns 31% to sci-fi, 13% to action, 12% to horror, 7% to fantasy and 7% to animation, and it correctly predicts only sci-fi. The first example also illustrates this problem. When "drives away with her" is changed into "falls in love with her", the model changes its prediction from drama to drama and romance. Finally, I qualitatively evaluate the model's learned word embeddings by looking at the nearest words to some random words. The nearest words to a

certain word are those in the vocabulary with embedding vectors that have the smallest cosine distance with the word's embedding vector. Table 5 shows top 10 nearest words of 6 nouns, 6 verbs, and 6 adjectives. Nearest words to nouns are very relevant. The results, however, are mixed for verbs and adjectives. Words such as kill, love, escape and cruel have relevant nearest neighbors, but manipulate, preserve, arrogant, and beautiful have totally irrelevant neighbors. One possible reason for this behavior is that verbs and adjectives are often used in a wider range of contexts than nouns. Hang's neighbors such as "smothering" or "re-connect" seem related to different meanings of the word. Miniaturized is an interesting example as it appears only 7 times in the train data, making the context specific. As a result, among its nearest words are karaati, gnomes, alchemist, meadow and fairy-tale, which may explain its influence on the model's prediction of animation in the previously discussed example.

# CHAPTER-4

# TESTING THE SYSTEM, SOFTWARE, TABULATION, GRAPHS




```
In [92]: wordcloud3 = WordCloud(background_color='white', width=3000, height=2500).generate(thriller)
         plt.figure(figsize=(8,8))
         plt.imshow(wordcloud3)
         plt.axis('off')
         plt.title("Words which indicate 'thriller' genre ")
         plt.show()
```

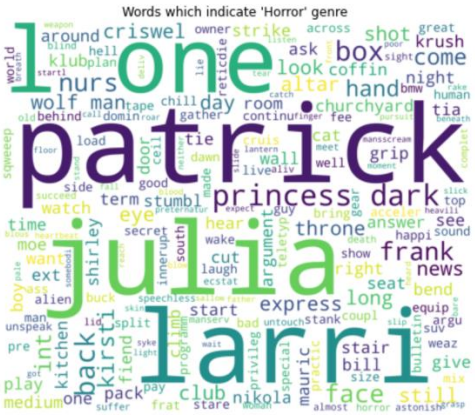

```
In [91]: from random import randint
         row = randint(0,test.shape[0]-1)
         sample_script = test.text[row]

         print('Script: {}'.format(sample_script))
         value = genre_prediction(sample_script)
         print('Prediction: {}'.format(list(genre_mapper.keys())[value]))
```

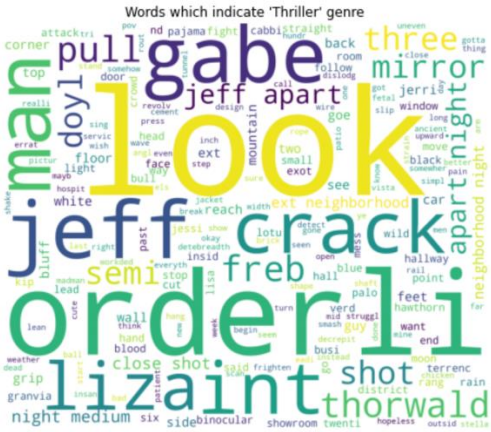
```
Script: e have to die?! Wynn restrains him. Pulls him away. The sheriff doesn't budge. WYNN Don't do this to yoursel
f, Sam. Let's go. Come on. Wynn ushers Loomis out of the room. Holdt stares apathetically. HALLWAY CONTINUOUS Wynn tr
ies to calm Loomis down. Jamie's doctor approaches. DR. BONHAM Dr. Loomis? WYNN What is it? DR. BONHAM I'm very sorry
... LOOMIS You let it get to her. How could you? DR. BONHAM Dr. Loomis, there's something else you should know. Durin
g surgery, we discovered that Jamie's uterus was hemorrhaging. We found this. displays a small vital It's placental f
luid. LOOMIS God in heaven. You don't tell me she was DR. BONHAM I estimate she gave birth no more than a few hours b
efore the attack. WYNN Then where's her baby? Jamie's covered body is wheeled out past them. Loomis regards Wynn with
a look of abject fear. LOOMIS I think I may already know ... CUT TO INT. TOMMY'S APARTMENT SAME CLOSE ON a chapter he
ading. Thorn The Devil's Rune. Beneath it, a bold depiction of the fami
Prediction: thriller
```

```
In [87]: newscore= score1*100
```

In [56]:
```
wordcloud2 = WordCloud(background_color='white', width=3000, height=2500).generate(horror)
plt.figure(figsize=(8,8))
plt.imshow(wordcloud2)
plt.axis('off')
plt.title("Words which indicate 'Horror' genre ")
plt.show()
```

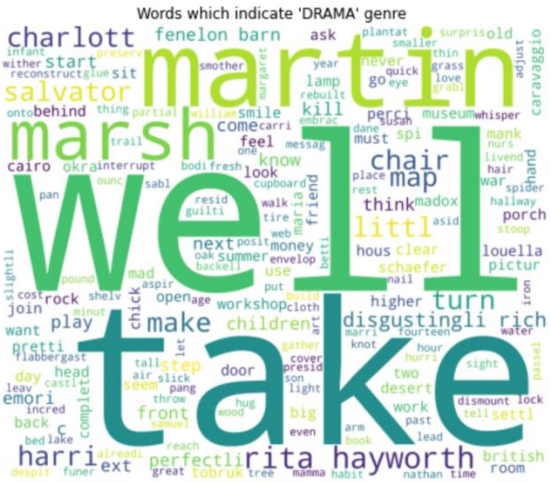



In [55]:
```
wordcloud2 = WordCloud(background_color='white', width=3000, height=2500).generate(thriller)
plt.figure(figsize=(8,8))
plt.imshow(wordcloud2)
plt.axis('off')
plt.title("Words which indicate 'Thriller' genre ")
plt.show()
```




In [54]:
```
wordcloud2 = WordCloud(background_color='white', width=3000, height=2500).generate(action)
plt.figure(figsize=(8,8))
plt.imshow(wordcloud2)
plt.axis('off')
plt.title("Words which indicate 'ACTION' genre ")
plt.show()
```

In [53]:
```
wordcloud1 = WordCloud(background_color='white', width=3000, height=2500).generate(drama)
plt.figure(figsize=(8,8))
plt.imshow(wordcloud1)
plt.axis('off')
plt.title("Words which indicate 'DRAMA' genre ")
plt.show()
```




In [73]:
```
drama_words = []
for i in list(df[df['genre']==4].index):
    drama_words.append(corpus[i])

action_words = []
for i in list(df[df['genre']==1].index):
    action_words.append(corpus[i])

comedy_words = []
for i in list(df[df['genre']==3].index):
    comedy_words.append(corpus[i])


thriller_words = []
for i in list(df[df['genre']==8].index):
    thriller_words.append(corpus[i])


horror_words = []
for i in list(df[df['genre']==5].index):
    horror_words.append(corpus[i])


scifi_words = []
for i in list(df[df['genre']==7].index):
    scifi_words.append(corpus[i])


romance_words = []
for i in list(df[df['genre']==6].index):
    romance_words.append(corpus[i])


drama = ''
action = ''
comedy = ''
thriller = ''
horror = ''
scifi = ''
romance = ''
```

In [9]:
```
df.isna().any()
```

Out[9]:
```
id       False
text     False
genre    False
dtype: bool
```

In [10]:
```
df.drop('id', axis=1, inplace=True)
```

In [11]:
```
corpus = []
ps = PorterStemmer()

for i in range(0, df.shape[0]):
    dialog = re.sub(pattern='[^a-zA-Z]', repl=' ', string=df['text'][i]) # Cleaning special character from the dialog/s
    dialog = dialog.lower() # Converting the entire dialog/script into lower case
    words = dialog.split() # Tokenizing the dialog/script by words
    dialog_words = [word for word in words if word not in set(stopwords.words('english'))] # Removing the stop words
    words = [ps.stem(word) for word in dialog_words] # Stemming the words
    dialog = ' '.join(words) # Joining the stemmed words
    corpus.append(dialog) # Creating a corpus
```

In [12]:
```
corpus[0:2]
```

Out[12]:
```
['eadi dead mayb even wish int nd floor hallway three night orderli lead liza door orderli white guy open door step r
oom three white guy mid look wild straight jacket jerri liza reach end rope shake head int decrepit hospit room night
ball fetal realli head press cement tri sing jerri blue moon blue moon int nd floor hallway three night liza stand le
an rail wall orderli sure go know bad orderli okay liza start hall orderli follow orderli got new patient last week w
ant see liza wave hopeless stop chicken wire window end hall look light break jerri somewher orderli look gotta get b
ack work',
 'summa cum laud launch brand new magazin call expos homag miss juli conroy xenia ohio juli grin juli know find excel
editor chief ted yellow page juli let finger walk suddenli music chang peopl ted grin ted play song extend hand dare
ask danc juli take hand better ted juli begin danc kiss b g charli jimmi feign tear charli sucker happi end hug jimmi
hold start rise nelson hous cloud xenia ted v guess everybodi pretti much live happili ever parent give groceri store
descend cloud quickli find ext london buckingham palac day mom dad take pictur smooch front palac ted v manag sneak a
way second honeymoon']
```

In [16]:
```
len(corpus)
```

Out[16]:
```
22579
```

In [7]:
```
movie_genre = list(df['genre'].unique())
movie_genre.sort()
movie_genre
```

Out[7]:
```
['action',
 'adventure',
 'comedy',
 'drama',
 'horror',
 'other',
 'romance',
 'sci-fi',
 'thriller']
```

In [8]:
```
genre_mapper = {'other': 0, 'action': 1, 'adventure': 2, 'comedy':3,
                'drama':4, 'horror':5, 'romance':6, 'sci-fi':7, 'thriller': 8}
df['genre'] = df['genre'].map(genre_mapper)
df.head(10)
```

Out[8]:

| | id | text | genre |
|---|---|---|---|
| 0 | 0 | eady dead, maybe even wishing he was. INT. 2ND... | 8 |
| 1 | 2 | t, summa cum laude and all. And I'm about to l... | 3 |
| 2 | 3 | up Come, I have a surprise.... She takes him ... | 4 |
| 3 | 4 | ded by the two detectives. INT. JEFF'S APARTME... | 8 |
| 4 | 5 | nd dismounts, just as the other children reach... | 4 |
| 5 | 6 | breadth of the bluff. Gabe pulls out his ancie... | 8 |
| 6 | 7 | uilding. A MAN in pajamas runs out into the ra... | 8 |
| 7 | 9 | ELLES AND RITA HAYWORTH Just disgustingly rich... | 4 |
| 8 | 10 | Memphis goes back into the garage, Budgy cack... | 8 |
| 9 | 11 | e reels as the world spins. Sweat pours off hi... | 1 |

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
import nltk
import re
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/kishkindghildial/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

# DISCUSSION OF RESULTS

In this study, we perform multi-class movie genre classification from plot summaries using Bi-LSTM where the class labels are Thriller, Horror, Comedy and Drama. We predict the genre of a movie by combining the decisions given for sentences of its plot summary, which is called sentence-level approach. On the other hand, when we use the whole plot summary for training without dividing it into its sentences, we call it as document-level approach. In order to measure the performance of the proposed method, we compare it with two baseline methods under different settings. First, we train an ordinary RNNs model using sentence-level and document-level approaches. We use the same representations used for our method while training RNNs model. Second, we train a logistic regression (LR) classifier using both settings in a similar way. While training LR model, we obtain Bag-of-Words (BoW) representation for each plot summary or sentence for document-level or sentencelevel approach, respectively. Then we fill the vector of plot summary or sentence with term frequency inverse document frequency (TD-IDF) [17] values. IDF weight for each word is calculated using only training dataset. There exist 27.336 unique words in the training dataset. Accordingly, each input is represented by 27.336 dimensional feature vector. Moreover, we also compare the results of our method with Bi-LSTM model trained with document-level approach. Note that, for all methods, the same pre-processing step is applied explained in Section II-A.

# CHAPTER-5

# CONCLUSION

This project explores several Machine Learning methods to predict movie genres based on plot summaries. This task is very challenging due to the ambiguity involved in the multi-label classification problem. For example, predicting adventure and thriller against true labels of adventure and action yields a Jaccard Index of only 33.3%. Word2vec+XGBoost performs poorly as the average of embedding vectors of words in a document proves a weak representation. A future direction is to apply doc2vec [19] to learn richer representations of documents. Experiments with both Naive Bayes and GRU networks show that combining a probabilistic classifier with a probability threshold regressor works better than the k-binary transformation and the rank methods for the multi-label classification problem. Using a GRU network as the probabilistic classifier in this approach, the model achieves impressive performance with a Jaccard Index of 50.0%, F-score of 0.56 and hit rate of 80.5%. There are several potential directions to improve the GRU network. As 46% of words in the vocabulary occur fewer than 20 times in the train data, most word embeddings get only a few weight updates. Using pretrained embeddings might be better for these words. In addition, about 75% of tokens are turned into “UNK”. As a result, the GRU network learns the same embedding for these words. Finding a way to adapt the UNK’s embedding based on context might make the network more powerful. Finally, the data is highly skewed, making the model biased towards popular genres such as drama or comedy. Dealing with this issue would further improve the performance.

# SCOPE FOR FURTHER WORK

The final results of the classification problem are mentioned in the result section. It is obvious that the total accuracy is not high, this could be because the nature of the plot summary that the team used as an input to the algorithm. The plot is mostly two or three sentences long; this size of text does not provide much useful information about the genre. The issue can be addressed by creating a more thorough and detailed database of movie storylines.

The threshold value plays a significant role in predicting genres for different movies, that means the final accuracies are very sensitive to changes in the threshold value. The current approach in calculating the threshold was explained in the training section of the report. Obviously, better approaches for calculating the threshold can be researched and deployed.

Another suggestion for future work is testing other classification methods in order to be able to compare results with each other. Support Vector Machine could be one of the methods to put to the test.

The plot summary used in this research proved that predicting a movie genre using only a short paragraph of the storyline summary, is not reliable. For future work, it would be a good practice to create a more thorough and more detailed movie plot database, a database that provides useful and meaningful information about the genre of the movies.

# REFERENCES


[1] Imdb data. ftp://ftp.fu-berlin.de/pub/misc/movies/database/.

[2] Xgboost python package. http://xgboost.readthedocs.io/en/latest/build.html.

[3] Pretrained google news vectors. https://code.google.com/archive/p/word2vec/, 2013.

[4] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016. [6] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies

with gradient descent is difficult. IEEE transactions on neural networks, 5(2):157–166, 1994.

[7] Alex Blackstock and Matt Spitz. Classifying movie scripts by genre with a memm using nlp-based features, 2008.

[8] Leo Breiman. Arcing the edge. Technical report, Technical Report 486, Statistics Department, University of California at Berkeley, 1997.

[9] Tianqi Chen and Carlos Guestrin.xgboostt: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pages 785–794. ACM, 2016.

[10] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259, 2014.