Movie Success Prediction Using BERT, LSTM, and Natural Language Processing

Manav Patel
Department of Electrical and Computer Engineering
Toronto Metropolitan University
Toronto, Canada
manav1.patel@torontomu.ca

***Abstract*—In a billion-dollar industry like the film and entertainment industries, it is important to have a thorough idea of how a script and film might perform at the box office. The 'Movie Success Predictor' aims to approximate the success of a future movie by analyzing data from IMDb and TMDb datasets and movie scripts from IMSDB. Key features were extracted including numerical, categorical and textual data. A Random Forest Model was developed to predict the ratings of a movie, by using features such as average ratings, vote counts, runtime, actors, etc. For the script sentiment analysis, a neural network was used. The neural consisted of BERT embeddings to contextual reasons and LSTM layers for capturing the sequential nature of the text data from a movie script. The data used was preprocessed using methods such as normalization and one-hot encoding. Training the model involved tuning hyperparameters and using Mean Squared Error for error evaluation. The model showed that certain features can be strong predictors of a movie's success and that the readability of a script can influence audience reception.**

***Index Terms*—Movie Success, Natural Language Processing, BERT, LSTM, Neural Networks, Random Forest, Sentiment Analysis, Feature Engineering**

I. Introduction

Neural Networks can be used to help predict the success of an upcoming movie, which can be beneficial to the billion-dollar film industry, The easy access to and knowledge of historical data and machine learning techniques has made it possible to leverage quantitative and qualitative analyses to make better movie success predictions. This paper explores the possibility of predicting the success of a movie using past data movie performance data and script data.

The movie datasets for this are sourced from IMDb and TMDb. The datasets included movie attributes such as cast, crew, ratings, release year, budget, runtime, etc. In addition to this, movie scripts were web scraped from the IMSDB site, into text files. A Random Forest model was used for predicting overall movie ratings and success and a neural network with BERT embeddings and LTM layers was used for script sentiment analysis to classify the scripts into various sentiments and calculate a sentiment score.

This two-model approach solves many challenges in predicting movie success. By combining the text analysis of a script with the structured data from IMDb and TMDb, this project shows a view of different factors that could influence a movie's potential success.

The sections of this paper are orgnized as follows: Section II. Related Work, Section III. Data Collection and System Models, Section IV. Results, and finally References. Section II. is about past work done related to this topic and past findings. It goes over literature and and different advancements that have been made related to this topic. Section III. goes over the different datasets and data sources used and the preprocessing steps, along with the architectures of the models implemented. Section IV. summarizes the of the findings evaluating the model.

II. Related Work

A study from an IEEE Computing Conference used machine learning algorithms such as Decision Tress, K-Nearest Neighbours, XGBoost and Deep Neural Networks, achieving an accuracy of around 90%. They ran into challenges regarding image resolution and false positives [5]. In another paper, a model was created to predict how profitable a movie can be using a neural network trained on social media data related to films. They used methods such as SVM and saw similar results to the previous group of researchers [4]. Another group of researchers from India developed a model to predict the success of a movie based on social media comments and tweets after a movie's trailer is released. They used a Random Forest Classifier with a Fuzzy system to classify a movie as a hit, semi-hit or a flop [6]. Three different groups of researchers developed models based on features like cast, producer, director, genres, etc and used Random Forest for classifying movies as hit or flop. Ruwantha et al. proposed an LSTM approach. Over 2.8 million tweets for various movies were transformed into feature vectors for analysis. The LSTM model was trained to classify tweets as positive or negative, which can be used to determine the success of a movie [11].

III. Data Collection and System Models

A. Dataset

For this project, we used three separate sources of data. The first dataset was the IMDb dataset [**?**]. This dataset contains detailed metadata about movies, such as ratings, runtime, release year, genres, etc. It originally contained 1,443,183 entries, but only 10,000 were chosen as the initial dataset due to the large size of the original dataset. Additionally, the TMDb dataset was used, which provided information mainly about movie casts and crews. This data was important for understanding the correlation between the popularity of cast or crew members and the movie's success [**?**]. For movie script analysis, 1,000 scripts from the IMSDb dataset were web-scraped and saved in text format. This data was used for sentiment analysis by extracting textual features using NLP techniques. Preprocessing techniques, such as normalization and encoding, were applied to these datasets to identify the features for the model. Missing values were either removed or the corresponding entries were ignored due to the large size of the dataset. The following equation was used for min-max scaling:

$$X_{\text{scaled}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{1}$$

Here, X is the original value, and X_{min} and X_{max} are the minimum and maximum values, respectively. This method is used to scale values between 0 and 1, or any given range, so they can be used as features for the model.

Data analysis was conducted on the data to look at patterns and insights in the data. The scatter plot in **Figure 1** shows the relationship between IMDb ratings and the number of votes received by the movie. From this figure, it is evident that popular movies receive a high number of votes compared to less popular movies which they don't receive that many votes. This shows how the number of votes a movie receives and the performance of a movie is correlated. Moreover, **Figure 2** is a figure of a histogram which represents the distribution of IMDb Average Ratings. This distribution shows that the average rating or distribution of the ratings was around 7. This shows how most movies have average to high ratings. This tells us that we cannot just rely on past ratings of a movie, other features are needed.

Feature Engineering was key to achieving good accuracy in the model. To come up with and derive the final features for the models, Categorical features such as 'genres' and 'titletype' were transformed using one-hot encoding. In one-hot encoding, categories are converted into binary values and each category would have either a 0 or 1 value, indicating whether it exists and has data, or not. This was done for the IMDb and TMDb datasets. for the IMSDb movie script dataset, sentiment analysis was done by assigning a numerical score to each script, which indicated the overall sentimental

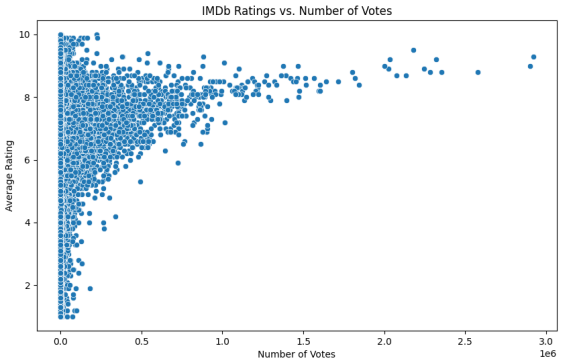

Fig. 1. Relationship between IMDb ratings and the number of votes received by the movie.

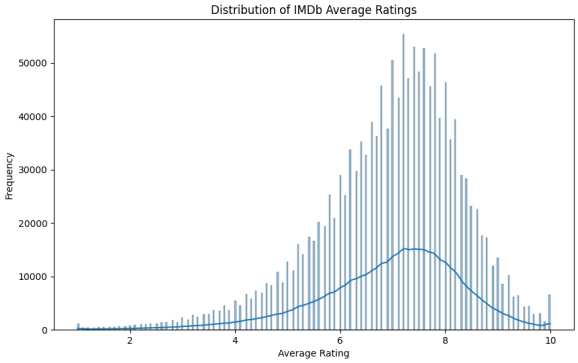

Fig. 2. Distribution of IMDb Average Ratings

tone of the text. Tis score was calculated using the Flesch-Kincaid Score formula. The Flesch-Kincaid Score formula is as follows:

$$\text{Flesch-Kincaid Score} = 206.835 - 1.015\left(\frac{\text{Total Words}}{\text{Total Sentences}}\right) - 84.6\left(\frac{\text{Total Syllables}}{\text{Total Words}}\right) \tag{2}$$

Various features were in the dataset, but nine of the features had a key influence on the target variable. This was highlighted mainly in the Random Forest Model.

B. Model Development and Architecture

In this project, two machine learning models were developed, a Random Forest model and a neural network incorporating BERT embeddings and LSTM layers.

The Random Forest model was trained on the IMDb and TMDb datasets using attributes such as crew, cast, genre, etc. Hyperparameter tuning was performed using GridSearchCV. Parameters such as *n_estimators*, *max_depth*, and *min_samples_leaf* were optimized. **Figure 3** illustrates the

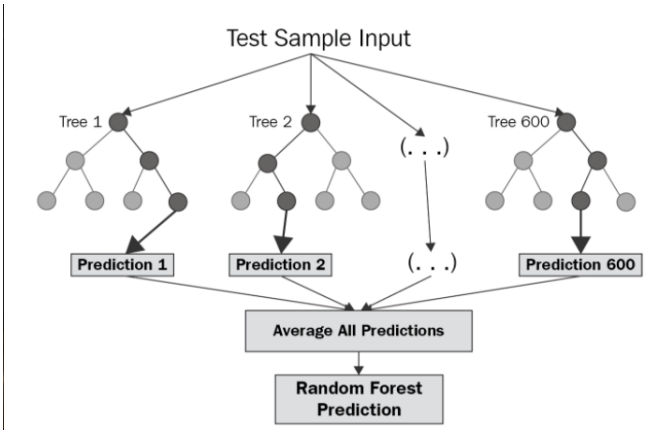

Fig. 3. Random Forest Model [17]

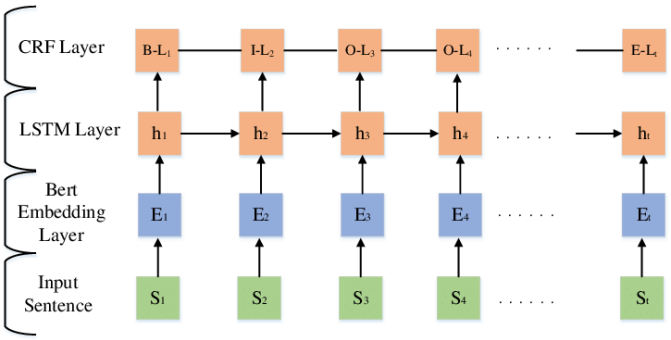

Fig. 4. BERT-LSTM Neueral Network [18]

process of how predictions are made using the Random Forest Model.

I implemented a neural network model integrating BERT embeddings and LSTM layers for the analysis of movie scripts. BERT is a model able to capture the context of words in a sentence. The IMSDb scripts were tokenized using the BERT tokenizer, which was then passed through the BERT model. These embeddings were fed into the LSTM layer, which is used to handle sequential data. There were additional two layers which included the ReLU activation function. Here the model was trained on 1000 scripts. **Figure 4** illustrates the neural network architecture of a BERT-LSTM Neural Network.

C. Evaluation Metrics

Mean Squared Error (MSE) was used as the loss function to measure the difference between the target values and the predicted data, as shown in the following equation:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{4}$$

For the Random Forest model, the model achieved a Root Mean Squared Error (RMSE) of 0.945 and an R^2 score of 0.201. Similarly, for the Neural Network model, the test evaluation yielded an MSE of 0.0022, indicating high accuracy in predicting sentiment scores.

IV. Numerical Analysis

To analyze the performance of the models, they were trained and tested using the datasets. All the experiments were conducted on Jupyter Lab, utilizing an Intel (R) i7 CPU @ 2.90 GHz, with 32 GB of RAM and a hard disk size of 1 TB. The models were implemented using the Keras API for deep learning and Scikit-learn for the Random Forest model.

The optimal hyperparameters selected for the models are defined in Table I.

TABLE I
HYPERPARAMETERS FOR MODELS

Hyperparameters	BERT-LSTM	Random Forest
Embedding Size	768	-
LSTM Hidden Layers	1	-
Neurons (LSTM)	64	-
Activation Function	ReLU	-
Dense Layers	2	-
Neurons (Dense)	128, 64	-
Output Layer	1	1
Epochs	40	-
Batch Size	32	-
Learning Rate	0.001	-
Optimizer	Adam	-
Bootstrap	-	True
Max Depth	-	10
Min Samples Leaf	-	4
Min Samples Split	-	2
Estimators	-	200

BERT-LSTM Model: The BERT-LSTM model was used to predict the sentiment score of movie scripts. The MSE was 0.0022. Despite the model's strong performance, it struggled with highly complex scripts.

Random Forest Model: The Random Forest model was used to predict the movie's success rating based on metadata and sentiment scores. The model achieved an RMSE of 0.945 and an R^2 score of 0.201. For this model, it would have been useful to include additional features.

The analysis shows the success and the limitations of the two models. The neural network was successful for sentiment analysis, and the random forest model gave reasonable results. Additional features and model refinement would enhance the outcome and performance of the models.

V. Conclusion

In this paper, we went over the possibility of accurately predicting the success of a future movie based on mainly its script and other features such as ratings, crew, etc. Integrating outputs of the Random Forest Model and the Bert-based

LSTM model has shown accurate movie success predictions. By combining metadata from TMDb and IMDb datasets, with sentiment analysis of movie scripts from IMSDb, we can conduct accurate approximations for a movie's success.

The Random Forest model used many movie attributes like the cast, crew, budget, ratings, runtime, release year, etc. to predict overall movie success. Because the model was able to handle a large number of features and inputs, it was a suitable choice to choose this model. According to the results, key features that impacted the performance of the model were ratings, votes, crew, and release year.

The neural network model with BERT embeddings and LSTM layers was great at analyzing the sentiment of a movie script. The model was able to successfully classify the scripts into different sentiments and compute a sentiment score. This allowed us to analyze the sentiment of a movie script and how the sentiment score related to how the audience received the movie.

Combining the results provided us with great accuracy and insights into how a movie might perform at the box office. Though there was success, there is also room for more improvement. Adding more features would have yielded better results. Also including social media sentiment could have further solidified the results.

In conclusion, the project shows the ability of machine learning and neural networks to predict the success of a movie, and such models and tools can contribute to a more efficient and successful film industry.

REFERENCES

[1] "IMDb Datasets." IMDb. Available: https://www.imdb.com/interfaces/

[2] "The IMDb Dataset Details." IMDb. Available: https://datasets.imdbws.com/

[3] "TMDb 5000 Movie Dataset." Kaggle. Available: https://www.kaggle.com/dataset/tmdb-5000-movie-dataset

[4] T. G. Rhee and F. Zulkernine, "Predicting Movie Box Office Profitability: A Neural Network Approach," in 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, USA, 2016, pp. 665-670, doi: 10.1109/ICMLA.2016.0117.

[5] N. Darapaneni et al., "Movie Success Prediction Using ML," in 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 2020, pp. 0869-0874, doi: 10.1109/UEMCON51285.2020.9298145.

[6] T. Sharma, R. Dichwalkar, S. Milkhe, and K. Gawande, "Movie Buzz - Movie Success Prediction System Using Machine Learning Model," in 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 2020, pp. 111-118, doi: 10.1109/ICISS49785.2020.9316087.

[7] Z. Balfagih, "Decoding Cinematic Fortunes: A Machine Learning Approach to Predicting Film Success," in 2024 21st Learning and Technology Conference (L&T), Jeddah, Saudi Arabia, 2024, pp. 144-148, doi: 10.1109/LT60077.2024.10468906.

[8] J. Ahmad, P. Duraisamy, A. Yousef and B. Buckles, "Movie success prediction using data mining," 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Delhi, India, 2017, pp. 1-4, doi: 10.1109/ICCCNT.2017.8204173.

[9] A. Bhave, H. Kulkarni, V. Biramane and P. Kosamkar, "Role of different factors in predicting movie success," 2015 International Conference on Pervasive Computing (ICPC), Pune, India, 2015, pp. 1-4, doi: 10.1109/PERVASIVE.2015.7087152.

[10] R. Dhir and A. Raj, "Movie Success Prediction using Machine Learning Algorithms and their Comparison," 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), Jalandhar, India, 2018, pp. 385-390, doi: 10.1109/ICSCCC.2018.8703320.

[11] W. M. D. R. Ruwantha, K. Banujan and K. Btgs, "LSTM and Ensemble Based Approach for Predicting the Success of Movies Using Metadata and Social Media," 2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), Zallaq, Bahrain, 2021, pp. 626-630, doi: 10.1109/3ICT53449.2021.9581601.

[12] A. Kanitkar, "Bollywood Movie Success Prediction using Machine Learning Algorithms," 2018 3rd International Conference on Circuits, Control, Communication and Computing (I4C), Bang

[13] IMSDb, "The Internet Movie Script Database," [Online]. Available: https://www.imsdb.com.

[14] Hugging Face, "BERT — transformers 4.9.0 documentation," [Online]. Available: https://huggingface.co/docs/transformers/en/modeldoc/bert. [Accessed: 02-Aug-2024].

[15] GeeksforGeeks, "Deep Learning — Introduction to Long Short Term Memory (LSTM)," [Online]. Available: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/. [Accessed: 02-Aug-2024].

[16] IBM, "Random Forest," [Online]. Available: https://www.ibm.com/topics/random-forest. [Accessed: 02-Aug-2024].

[17] M. Vaddi, "Random Forest Regression," Level Up Coding, [Online]. Available: https://levelup.gitconnected.com/random-forest-regression-209c0f354c84. [Accessed: 02-Aug-2024].

[18] M. Vaddi, "Random Forest Regression," Level Up Coding, [Online]. Available: https://levelup.gitconnected.com/random-forest-regression-209c0f354c84. [Accessed: 02-Aug-2024].