Anime Genre Text-Based Multi-Class Classification using Synopsis

Mohamed Ali
Computer Science
Haverford College
mali1@hc.edu

Menna Khaliel
Computer Science
Bryn Mawr College
mkhaliel@bmc.edu

Joseph Tadrous
Computer Science
Haverford College
jtadrous@hc.edu

Abstract

In this project, we aim to classify anime genres based on the anime synopsis. We train our model on a Kaggle database that was crawled using Animelist API.

Synopsis could be a useful indicator of the story line and genre of an anime and want to investigate whether it can predict the genre to which the anime belongs. After preprocessing our data and conducting some exploratory data analysis and visualizations to better understand the data, we loaded pre-trained GloVe embeddings and the token based text embedding trained on English Google News 200B corpus. We used our GloVe embeddings in two different LSTM models with unbalanced dataset and used Google News pre-trained embeddings in other basic sequential model and LSTM bidirectional model, both with balanced dataset. We trained and compared the models' performance and finally reached an accuracy of 65.4% in the LSTM bidirectional model with balanced dataset and one-hot-encoding of labels.

1 Introduction

Anime has a large fan base and anime genre classification can be very helpful in suggesting new animes of the same genre to users. Since most anime fans get the information they need to decide whether to watch an anime or not from the synopsis, anime synopsis can be useful in anime genre classification. Having a strong model that can predict anime genre can be of great use in predicting the genre of new animes that has not been classified. Our goal is to classify anime genres based on just the wording of the synopsis.

Current methods use a variety of data to classify anime genres, including movie trailers, video clips, subtitles, synopsis, and posters. Since we believe synopsis can be the most telling feature of an anime's genre, we utilize LSTM to analyze it and take into account the word ordering. We use logistic regression as a base to compare and evaluate the LSTM accuracy. We use GloVe word embedding technique because we think that word similarity can play an important role in the classification process.

2 Related Work

Binary and Multi-class Classification problems and their proposed solutions and methods have a myriad of research and projects contributing to them. Application of classification models can be seen used in many fields and to aid with many problems. Whether it's social media, e-commerce, security, business, marketing, or biology, classification models contributed to solving critical problems. There are also several projects and applications for classification models in the media and entertainment industry, which our project contributes to. Many models for multi-classification have been explored and proposed to help with media tagging and labeling.

In their paper, "Multi-Label Classification Method for Multimedia Tagging" the authors explored the problem of automated multimedia tagging to increase the efficiency of media search and retrieval. The research investigates the performance of their developed multi-label decision trees which uses clustering to separate labels and performs iterative split decision to achieve best binary split. The model has been tested with several datasets to test its performance. In one of the tests, the model is tested on tagging music artists and bands based on textual descriptions. The Proposed multi-label decision tree model performed better than the compared classification models for almost 50% of labels (Ma Sethi Patel, 2010).

Many papers focus on classifying media through context data. In their project, Rafael B. et al go further to use a multimodal approach for multi-label classification for movie genres. Their model used not only synopsis, but also movie trailers, video clips, subtitles, and posters. After testing with several models and descriptors, the best performance on textual data (subtitles and synopsis) was achieved using LSTM models. They calculated F-score for the LSTM models and the results were 0.674 for and 0.725 for movie subtitles.

3 Approach

3.1 Data

The dataset used in our experiments contains information about more than 18,000 anime crawled from https://myanimelist.net/. The dataset contains a list of anime with the corresponding features including titles, genres, synopsis, popularity, etc. We are only interested in the genres and synopsis for the purposes of our experiments. Our input for the model is the synopsis embedded with the GloVe embeddings and our output is the appropriate genre.

3.2 Preprocessing

Using pandas, we read the anime dataset in a panda dataframe. We dropped all the empty (null) rows. In addition, we kept only the synopsis and genre columns. Then, the genre column was cleaned through creating a new genre column and keeping only the first genre and removing all non-character symbols. The genre “Sci-Fi” was changed to “SciFi” since the tokenizer considered “Sci-Fi” two separate tokens: “Sci” and “Fi”. We had a total of 43 genres. However, two of the genres (ShoujoAi, Yuri) never appeared as the first genre in our dataset. Thus, the length of our genres were considered to be 41 in the single label classification. We, then, clean the synopsis column through creating a new column and removing all the stop words, tags, punctuations, etc. We split our dataset into training and test datasets with 80-20 split. The training dataset contains 14,629 rows. Upon conducting some exploratory data analysis, we found that the data is significantly unbalanced for genres labels. We found that Action, Comedy, and Hentai accounted for 22.1%, 17.6%, and 10.8% respectively, which is approximately half of the dataset. Figure (1) shows a pie chart with information on the top5 genres distribution.

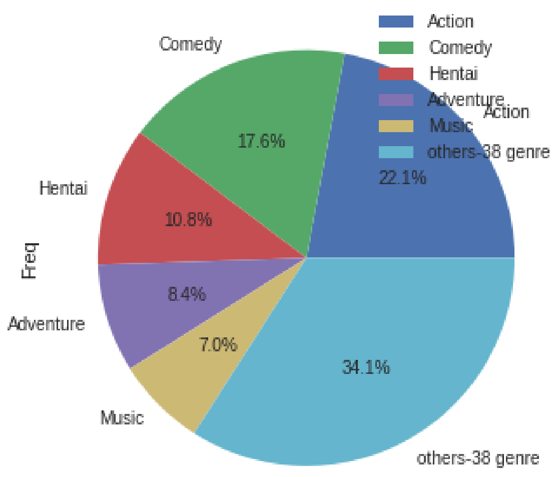

Figure 1: Pie chart showing distribution of top5 and all other genres.

devised two ways to feed our labels to the neural networks. Firstly, a Tokenizer was generated for our labels using a list of distinct genre labels. Then, we convert the training and test labels to their corresponding numerical tokens. Hence, we can represent the genres with numbers. We used sparse categorical cross entropy to evaluate the loss. Secondly, one-hot encodings were generated for our labels. Each label was represented by a one-hot encoded vector of length 41 where all indices are zeros except for the index of the relative label. In this case, we changed the sparse categorical cross entropy to just categorical cross entropy which is suitable for one-hot encodings.

3.3 GloVe

There are many techniques that can be used for word embeddings. However, simple techniques like Bag Of Words fail to capture context, position in text, semantics, etc. Therefore, we decided to use GloVe embeddings. GloVe embeddings is helpful in computing word similarity which is profound in our approach of predicting the genres based on its synopsis. We load the word embeddings from Stanford GloVe trained on Wikipedia 2014 + with 6Billion tokens, 400K vocab, uncased, 200 dimension vectors. Using the GloVe vectors generated, we converted the synopsis into vectors and created the paddings accordingly.

3.4 Model description

3.4.1 LSTM with unbalanced data, GloVe, and numeric encoding

For our first model, we used Long Short-Term Memory (LSTM). An LSTM has a similar control flow as a recurrent neural network. However, it has operations within its cells that allow it to keep or forget information. Our LSTM model has 3 layers of dense 128 and two hidden layers. The last layer of our neural network contains 41 neurons which represent every class with activation function 'softmax'. The labels passed to our model were encoded using a numeric tokenizer. The following parameters are used in our model: batch size=64, epochs=15, optimizer= adam, learning rate= 0.003, loss=sparse categorical crossentropy

3.4.2 LSTM with unbalanced data, GloVe, and one-hot encoding

This model's architecture is essentially the same as the first one. However, in this model, the labels are encoded using one-hot encodings instead of using a tokenizer. Our aim is to compare the results of this approach to the first one. We don't expect any significant differences in the result. Also we used categorical cross entropy instead of sparse to evaluate the loss. The following parameters are used in our model: batch size=64, epochs=15, optimizer= adam, learning rate= 0.003, loss=categorical crossentropy

3.4.3 Sequential model with balanced data and one-hot encoding

For our third model, we used a sequential model that consists of 3 layers of dense 128, one layer with dense 64, one layer with dense 32 and the final layer with 41 neurons and softmax activation. It also has 4 hidden layers. The labels were encoded using one-hot encoding. In this experiment, based on the frequency distribution of the labels, weights were assigned so that more dominant genres (such as action and comedy) are penalized in comparison to less common genres.

3.4.4 Bi-directional LSTM model with balanced data and one-hot encoding

For the final model, the model's architecture, data and encodings are similar to the sequential model. However, a bi-directional LSTM layer was added to the network. A bi-directional LSTM consists of two LSTMs: one takes the input in a forward direction, and the other in a backwards direction. We expect this model to be the most successful since it effectively increases the amount of information available to the network and improves the context available to the algorithm. For example, it has the knowledge of the words following and preceding a word in a sentence.

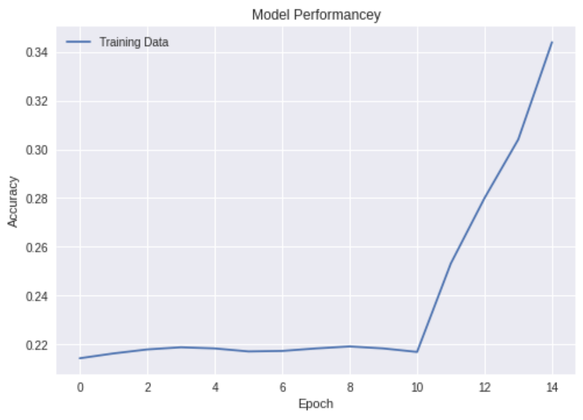

Figure 2: Plot of epochs against training accuracy for the base LSTM model.

4 Evaluation

4.1 Evaluation Methods

We evaluate each model by computing and plotting the model's accuracy on test data and the model's loss function. We compare models to see which model achieved the best classification accuracy among the four. We refer to our first model, LSTM with unbalanced data and vector numeric encoding, as the base model for all comparisons.

4.2 Results

4.2.1 LSTM with unbalanced data, GloVe, and numeric encoding.

This model had a low average training accuracy during the first 10 epochs, around 21%, but started to increase relatively fast after the 10th epoch, reaching an accuracy of 35% at epoch 14. The model scored a final test accuracy of 26.9%. The model's loss showed a similar trend in the other direction. Figures 2 and 3 show the plots for the model's accuracy and loss, respectively.

4.2.2 LSTM with unbalanced data, GloVe, and one-hot encoding.

Since the difference between this model and the base model is the label vector encoding, we expect the results to be very similar to that of the base model. Indeed, the LSTM with one-hot encoding

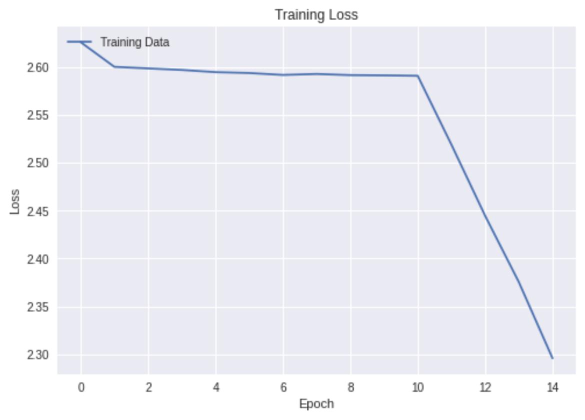

Figure 3: Plot of epochs against training loss for the base LSTM model.

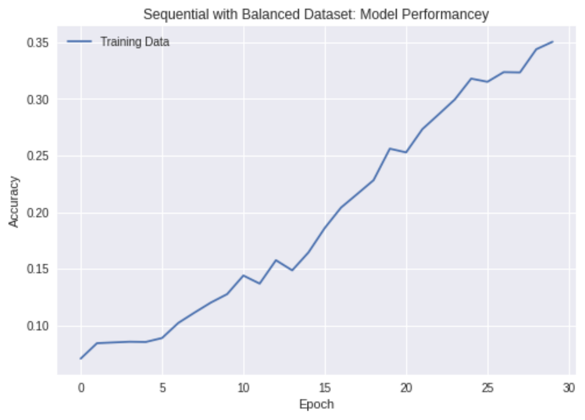

Figure 4: Plot of epochs against training accuracy for the sequential model.

model was only able to achieve 1% more accuracy than the base model. The plots for this model are very similar to the base model's plots and thus not included.

4.2.3 Sequential model with balanced data and one-hot encoding

This model scored the lowest test accuracy of all four models, with a final training accuracy of 35% after 30 epochs and a final test accuracy of 20.8%. The plots for accuracy and loss are shown in Figures 4 and 5.

4.2.4 Bi-directional LSTM model with balanced data and one-hot encoding

This model has proven to be the best model among the four, with a test accuracy of 65.4% after 30 epochs (more than double the base model). The model's accuracy increases steadily and rapidly each epoch and the loss significantly decreases. Figures 6 and 7 illustrate the training accuracy and

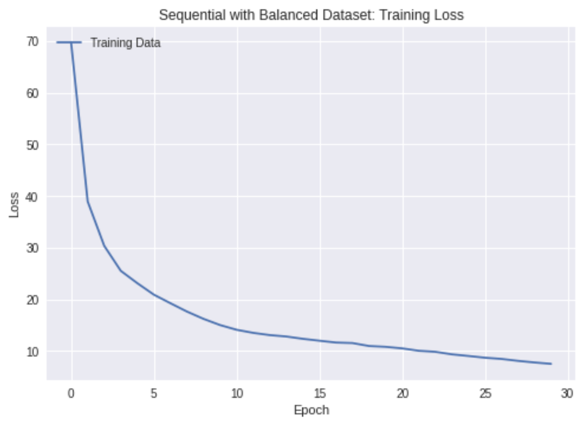

Figure 5: Plot of epochs against training loss for the sequential.

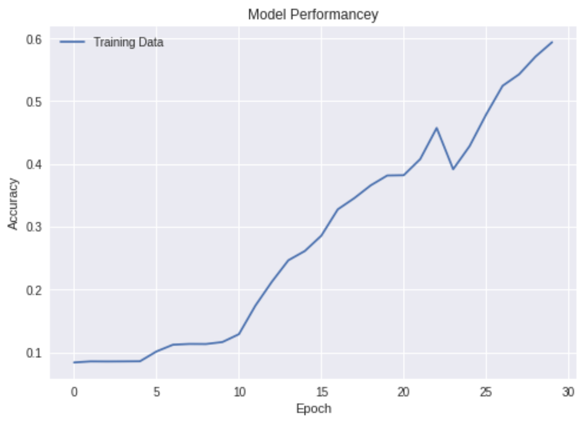

Figure 6: Plot of epochs against training accuracy for the bi-directional LSTM model.

loss trends.

5 Analysis

Upon inspection of the dataset, we found that there were multiple labels not strongly representative of the synopsis contextual indication of the genre. For example, some noticed synopsis mention 'fantasy', 'mage' and had a label of 'Action' genre instead of 'Fantasy' or 'Supernatural'. Although the 'Action' genre could make sense in the context of the anime story, it won't necessarily be the strongest or top genre classification for it. This would be okay if it was a multi-label classification problem, which it isn't. Obviously, the labels were not a perfect predictor or top tagging of an Anime's genre, this is most likely due to extracting the first genre out of the genre list for an anime and considering this first genre to be the single label. Since our approach here is to build a Multi-class single classification model and output one genre

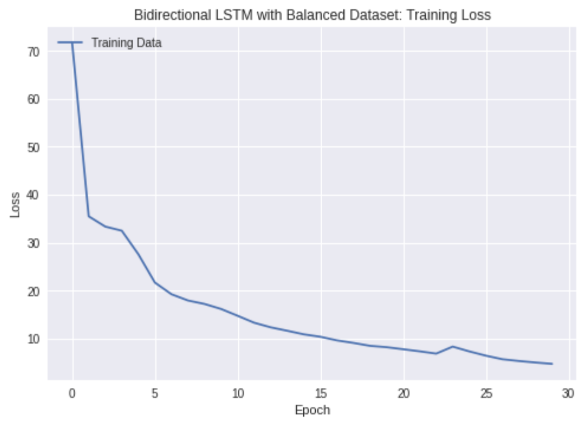

Figure 7: Plot of epochs against training loss for the bi-directional LSTM model

tagging per anime, choosing only one label is inevitable, but choosing the first label didn't prove to be a strong approach. As mentioned in the preprocessing section, total anime genres were 41 which generated the same number of classes for our classification model. However, when reflecting on the results of the final model, it is surprising that the model was able to achieve 65.4% accuracy, despite the fact that training cycles for the final model almost always started with very high values of loss and low values for accuracy, but this is most likely due to having many output classes in the last layer and having a relatively small dataset.

6 Conclusion

Our goal is to achieve a high-performance model for anime genre classification. We experimented with four different models, a base model of LSTM layer and GloVe embeddings and numeric encoding, an LSTM model with GloVe and one-hot encoding, a sequential model with pre-trained word embeddings and one-hot encoding, and a bidirectional model wit pre-trained word embeddings and one-hot encoding as well. As we expected, the first two models turned out to be very close in performance. We were expecting the bi-directional LSTM model to outperform the base model because of the added feature of backwards memory, but we were surprised by how much better it scored compared to it. With the sequential model being made out of just dense layers, we didn't have high expectations for it. Yet, it proves that, even with such low accuracy, LSTMs are an improvement over fully connected layers. In the future, we plan to clean and organize the dataset more to eliminate any potential bias or error. We will also run the models on different datasets and see if the performance of the four models is comparable to what we had with the current dataset.

References

Rajanna, Arjun Raj. 2015. *Deep Neural Networks: A case study for music genre classification* 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)

Kumar, Akshi and Rajpal. 2018. *Genre classification using word embeddings and deep learning*. 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)

Jefferey Pennington. 2014. *GloVe: Global Vectors for Word Representation*. Stanford

Ali Ertugrul. 2018. *Movie genre classification from plot summaries using Bidirectional LSTM*. 2018 IEEE 12th International Conference on Semantic Computing (ICSC)

Ma, Aiyesha Sethi, Ishwar Patel, Nilesh. 2018. *Multi-Label Classification Method for Multimedia Tagging*. .