Visual, Audio, Metadata Feature, Which Is the Most Predictive Feature in Movie Genre Classification

Jason Jia

1 Introduction

With the increasing need to classify movie genres to provide a customized experience for movie viewers as well as provide recommendation systems, machine learning is widely employed in this field (Sung & Chokshi). The genre of a movie contains a great amount of information within the word. It is important to highlight how different aspects of movie data can contribute to the genre. In this paper, we will discuss how the textual and audio, visual features utilized to classify movie genres. The features and class labels were derived from published datasets (Deldjoo, Constantin, Ionescu, Schedl, & Cremonesi, 2018; Harper & Konstan, 2015). We selected the most predictive feature based on its corresponding model performance and behavior.

2 Literature Review

There have been attempts in movie genre classification studies. Hoang implemented several machine learning methods including Naïve Bayes and RNN model by using summarized plots(Hoang, 2018). He achieved to identify genre in 80.5% of all cases. Additionally, visual and audio data have been taken advantage of in machine learning as well. Wehrmann and Barros implemented the Convolution neural network on multiple-genre classification by using the move trailers(Wehrmann, 2017). He showed efforts to classify the genres based on visual features with the CNNs model.

3 Model Training and Evaluation

In this report, we approached to analyze the textual data and visual, audio data separately. During preprocessing the dataset, we dropped the "YTId" feature as the YouTube link will not contribute to data mining.

3.1 Text analysis on the "*tag*"

We showcased how text-based data processed with Natural Language Toolkit to solve the multi-class classification challenge. We aimed to build a model that can predict the genre of a movie based on the title, year of release, and human-annotated tags. However, the tag feature contains information on human annotation which gives extra information of the movie plot and carries far more human sentiment information. Thus, we use only the tag features over the movie title.

A supervised classification algorithm was built after the label classes transformed by integer encoding. Additionally, an imbalance was observed in classes as we counted each class occurrence of the dataset. The class occurrence distribution is shown in the following figure:

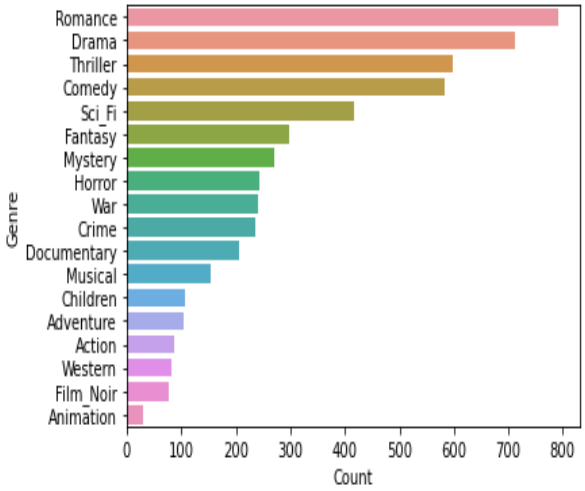

Figure 1 Label Class Distribution

During the data preprocessing, we also observed that the tag feature contains numbers of stop-words which ought to be removed since they carry little meaning to the text and increase the noise to the data. A text cleaning function was designed and applied with Lambda function to manipulate the data strings to extract better features from clean data and reduce the noises. We visualized the frequency of words before and post to text cleaning. These plots are shown in Appendix A.

Before training, TF-IDF was employed to convert text to features from the cleaned tag data due to the unproportionate word occurrence. A logistic regression model was built based on the training features and label classes. Label classes were inversely transformed to generate the prediction results. We also applied the grid search method to find the optimal model parameters. The logistic regression model eventually achieved performance shown in the following table.

```
In [201]: print(metrics.classification_report(classes,y_valid))
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         0
           1       0.00      0.00      0.00         0
           2       0.00      0.00      0.00         0
           3       0.00      0.00      0.00         1
           4       0.50      0.46      0.48        41
           5       0.00      0.00      0.00         3
           6       0.39      0.58      0.47        12
           7       0.53      0.30      0.38        77
           8       0.28      0.56      0.37         9
           9       0.25      1.00      0.40         1
          10       0.38      0.38      0.38         8
          11       0.10      0.20      0.13         5
          12       0.11      0.67      0.19         3
          13       0.43      0.30      0.35        74
          14       0.75      0.63      0.69        19
          15       0.43      0.34      0.38        35
          16       0.38      0.73      0.50        11
          17       0.00      0.00      0.00         0

    accuracy                           0.38       299
   macro avg       0.25      0.34      0.26       299
weighted avg       0.46      0.38      0.40       299
```

Figure 2 Logistic Regression Model Performance

During building the model, we used "average = micro" that is to use micro-averaged precision. As the micro-averaged precision of multiple classes is given by:

$$Precision_{\text{micro}} = \frac{TP}{TP+FP} = \frac{TP_{C1}+TP_{C2}+\cdots+TP_{Cn}}{TP_{C1}+FP_{C1}+TP_{C1}+FP_{C2}+\cdots+TP_{Cn}+FP_{Cn}}$$

As the micro-precision calculates the average of the error counts of each class, the classes with more instances have much more TP (true positives) and FP (false positives) comparing to classes with far fewer instances. This resulted in that it biases towards the performance of large classes in the dataset since we wish the model at least would be able to classify most instances. This is validated by assessing the individual F1-Score against the number of instances that exist in the validation dataset.

Consequently, larger classes are assigned with larger weights. Additionally, validating dataset has no records of some labels that were validated, those are the ones in a low frequency of occurrence. This model achieved an F1-Score of 38.5%. This model was evaluated and improved by lowering the threshold value as well as performing a grid search to find optimal parameters.

3.1 Deep Learning Audio and Visual Features

A baseline model based on a Multi-layer Perceptron classifier was built which achieved 17.3% accuracy.

We adopted another library *Keras* aiming to improve the performance. Another feed-forward and fully connected baseline model was built, which achieved an accuracy of 32.5% on the visual dataset. We specified some additional properties required when training the neural network. We improved the training of the network by finding the best set of weights that mapping inputs to classes in the dataset.

Our approach was to specify the loss function to evaluate a set of optimal weights, in conjunction with the optimizer to search through different weights for the neural network. More specifically, categorical cross-entropy was used as the loss argument. Since the cross-entropy calculates the difference between actual and predicted probability for all classes, we plotted the following graphs. The top one is cross-entropy loss over epochs for the training and test dataset, and the bottom graph shows the classification accuracy over epochs for the audio dataset.

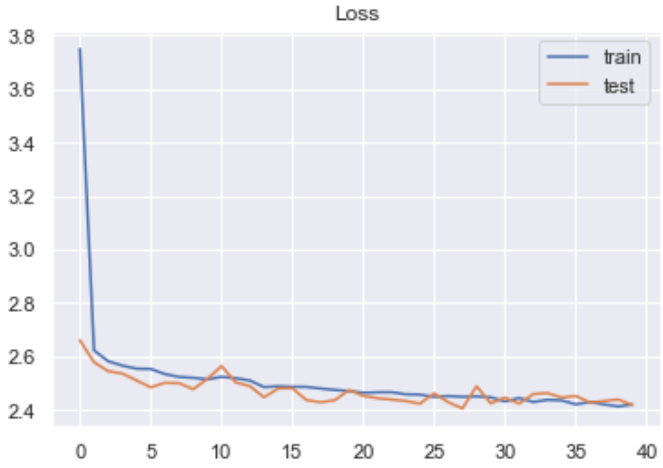

Figure 3 Cross-Entropy loss VS Epochs

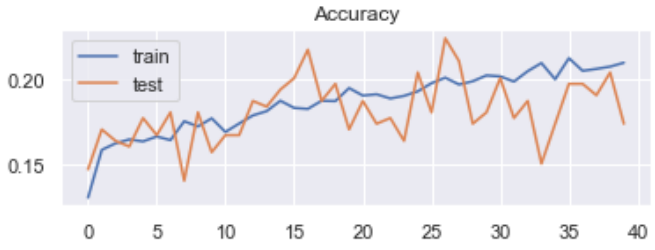

Figure 4 Accuracy VS Epochs

The plots show the model seems to have converged, although very bumpy. We adjusted some parameters according to the evacuation analysis so that the model learns a good mapping of rows of input to the output classes. We also defined optimizer as the efficient stochastic gradient descent algorithm "Adam". This version of gradient descent automatically tunes itself and returns good results in a range of

problems (Guides).

It is observed from the logistic regression model developed that it is simple and easy to implement a classification algorithm. In a multiple class problem, it can be easily generalized to map to multi-classes. During training the logistic regression model and neural network model, the former model took significantly less time as it takes less computing power. Whereas the latter modeling technique requires decent GPU which can be problematic when it is operated under a normal laptop. Particularly, it is observed that the training time is significantly proportionate to the number of hidden layers and neurons during training. Furthermore, overfitting occurred when we trained models with a large number of neurons. To avoid this, numbers of trial and error were performed before reducing the number of hidden layers to three. The number of neurons was determined by the following empirical formula suggested by Hagan (Demuth, 2014 #4):

$$N_h = \frac{N_s}{(\alpha * (N_i + N_o))}$$

Where:

Ni = number of input neurons.
No = number of output neurons.
Ns= number of samples in the training data set.
α = an arbitrary scaling factor usually 2-10.

The model with lower α value behaved more general and less overfitting during the trial and error process. Since we do know how noisy the training data is, we want to keep the model more general and not to use too many artificial neurons. Otherwise, the training set will be memorized, which will not be useful for the new test dataset. Thus, we want to limit the number of free parameters (which carry nonzero weight) to a small portion of features dimensions. We determined the number of neurons in hidden layers just enough to learn the features.

According to the training model audio dataset achieved higher accuracy than visual, resulted in approximately 4% difference will not conclude audio features are more predictive to this movie genre classification problem as the audio dataset has over 5 times more features than the visual dataset. However, the F1-score of visuals is much higher than the audio one. This true measure considers both precisions and recalls representing the overall performance.

4 Conclusions

Among the text metadata, visual and audio data, the metadata outperformed the other two types from accuracy and F1-score statistically. Furthermore, metadata features consist of far more useful lexical resources for each instance than visual and audio features. The tag features generated based on human annotation, which carries a clear tone and sentiment. Additionally, based on the word frequency plot, we observed that some words frequently present in certain classes, such as “thriller” in “Thriller” classes. Those highly anticipated words with high probabilities help the model to map to its corresponding class. In conclusion, text features are more favorable to predict movie features comparing to audio and visual features.

References

Guides, K. D. Adam. Retrieved from https://keras.io/api/optimizers/adam/

Harper, F., & Konstan, J. (2015). The MovieLens Datasets. *ACM Transactions on Interactive Intelligent Systems, 5*, 1-19. doi:10.1145/2827872

Hoang, Q. (2018). Predicting Movie Genres Based on Plot Summaries.

Sung, S., & Chokshi, R. Classification of Movie Posters to Movie Genres.

Wehrmann, J. (2017). Movie Genre Classification: A Multi-Label Approach based on Convolutions through Time. *Applied Soft Computing, 61*. doi:10.1016/j.asoc.2017.08.029

Jones, M., Way, D. & Shririan, Y. (2019). Genre-detection with Deep Neural Networks. Stanford CS 230: Deep Learning.

Appendix A

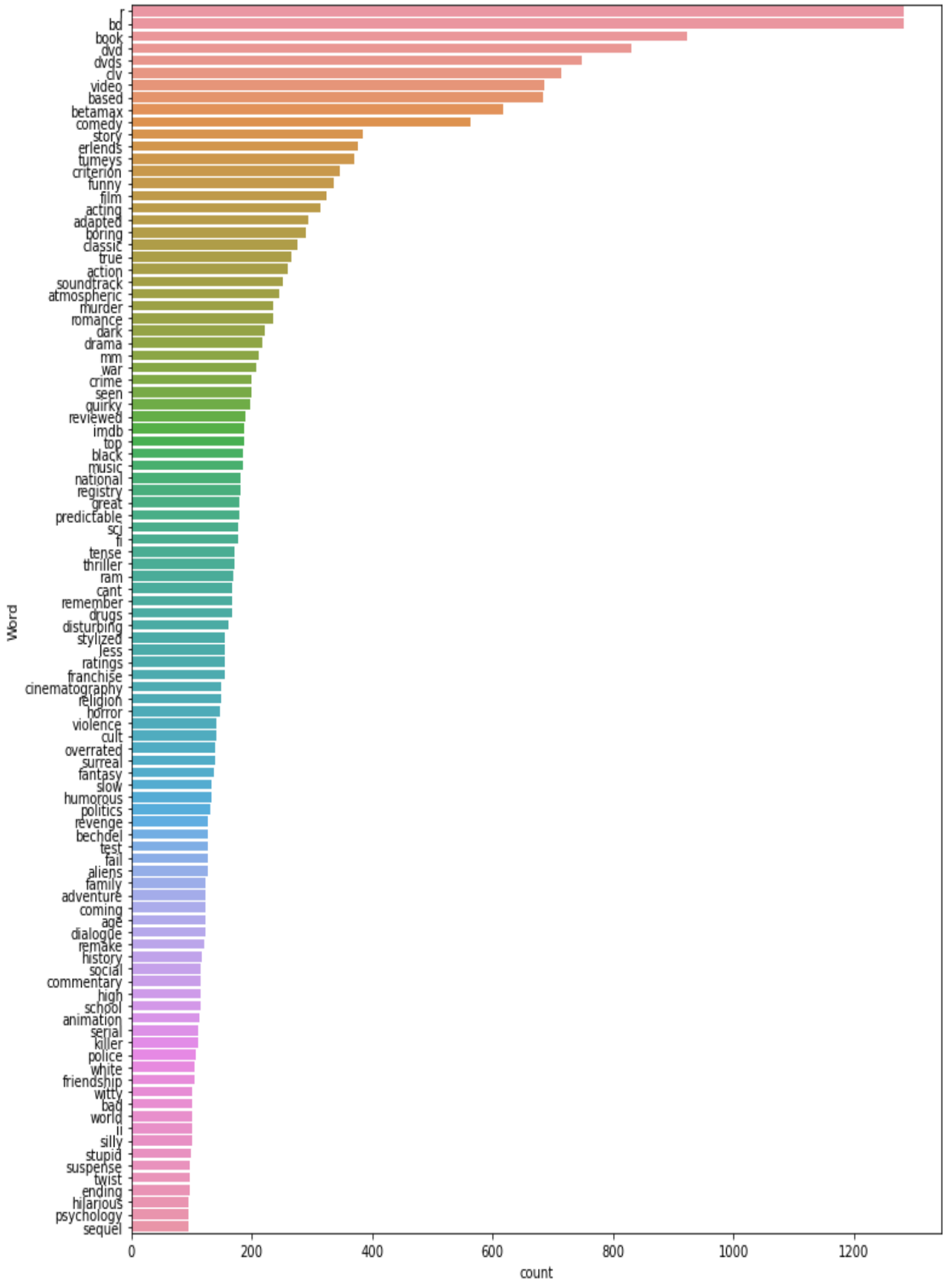

Figure 5 "Tag" Word Frequency (After Text Cleaning)