Movie Classification Project - Final Report

Jiejun Lu, Hongxiang Qiu, Weidong Xu, Zeyu Zhao
AC209B Group #18
Harvard University

1 Problem Statement and Motivation

In this project, we implemented several machine learning models to predict the genres of movies using their **overview descriptions and posters** (since we want to predict genres for new movies, we didn't use features like *ratings*, which are not available for new movies). We further compared those models and made a machine learning pipeline for the best model. **The best model is a concatenated model composed of an encoder for posters and an encoder for overview texts**.

Our pipeline can be used to automatically tag the genres of movies on movie-hosting websites like Netflix and movie-database websites like TMDB.

All code for scraping data and building the models are submitted with this report.

2 Introduction and Description of Data

In 2016, 736 new movies were released in US cinemas [1], with more on websites like Netflix and YouTube. And the number of new movies released each year is still growing. Therefore, movie genre classification, which helps people quickly find the movies they want to watch, becomes more and more important.

In our project, we built machine learning models to automate the movie genre classification task. The input features for our models are the overview description texts and poster images of movies and the outputs are the predicted genres of the given movies.

The first step of building any machine learning model is obtaining the training and test datasets. For this project, we scraped all 361,622 movies from TMDB using the python library *tmdbsimple*. As for data cleaning, we removed movies with short overview length (less than 10 words) since the majority of them are empty/'No Overview'/'Not found'/ etc., or not enough to describe the movie. We also removed movies without available posters as we would like to integrate poster information to our final model. We are interested in recent movies, so after data cleaning, we only kept the latest 30,000 movies, including their overview descriptions, poster images and ground truth genres. For TMDB dataset, there are 19 possible genres in total.

Before building our models, we performed the exploratory data analysis (EDA) and got some noteworthy findings on our data. Fig. 1 shows the histogram of the count of movies in corresponding genres. We can see the dataset is unbalanced. And intuitively, a movie can have multiple genres. For example, the well-known *Titanic* movie has genres *drama, romance and thriller*. Fig. 2 shows how genres appear together. It makes intuitive sense to see certain "exciting" categories group together, including crimes, horror, mystery, thriller, and "uplifting" categories group together, including animation, fantasy, family and adventure.

We also examined whether the overview descriptions and posters can be used to predict genres. Fig. 3 below shows posters for a documentary movie (left) and an animation movie (right). We can see their styles are pretty different. Fig. 4 shows the word cloud we generated for crime movies (top) and romance movies (bottom). We can see their key words are different ("Crime" v.s. "Love"). Therefore, we believe both the posters and overview descriptions can be useful to predict genres of movies.

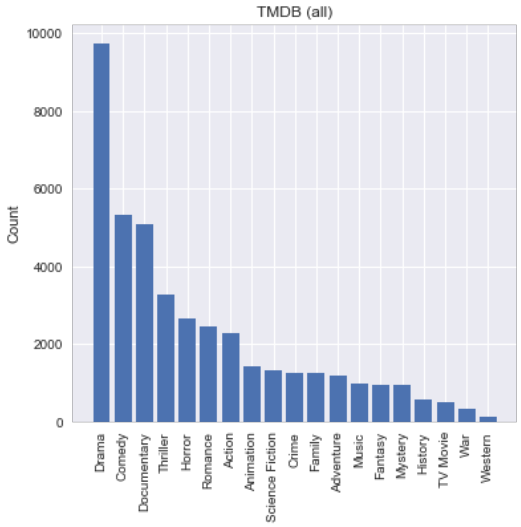

Figure 1: Histogram of Movie Genres

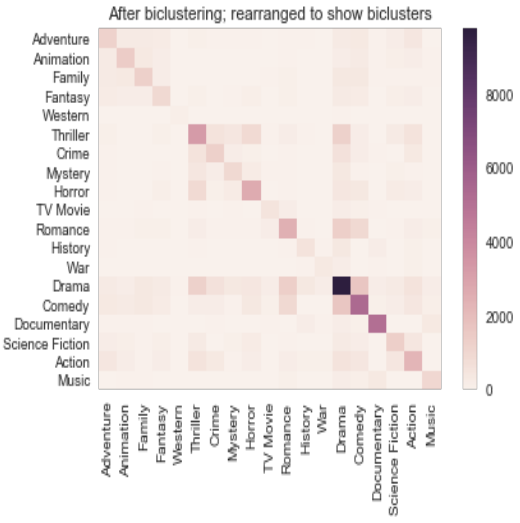

Figure 2: Co-occurrence of genres

Figure 3: Posters for Different Genres

Figure 4: Word Cloud of Different Genres

As aforementioned, movie genre classification is a multi-label classification problem. For models solving such problems, evaluation metrics for binary problems can not be easily applied (for example, if using AUC, we will now have 19 values for all categories instead of 1, and thus we can't rank our models easily). Previous work [2, 3, 4] used F_1-score (also called F-score or F-measure) as the evaluation metric. In our project, we choose **micro averaged F_1-score** [2] as the metric. The equations for calculating the micro averaged statistics are:

$$\begin{cases} \text{Precision: } P_{micro} = \dfrac{\sum_{j=1}^{k}\sum_{i=1}^{n} Y_i^j Z_i^j}{\sum_{j=1}^{k}\sum_{i=1}^{n} Z_i^j} \\ \text{Recall: } R_{micro} = \dfrac{\sum_{j=1}^{k}\sum_{i=1}^{n} Y_i^j Z_i^j}{\sum_{j=1}^{k}\sum_{i=1}^{n} Y_i^j} \\ F_{1-micro} = \dfrac{2\sum_{j=1}^{k}\sum_{i=1}^{n} Y_i^j Z_i^j}{\sum_{j=1}^{k}\sum_{i=1}^{n} Z_i^j + \sum_{j=1}^{k}\sum_{i=1}^{n} Y_i^j} \end{cases} \tag{1}$$

Y_i^j and Z_i^j in equation 1 means true and predicted label for sample i class j, respectively. By using $F_{1-micro}$ as the single metric, we can easily rank our models.

3 Related Work

Overview texts can be processed with natural language processing (NLP) algorithms. The conventional methods first generate features (typically bag-of-words, word2vec [5] and GloVe [6]), and then train classifiers on the generated features. In our project, we tried aforementioned features and multinomial naive Bayes (NB) and Support Vector Machine (SVM) classifiers.

In recent years, deep learning has shown remarkable performance in language modeling, especially with the usage of recurrent neural networks. The emergence of RNN (Recurrent Neural Network) structures like LSTM (Long Short Term Memory) has broken records of many language-related tasks [7]. The basic idea of RNN is the output of hidden layers could be used as inputs as well. Variants like LSTM and GRU (Gated Recurrent Unit) [8] provides gates that control the updating of the internal state and output of information. LSTM and GRU mimic human memory and simultaneously alleviate the gradient vanishing problem that exhibits during RNN training.

Deep learning has also been successful on image classification tasks. However, training a image classification network from scratch requires (1) a large dataset and (2) a good architecture design. With only 30,000 images, it's not likely we can train a good neural network. As suggested in Stanford CS231n course [9], transfer learning comes to the rescue. Since convolution layers can represent general abstract features of an image, we can fix convolution layers of some pre-trained network and retrain the fully connected layers (or together with last few convolution layers) for our new task. VGG16 [10] and ResNet-152 [11] are well-known CNN (Convolutional Neural Network) architectures for image classification and the pre-trained weights on ImageNet for those architectures are available. We used those two pre-trained networks in our project.

4 Modeling Approach

4.1 Text-Only Conventional Models

As baselines we have implemented several conventional models for text classification. The basic structures are very similar: we get vector representation of overview texts and use 19 classifiers to predict the genres of the movies. Our approach (using 19 classifiers) is also called "binary-relevance method" in multi-label learning.

For vector representation, we have tried bag-of-words, tfidf [12], and word2vec [5] embeddings. Bag-of-words simply sums the one-hot vector of words; tfidf [12] weighs the vectors by term frequency (how many times it occurs in the overview) divided by document frequency (how many times it occurs in all overviews); word2vec [5] converts words into their lower dimensional representations and we take the mean of all words in the text (the lower dimensional representations are obtained using CBOW or skip grams).

For the classifier part we tried support vector machine and multinomial naive bayes.

4.2 Text-Only RNN

We tried to use RNN to predict the genres based on overview texts. The basic idea of our text-only model is to use RNNs to encode the input text into vectors, and then use a classifier network to predict the genres of the movies. We implemented 3-layer bidirectional GRU or LSTM as our encoders, and our classifier network (Fig. 7) is made by 19 subnetworks, each having 3 fully-connected layers. To get encodings, we stacked the outputs from last cells in forward pass and first cells in backward pass (Fig. 5). The 19 classifiers use these encodings to predict whether the movie belongs to each genre or not. The entire network architecture is shown in Fig. 6.

We trained the network from scratch, the input words were first converted into embeddings. We didn't train the embedding layer, instead we used the pre-trained GloVe[6] (6B.300d) embedding.

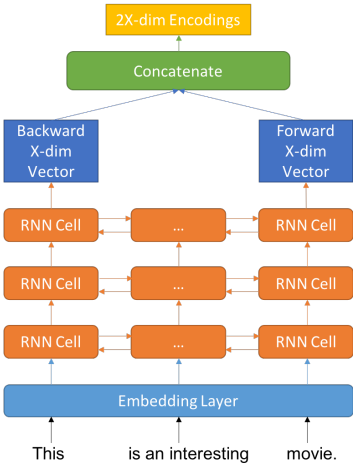

Figure 5: Bidirectional Encoder

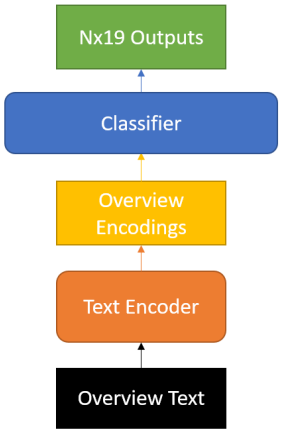

Figure 6: RNN Network

4.3 Poster-Only CNN

We also tried to use the posters to predict the genres. As discussed in the previous section, it is better to use transfer learning. In our project, we tried both VGG16 and ResNet-152. We first got the pre-trained weights of those two models on ImageNet and the architecture of the network is shown as the left branch of Fig. 8. We then threw all fully connected layers of this network (the dotted part) and used the output of the last convolution layer as poster encodings. The encoding was then fed to a similar classifier network (Fig. 7) like in text-only RNN. The whole architecture is shown as the right branch of Fig. 8

For the training, we only trained the last 2-3 convolution layers and fixed the rest of convolution layers with the ImageNet weights since we believe the network trained on ImageNet can also capture the abstract features for images in our task.

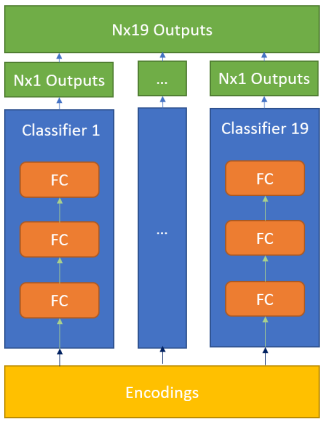

Figure 7: Classifier Network

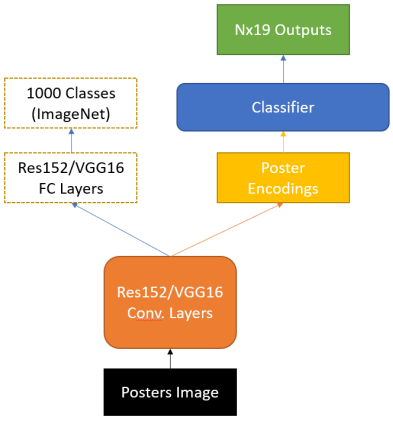

Figure 8: CNN Architecture

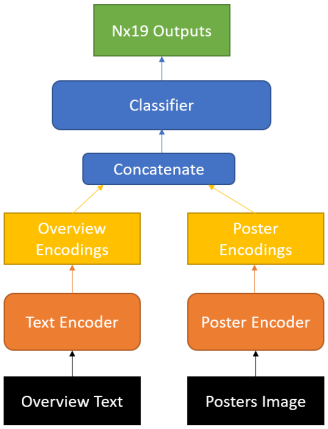

Figure 9: Combined Network

4.4 RNN + CNN Combined

We then combined the text-only RNN network with poster-only CNN network. We stacked the encodings from both networks together and then fed the encodings to 19 3-layer-fully-connected feed forward networks. The architecture is shown in Fig. 9.

Due to limitation of our GPUs, training the entire network (joint training) is very hard because it requires huge GPU RAM. Instead we **fixed the encoders and only trained the classifier**.

This combined network is our final model and achives the best results in our validation and test datasets.

5 Project Trajectory

According to our EDA, we found that the dataset is unbalanced in genre distribution. Therefore, we decided to use F-measure instead of accuracy. In addition, both texts and images seem to be useful features for classification task from EDA, so we planed to train classifiers for these individuals' features and their combined features.

For overview texts, while evaluating the conventional models, we found that different feature extraction methods affect the results, and therefore, it is natural to consider using deep neural network to learn the right representation of text data. Since text is a kind of time-series data with sequence dependencies, we applied RNNs.

For posters, we trained CNNs for the classification task. However, training good CNNs from scratch is difficult, so we used transfer learning. Our poster-only models are not as good as the text-only models and it makes sense since overview texts intuitively contain more information than posters and predicting movies' genres with only posters is very hard even for human beings.

Although poster-only models don't perform well, we believe posters contain some information that does not exist in overview texts. Therefore, we combined the poster-only network and text-only network by stacking the encodings from both networks together, and fed to another classification network. This combined model evetually achieves the best result.

6 Results and Interpretation

We report the results of both our baseline models (SVM and Naive Bayes) and deep learning models in the table below. Validation set (val) is used to select model and test set (test) is used to approximate our models' performance on future data.

Model		F_1 (val)	Accuracy (val)	F_1 (test)	Accuracy (test)
(Baseline) Conventional Models	SVM (bag-of-words)	0.50	0.93	0.49	0.93
	NB (bag-of-words)	0.42	0.91	0.42	0.91
	SVM (tfidf)	0.33	0.93	0.32	0.93
	NB (tfidf)	0.14	0.93	0.14	0.93
	SVM (word2vec)	0.28	0.93	0.28	0.93
Deep Learning Models	TextOnly (GRU)	0.51	0.93	0.50	0.93
	TextOnly (LSTM)	0.53	0.94	0.51	0.93
	PosterOnly (ResNet-152)	0.45	0.93	0.43	0.93
	PosterOnly (VGG16)	0.42	0.92	0.40	0.92
	Combined (LSTM+ResNet-152)	**0.54**	**0.95**	**0.52**	**0.94**

Table 1: Results on Validation Set and Test Set

From Table 1, we have the following observations:

1. The combined model using both plot descriptions and posters provides the best result in both validation and test set, slightly higher than RNN models with only text embedding features. **All text-included deep learning models beat our baseline models (conventional models)**.

2. For the conventional machine learning models, using bag-of-words representation for text features has better predictability and it gives a result very close to our best model. The reason might be that TMDB plot discription is generally short, so simply identify certain words could be able to distinguish the genres. While using tfidf or embeddings is likely to deviate from this simple feature.

3. Models with only poster information didn't perform well. It is very hard even for human to accurately predict simply based on posters (comparing to replying on overview texts). But combining it with text features could slightly improve our model's performance.

Genre	F_1	Precision	Recall	Accuracy	AUC
Action	50.31%	61.54%	42.55%	94.73%	86.99%
Adventure	36.64%	66.67%	25.26%	97.23%	87.94%
Animation	45.61%	71.23%	33.55%	95.87%	89.01%
Comedy	53.68%	61.69%	47.51%	85.73%	83.52%
Crime	33.92%	76.32%	21.80%	96.23%	83.47%
Documentary	56.14%	64.36%	49.79%	87.87%	85.37%
Drama	60.30%	60.21%	60.40%	74.77%	79.85%
Horror	59.64%	80.56%	47.35%	94.77%	91.53%
Music	31.65%	73.33%	20.18%	96.83%	78.04%
Mystery	40.00%	95.45%	25.30%	97.90%	81.92%
Romance	47.09%	68.99%	35.74%	93.33%	82.59%
TV Movie	31.82%	70.00%	20.59%	99.00%	75.12%
Thriller	46.09%	64.84%	35.76%	90.80%	86.01%

Table 2: Statistics for Some Genres of Combined Network (Shown in Table 1) on Test Set

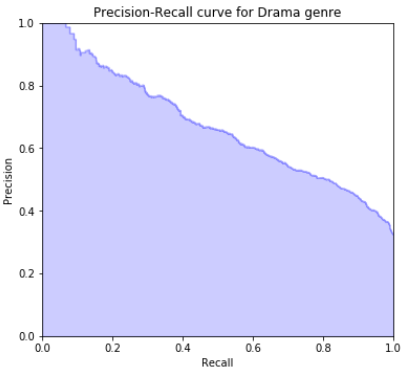

Figure 10: *Drama* PR curve

From Table 2, we have several observations:

1. All genres have reasonable F_1-score and AUC. Genres with larger sample sizes have better evaluation metrics (also see Fig. 1). The classificaiton model is influenced by unbalanced data set to some degree. The discrepency between accuracy and F_1/AUC is a consequence of such unbalance.

2. TV Movie and Music have the worst AUC and F_1 score among all genres. Both of them have limited sample size and are not closely correlated with other genres. There might not be enough training data for our model to specify their unique features.

3. Human error might exist in ground truth labels since it is relatively difficult to distinguish certain correlated topics, such as horror and thriller. Therefore, considering these human error, our evaluation statistics are within reasonable range for all genres.

We also plot an example precision-recall (PR) curve for genre *Drama* as shown in Fig. 10 for our combined network on test set.

7 Conclusions

In this project, we implemented several machine learning models to classify genres of movies using their plot descriptions and posters. We sucessfully combined these features together and made a machine learning pipeline to automatically tag the genres. The combined model with encodings from text-only RNN and poster-only CNN provided the best result among all other models. However, our best model did not outperformed the best baseline model a lot. Several proposed improvements are discussed in the next section.

8 Possible Future Work

One important characteristic of our dataset is it's multi-labled. We have tried applying binary relevance method to deal with the problem (i.e. train 19 classifier each responsible for one class). However, it could be improved in several ways:

- The relation among genres could be used as priors. For example, many thriller movies might also be in horror genre. Our method didn't consider such priors.
- For each genre, the dataset is very unbalanced. If we want to maximize F_1-score, we may consider using weighted loss functions or undersampling.

In addition, more combinations of hyper-parameters can be tried. We can also consider changing the training process. For example, unfix some layers of CNN or RNN during the training of the combined model. We might also consider using different embeddings (e.g. GloVe pretrained on larger dataset). In addition, We might also consider fine-tuning the embedding layer.

References

[1] Follows, S. (2017). How many films are released each year? https://stephenfollows.com/how-many-films-are-released-each-year/ accessed Apr. 21, 2018.

[2] Sorower, M. S. (2010). A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis, 18*

[3] Frnkranz, J., Hllermeier, E., Menca, E. L., & Brinker, K. (2008). Multilabel classification via calibrated label ranking. *Machine learning*, 73(2), 133-153.

[4] Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. *Machine learning*, 85(3), 333.

[5] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111-3119.

[6] Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532-1543.

[7] Sak, H., Senior, A., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Fifteenth annual conference of the international speech communication association*.

[8] Cho, K., Van Merrinboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*.

[9] Fei-Fei, L., Johnson, J., & Yeung, S. (2018). CS231n: Convolutional Neural Networksfor Visual Recognition. http://cs231n.github.io/transfer-learning/ accessed Apr. 21, 2018.

[10] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*.

[11] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778.

[12] Leskovec, J., Rajaraman, A., & Ullman, J. D. (2014). *Mining of massive datasets*. Cambridge university press, pp. 1-17.