CS230

Classification of Movie Posters to Movie Genres

Samuel Sung
Environmental Engineering
Stanford University
ssung10@stanford.edu

Rahul Chokshi
Environmental Engineering
Stanford University
rchokshi@stanford.edu

Abstract

The poster of a movie is an essential tool in the film industry and a well-designed poster can be pivotal towards the success of a movie. To ensure that a movie poster is designed to attract the correct audience, we built a deep learning model that identifies the movie genre conveyed by a movie poster. To find an optimum model, we implemented customised versions of three standard deep learning architectures for image classification: ResNet-50, VGG-16, and DenseNet-169. DenseNet-169 gave us the best results, outperforming the other two models and models designed in previous attempts at the problem. Although the results were positive and a step forward from previous studies, there still remains room for improvement. We forecast that this improvement could come through a number of techniques, the most probable being, the use of an object identifier algorithm such as YOLO, collection of better data, and application of other convolutional neural network (CNN) architectures to the problem.

1 Introduction

Genre classification has been a deeply explored subject in deep learning with good reason. The genre of a piece of art encodes a great deal of information about the piece within a single word. It is a concise and effective way to highlight the similarities and dissimilarities between different works. When it comes to movies, the genre is often the deciding factor for a viewer to make a selection between various options.

In the film-making industry, the poster of a movie has an incredibly important role. Traditionally, the poster has delivered the first introduction of a movie to viewers - through displays at cinema-houses, within newspaper advertisements, or as DVD-covers. More recently, with the advance of digital streaming platforms, the role of the poster has increased in importance. For viewers browsing through most digital platforms, the visuals in the poster are what make a certain movie distinguishable within the abundance of content.

The motivation behind our project is to provide a tool that ensures that a movie poster is designed to stimulate the correct audience. This is done by identifying the genre information that the visuals in the poster convey. A well-designed poster is able to convey the genre of a movie to a human observer, who has no prior knowledge of the movie, at a glance. Therefore, we expect a movie poster to have visual attributes that can be associated with different genres by a deep learning algorithm. With this in mind, we designed a model that takes the image of a movie poster as an input, passes it through a Convolutional Neural Network (CNN), and outputs the different movie genres that the poster falls within. Such a model would be very useful for the film-making industry to optimize the design of their posters. It also has potential applications within digital streaming platforms, such as expedited sorting of their content library and provision of user recommendations.

2 Related work

There have been numerous studies in genre classification using deep learning. The majority of genre classification models have focused on detecting musical genres from audio recordings. Several of these have used a CNN for the task by using a spectrogram of the audio file as an input [1], [2]. Fewer studies have focused on genre classification of images. Some examples of such studies have aimed to predict painting genres [3], musical genres using artwork [4], and book genres using book covers [5]. Similarly, few studies have approached the problem of movie genre classification within the broader genre classification theme. Within this, most efforts have focused on genre classification based on the movie trailer [6] or the plot synopsis [7]. Genre classification based on movie posters has been approached less frequently. We suspect that this is because it is a multi-label classification problem, which makes it more difficult than a binary classifier, and it is limited to information from a single image.

One of the earliest attempts at movie poster classification is described in [8]. The authors used Naïve-Bayes, C4.5 Decision Tree, and k-Nearest Neighbors as base classifiers with a RAKEL ensemble method. The low-level features used for the classification were extracted from the movie posters using the Dominant Colors, GIST and Classemes methods. This approach carries the advantage of being computationally simpler than a Neural Network but is limited to the assessment of only low level features. More relevant to our project are the methods employed by [9] and [10]. The authors in [9] tested several methods for the classification including k-Nearest Neighbors, a modified ResNet-34 CNN, and a CNN with a custom architecture. Although their assessment was extensive, the dataset used for the study was heavily skewed towards certain genres (Eg. Drama, Romance). The loss function used in training their Neural Networks was not weighted to compensate for the under-representation of other genres, which could have led to errors in their predictions. The authors in [10] designed a complex architecture that combined a CNN and a YOLO object identifier to do the classification. Despite it being a multi-label classification problem, the output from the final fully connected layer in their model was put through a softmax activation. To obtain a multi-label output following that, each label within the output vector was given a different decision threshold. The threshold for each genre was chosen through a separate algorithm that minimized the distance between the output vector and the ground truth vector during training. This design could have possibly led to a drop in performance.

From the results obtained for each study, it appears that the best performance was achieved by the models in [9]. Therefore, to the best of our knowledge, this is the state-of-the-art model for this problem.

3 Dataset and Features

The dataset used for training the model consists of two components. The first is a set of movie posters and the other, a set of their genre labels. This dataset was produced using a text file obtained from Kaggle [11], which contains a database of movie information including movie titles, genres, IMdB scores, and URLs to images of movie posters. The original database contains 40109 movie titles. Upon removal of titles that don't have any genres listed and titles that have a missing or broken URL to the poster image, we were left with 36898 titles. We scraped the poster images from the URLs for these and obtained the RGB pixel data for each. The pixel data of a single poster were of size 268 x 182 x 3. This was resized to 256 x 256 x 3 for all the images. The corresponding genre labels were converted to multi-hot vectors as illustrated in Figure 1. (Note: The original database contains more genres than displayed in the figure. The figure is mainly for expository purposes.)

The database contains a total of 28 genres. Some of these genres contain relatively few examples. We decided to exclude these genres from the training because they would have led to a very imbalanced dataset. We only included genres that had 4500 examples or more. This criteria narrowed our list of genres to 6. These are Action, Comedy, Crime, Drama, Romance and Thriller. For examples that do not fall into any of these 6 genres, an additional label 'Other' was included. Therefore, a multi-hot vector of the form shown in Figure 1 was condensed to a multi-hot vector with 7 entries. The data was not normalized because we planned on using pre-built Keras models, which come along with an input pre-processing function. The pixel data for all the examples were stored as a numpy array of

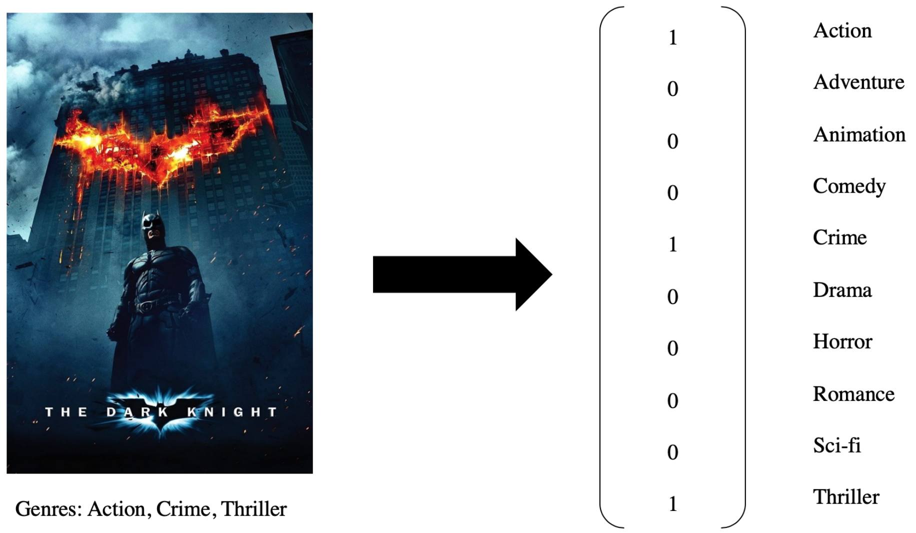


Figure 1: Movie genres labeled as multi-hot vectors

size 36898 x 256 x 256 x 3 and the label data were stored as a numpy array of size 36898 x 7. This comprises our dataset.

The size of our dataset is relatively small compared to the datasets of modern deep learning models. Hence, we decided to split the dataset into an approximately 80:10:10 ratio for the training set, development set, and test set respectively. This gave us a break up of 29887 training examples, 3321 validation examples, 3690 test examples.

4 Methods

The dataset was trained on modified versions of three different model architectures: ResNet-50, VGG-16, and DenseNet-169. The architectures used are described below:

4.1 Modified ResNet-50

We implemented a standard ResNet-50 architecture with some modifications. The final fully-connected layer of 1000 units was replaced by 3 sequential fully-connected layers of 1024, 128, and 7 units respectively. These layers have ReLU, ReLU, and sigmoid activations respectively. Unlike the standard ResNet-50 architecture, our model has a sigmoid activation to the final layer instead of a softmax activation because we are dealing with a multi-label classification problem. The entire model consists of 25,817,991 parameters, out of which, 25,764,871 parameter were trainable.

4.2 Modified VGG-16

Similarly, we implemented a standard VGG-16 architecture with modifications. The final fully-connected layer of 1000 units was once again replaced by 3 sequential fully-connected layers of 1024, 128, and 7 units with ReLU, ReLU, and sigmoid activations respectively. The entire model consists of 15,372,103 parameters, all of which were trainable.

4.3 Modified DenseNet-169

Finally, we implemented a standard DenseNet-169 architecture with similar modifications. The final fully-connected layer of 1000 units was once again replaced by 3 sequential fully-connected layers of

1024, 128, and 7 units with ReLU, ReLU, and sigmoid activations respectively. The entire model consists of 14,479,943 parameters, out of which, 14,321,543 were trainable.

4.4 Loss Function

We used the weighted sum of binary cross-entropy losses for each individual genre as the loss function for training. The genre weights are inversely proportional to the number of occurrences of that genre in the dataset. Larger weights are assigned to under-represented genres to balance the effect of all the genres on the total loss. The loss function is described by the equation below:

$$L = \sum_{n=1}^{7} W_n(y_n \cdot log(\hat{y_n}) + (1 - y_n) \cdot log(1 - \hat{y_n})$$

W_n is the genre weight of genre n. y_n is the ground truth value, and $\hat{y_n}$ is the predicted value of genre n. The average value of the loss function over all the examples in a mini-batch was used as the cost function for training.

5 Experiments/Results/Discussion

Our selection of hyper-parameters included the hyper-parameters within an Adam optimizer, mini-batch size, number of epochs, and the decision thresholds for each model. We chose to use an Adam optimizer with all our models because it optimizes the learning rates used in gradient descent. Default hyper-parameters within the Adam optimizer were used because of their robustness in transferring between various datasets.

For the modified ResNet50, we chose a mini-batch size of 32, trained the model for 20 epochs, and chose a decision threshold of 0.3. This combination was found through a heuristic approach and was best suited for accurate results within an acceptable computation time. At the end of training, the model achieved a training accuracy of 0.79 and validation accuracy of 0.79. The modified VGG-16 model had a mini-batch size of 32, was trained for 10 epochs, and has a decision threshold of 0.3. It achieved a training accuracy of 0.78 and a validation accuracy of 0.78. The modified DenseNet169 model had a mini-batch size of 32, was trained for 10 epochs, and has a decision threshold of 0.35. It achieved a training accuracy of 0.79 and a validation accuracy of 0.79. The similarity between training and validation accuracies between all three models indicates low variance in the algorithms.

Since most entries in the multi-hot vector of genre labels are zero for most examples, accuracy is not a good metric to assess the performance of the model. A model that predicts that all movies fall in none of the genre options would also achieve a reasonably high accuracy. Therefore, we decided to assess the performance of the models based on the F1 scores and the AUC of the Receiver Operating Characteristic (ROC) curve. The F1 score is defined as:

$$F1\,score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

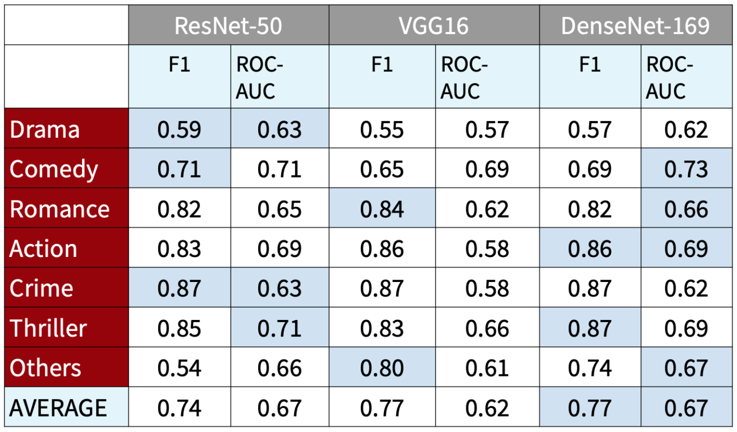
$$Precision = \frac{True\,positives}{True\,positives + False\,positives}; Recall = \frac{True\,positives}{True\,positives + False\,negatives}$$

The ROC curve is the the plot of True Positive Rate (TPR) vs. the False Positive Rate (FPR) for different decision thresholds. The TPR is the same as the Recall and the FPR is defined as:

$$False\,Positive\,Rate = \frac{False\,positives}{True\,positives + False\,negatives}$$

The area under the ROC curve gives the AUC for a classification problem. These metrics were calculated for each genre within each model. The results obtained are shown in Figure 2.

The F1 score for all three models outperformed the F1 score for the state-of-the-art model in [9]. DenseNet-169 appears to be the best performing model because it achieves the highest F1 score



	ResNet-50		VGG16		DenseNet-169	
	F1	ROC-AUC	F1	ROC-AUC	F1	ROC-AUC
Drama	0.59	0.63	0.55	0.57	0.57	0.62
Comedy	0.71	0.71	0.65	0.69	0.69	0.73
Romance	0.82	0.65	0.84	0.62	0.82	0.66
Action	0.83	0.69	0.86	0.58	0.86	0.69
Crime	0.87	0.63	0.87	0.58	0.87	0.62
Thriller	0.85	0.71	0.83	0.66	0.87	0.69
Others	0.54	0.66	0.80	0.61	0.74	0.67
AVERAGE	0.74	0.67	0.77	0.62	0.77	0.67

Figure 2: F1 scores and AUC for the three models

and AUC of the three models. An F1 score of 0.77 indicates that the DenseNet-169 model correctly predicted the true genres of several posters and an AUC of 0.67 indicates that it is capable of distinguishing between different genres and does not predict randomly. The ROC curves for the DenseNet-169 model are shown in Figure 3.

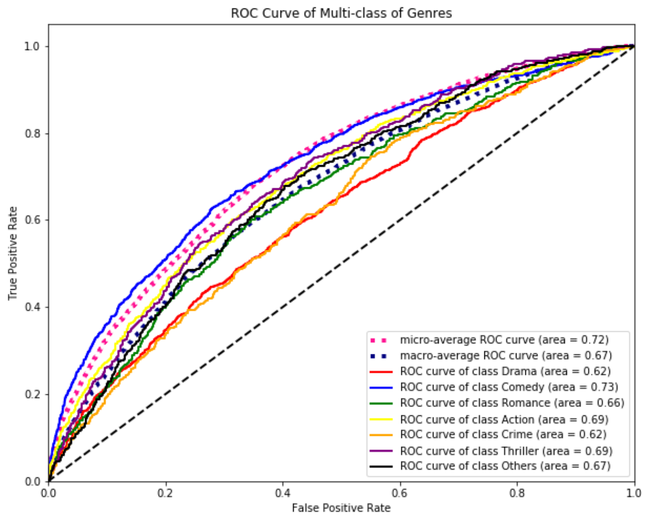


Figure 3: ROC curves for DenseNet169

6 Conclusion/Future Work

Movie genre classification based on movie posters has many practical applications and our models got good results at this task. Specifically, the modified DenseNet-169 model was the most effective at this classification, observed through the high F1 score and AUC achieved. We believe that there is still room for improvement. Future work should be focused on including a YOLO algorithm in the model to identify distinct objects within the posters that can be connected to certain genres. A more balanced dataset would also improve the performance of the model. Finally, there remain several high performing CNN architectures such as Inception, Xception, and MobileNet that can be implemented on the problem and may result in better performance.

7 Contributions

Rahul was responsible for identifying and understanding past attempts at the problem, identifying the correct metrics for the problem, analysing the performance based on the metrics, and production of the report. Sam was responsible for extracting data, pre-processing the data, implementation of the models, calculating metrics for each of the models, and production of the poster. Both collectively contributed to ideation of the various components of the project.

References

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

[2] O' Bierne, A. & Zamora, A. (2018). Music Genre Classification. *Stanford CS 230: Deep learning*.

[3] Golara, S. (2019). Painting Genre Classification. *Stanford CS 230: Deep Learning*.

[4] Koenig, C. (2019). Classifying Album Genres using Album Artwork. *Stanford CS 230: Deep Learning*.

[5] Iwana, B. K., Rizvi, S. T. R., Ahmed, S., Dengel, A., & Uchida, S. (2016). Judging a Book by its Cover. *arXiv preprint* arXiv:1610.09204.

[6] Simões, G. S., Wehrmann, J., Barros, R. C., & Ruiz, D. D. (2016). Movie genre classification with convolutional neural networks. *In 2016 International Joint Conference on Neural Networks (IJCNN)* (pp. 259-266). IEEE.

[7] Battu, V., Batchu, V., Reddy, R. R., Reddy, M. K., & Mamidi, R. (2018). Predicting the Genre and Rating of a Movie Based on its Synopsis. *In Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*.

[8] Ivasic-Kos, M., Pobar, M., & Ipsic, I. (2014). Automatic movie posters classification into genres. *In International Conference on ICT Innovations* (pp. 319-328). Springer, Cham.

[9] Barney, G., & Kaya, K. (2019). Predicting Genre from Movie Posters. *Stanford CS 229: Machine Learning*

[10] Chu, W. T., & Guo, H. J. (2017). Movie genre classification based on poster images with deep neural networks. *In Proceedings of the Workshop on Multimodal Understanding of Social, Affective and Subjective Attributes* (pp. 39-45).

[11] Neha. (2018). Movie Genre from its Poster. *https://www.kaggle.com/neha1703/movie-genre-from-its-poster*.