Final Report: Movie Genre Classification of Movie Posters
By Pratik Patel and Harsh Patel

Introduction:

Throughout our life we continue to hear the famous idiom of *"Don't judge a book by its cover"* , which in the metaphorical sense means not to prejudge the worth or value of something just alone from it's outward appearance. Even the literal meaning is taught to use when we choose which book to read when we were younger. While we may not be able to learn everything, a person can learn alot just from a cover alone. Just by looking at the cover of book people can learn what the book is mostly about. In the world of films, film posters are an important part of aesthetically appealing to people as well as allowing people to understand a few things about what the movie is about such as the genre of the film. While a human can easily distinguish a movie poster between a horror film from a comedy film, we want a machine to be able to complete the task of classifying movie posters by genres.

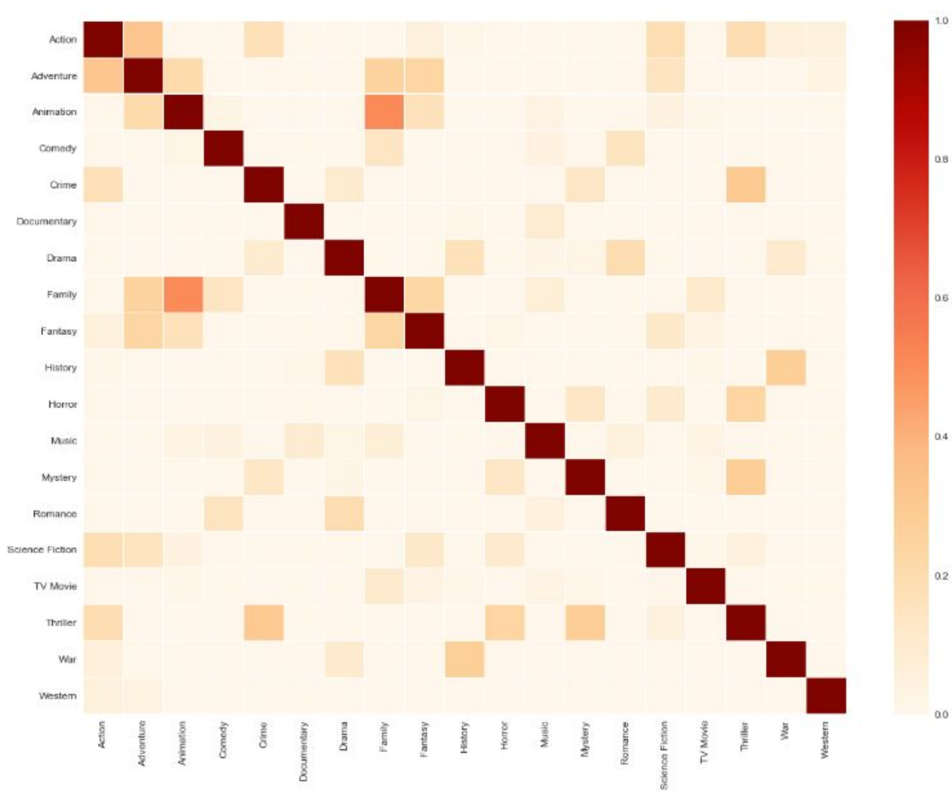
We first drew inspiration of this project due to how I(Pratik) am an advent moviegoer and loves watching movies to the point where I keep track of the movies I have seen on an account on IMDB. The thought came from how there lately has been a trend amongst film posters for different genre movies.The idea came from a few similar articles showing how many film posters are very similar and basically each genre now has its own cliche style for a film poster that companies just match(as seen below). Film posters are a key element in promoting, advertising, and attracting people to watch the movie. A typical film poster today, shows the

popular celebrities in the movie as well as is able to convey the genre or genres of the film without expliciting saying so through the colors and "mood" of the poster. For example, typical comedy films use bright colors such as yellow or a white background while an action movie will have these more darker tones like black. Another more difficult task for a machine is how a typical science-fiction or adventure movies have a man looking into the distance or how in romance movies there is always a man and woman standing back to back.

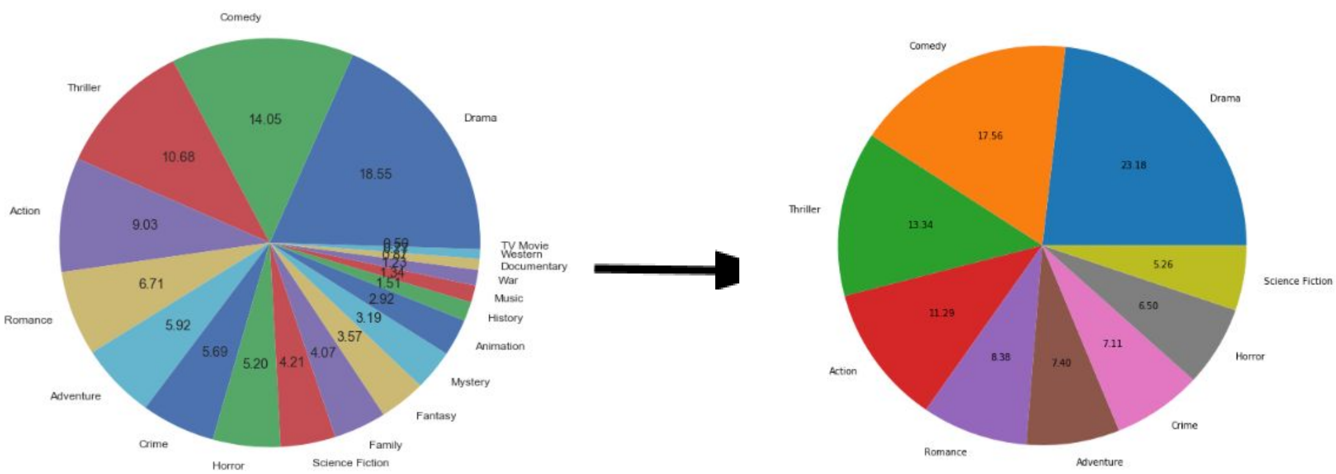
With this trend in mind we saw some research papers attempting to accomplish the same task with some using black and white images and others creating a model but a couple of years ago when movie posters were not that impactful. Therefore, we decided to attempt to try to predict movie genres just based off the film poster that are colored and for movies that are more new.

<u>Methodology:</u>

The initial step was to obtain 10,000 movies with color images as well as the genres of the movie. We also wanted the movies to be more popular and current movies. After getting approved and receiving a TMDb API key, we were first able to obtain the unique TMDb id for nearly 11,000 of the most popular movies on their database. Once we knew the TMD id we used the API, to access each individual movie which was formatted as a JSON file which allowed us to easily access all the information we wanted such as title, IMDb id, the pathway to the poster and the genres that was all stored into a csv file. The process took some time as we could only retrieve 40 movies per 10 seconds. Using pandas to open the csv file, we then performed some exploratory data analysis on the data. TMDb uses 19 genres to classify a movie's genre being Action, Adventure, Animation,Comedy, Crime, Documentary,Drama, Family, Fantasy,History, Horror, Music, Mystery, Romance, Science Fiction,TV Movie, Thriller, War, and Western. From the EDA, we saw that two major insights about the genres. The first insight was



determined from a correlation mapping of each genre with one another (the heatmap) , which from we learned that many of the genres correlate such as History and War or Action and Adventure. The second insight was from the pie chart in which we saw that nearly 10 genres of the 19 only made up 10% of the entire dataset (only 1,000 entries). With these two insights, we decided that it was necessary to drop the genres that only contained a few movies to help increase the accuracy of the convolutional neural network and doing so would not affect the data greatly as many of the genres dropped had high correlational values with ones kept. In the end, we created our final dataset by one hot encoding the genres we kept, Action, Adventure, Comedy, Crime, Drama, Horror, Romance, Science Fiction, and Thriller, and removed all blank rows (did not have any of the genres kept) to obtain a dataset with 10,590 movies. Using this final dataset, we then scraped the TMDb website for the film posters using the poster path to obtain color images of 10,590 movies with image size of 500 by 750.

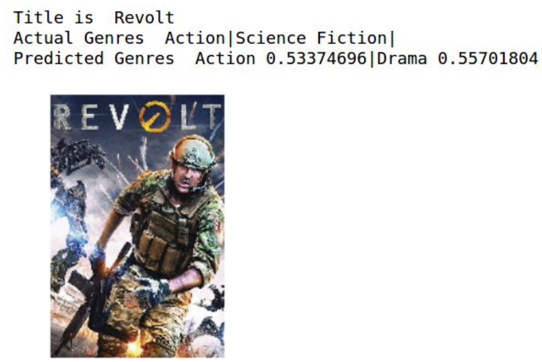


The next step was to preprocess the images which we used generic techniques to clean images like resize all pictures to a specific image size in our case 154 by 154, and normalize RGB values. We used Python Imaging Library (PIL) to extract and create a matrix of each pixel's channel value to create a 3-dimensional matrix of size (154,154,3) which is our input variable and our output variable is an nine element array containing binary values for each genre. Using research

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 150, 150, 64)      4864
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 148, 148, 64)      36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 74, 74, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 74, 74, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 72, 72, 32)        18464
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 70, 70, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 35, 35, 32)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 35, 35, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 39200)             0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                2508864
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 9)                 585
=================================================================
Total params: 2,578,953
Trainable params: 2,578,953
Non-trainable params: 0
_________________________________________________________________
```

papers as a guidance, we created a convolutional neural network(CNN) that initially applies 64 filters the size of 5 by 5 with 3 more convolutional layers and 2 Denses layers were the final dense layer used sigmoid activation and the model was compiled with binary cross entropy. In the end, we created a model with 2,578,953 parameters.

<u>Results:</u>

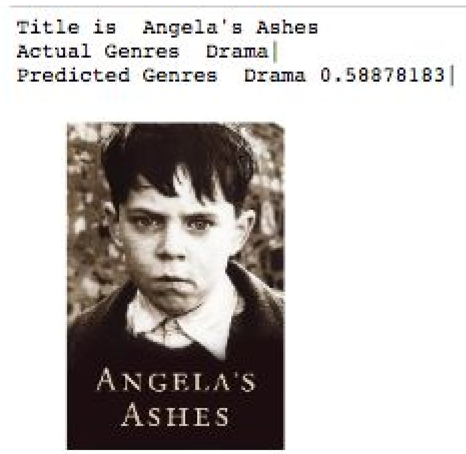


During our experimental phase we tried running a simple model with 500 pictures and 100 pictures for testing. Afterwards we did eventually settle on the model shown above. Next, we ran larger scales of the model with 5000 points of training data and 1000 points of test data. Initially we ran it with only 3 epochs since the algorithm takes a long time to run. It resulted in a training accuracy of 72% and a loss of 0.5. We then later increased the epochs to 10 and used the entire dataset with a 2:1 training:testing split. We obtained a training accuracy of 83.41% and a loss of 0.3602 after its final epoch. Which shows us that more epochs and more data was increasing the overall accuracy and ideally we would have like to run it with 50 epochs if possible.

Using the model we then predicted the values for the testing data and used a normal threshold value 0.5 to determine if the model actually predicted a genre or not. When we looked at a couple of the predicted and actual genres of a movie we saw that overall, it did pretty good with drama, comedy, and thriller movies. After running model.evaluate we were able to get a test score of 0.5 and a test accuracy of 78% which was slightly less than the training accuracy of 83% that we received.

From our research we found that others had better results from running black and white images into the dataset rather than colored images. In our attempt to do this we ran 10 epochs on the entire dataset with the 2:1 training:testing split similar to our previous experiment. After 10 epochs, it resulted in a training accuracy of 82% and a loss of 0.3928. This was very similar to our previous results. It is likely that if the epoch number kept increasing that we would eventually see one beat the other out in terms of accuracy. When running model.evaluate we obtained a test score of 0.5 and a test accuracy of 0.518 and an testing accuracy of 77.5%. Again, this was similar to our colored results. So overall, we did not see a performance increase from using black and white images during our experiment.

The trained model predicted the genre the had the highest score given. The figures shown above are examples of the algorithm at work. You can see how our model would predict when given the following test images. Our model is flexible and is able to predict either one or multiple genres using a threshold value or by getting the top genres.



Conclusion:

Overall, we were able to predict couple distinct genres from our model specifically being Drama, Thriller, and Comedy. We learned that being able to predict genres just from a movie poster like a human is not an impossible task, yet there still is a long way to improve the model and so many ways to expand this project. Generally, we definitely learned that while the task is possible as whenever we increased the epochs or even the size of the training data the accuracy increased. However, the overall problem of this project was managing our time as just creating and fitting our model took a great deal of time. Some of the tasks we wished to accomplish but could not that dealt with the improvement of the model itself was using more of the image itself instead of reducing it to 154 pixels which then would allow us to create a more deeper CNN. This reduction ended up compressing the pictures to about 10% of the original size, and thus we lost out on 10 times the possible data points for each image with this method. In conclusion, we feel as if time was not a constraint we could create a higher accuracy model just by adding more epochs and training data as accuracy continued to increase and loss kept decreasing just at a slow rate. While we weren't able to fully judge a book by its cover, we were able to take a step closer in doing so.

References:

http://thechive.com/2016/12/12/movie-posters-that-look-oddly-similar-17-hq-photos/
https://gizmodo.com/5857101/have-you-ever-noticed-that-all-movie-posters-look-the-same
https://www.cs.ccu.edu.tw/~wtchu/papers/2017MUSA-chu.pdf
https://pdfs.semanticscholar.org/a47f/d31a18b3b1b5d02815a9220ab806ceb58edd.pdf