# Film Classification by Trailer Features

Edmund Helmer and Qinghui Ji
December 14, 2012

## 1. Introduction

*The Matrix*, a sci-fi film released in 1999, was famous for telling viewers in its trailer that: "Unfortunately, no one can be told what the Matrix is. You have to see it for yourself." And film-goers did, resulting in over $460 Million worldwide box office gross, four Academy Award wins, and an 87% critic approval rate (according to Rotten Tomatoes). The decision to see any film is based on many factors: cast, critical opinion, recommendations of our friends; but often it's simply because we like the trailer. So we ask the question, what information can a machine extract from trailers?

For this paper, we have collected a set of trailers (312 with video, 100 with subtitles), extracted features from the video feeds and the subtitle texts, and attempted to classify the genre and MPAA rating of each film.

## 2. Data

### 2a. Video Features

The potential feature space from a trailer is massive, including an RGB tuple for every pixel for every frame of every trailer, as well as potential object detection. In order to prune the space into a more manageable set, we used three main categories of video features: (1) Mean Frames, (2) Scene Variation, (3) Face Recognition. To obtain the trailers, we used Python to scrape the-numbers.com, film site, to download all trailers that the site hosts (312); and the Python module OpenCV provided the tools to complete the video analysis.

(1) To create what we call "Mean Frames," for each pixel location, we averaged all values of that pixel over the frames of the trailers, and in effect created a single "blurred" image for each trailer. Fig. 1 shows two example Mean Frames: *Finding Nemo* and the recent release, *Lincoln*. While image blurring is a standard technique used for background detection,[1] as far as we know, the creation of mean frames is an original concept in visual analysis, though similar to key-frame detection (described in more detail later).

Fig. 1: Mean Frames of *Finding Nemo*, *Lincoln*

From the Mean Frames, we extracted the RGB histograms, which represent the distribution of red, green, and blue used in the trailer, as shown in Fig. 2. Each color's vector of intensities were then recorded as features.

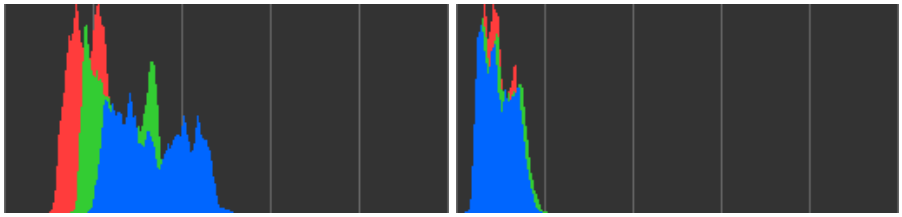

Fig. 2: Mean Histograms *Finding Nemo*, *Lincoln*

(2) Key frame generation is a standard task in video processing, which splits any video file into its separate scenes, and key frames are the splitting points. One method used to create key frames is to measure the change in each pixel's RGB values between each consecutive frames, and returns the frames following the highest changes.[2] We used this idea to measure the number of scene changes by keeping an average of the frame changes over each trailer, with the idea that some genres or MPAA ratings may be described by faster or slower cutting of scenes.

(3) Finally, we hypothesized that some genres and MPAA ratings would feature more or fewer people in their trailers, and that face recognition algorithms would provide a reasonable proxy for this. Fig. 3 demonstrates face recognition from one trailer of *Harry Potter and the Deathly Hallows: Part 2*, one example of the algorithm in process.

Fig. 3: Face recognition from *Harry Potter and the Deathly Hallows: Part 2*

### 2b. Text Features

Words spoken in a trailer are text files downloaded from the Youtube closed-caption database and was read into a matrix, similar to the 'spam' dataset. Each row is one sample and each column is a word. Due to the scarcity of English subtitle resource online, we downloaded Spanish versions. First we deleted all the words that are less than two characters. Then using TreeTagger packages from University of Stuttgart, we scrubbed all the words: words like articles ('el'), preposition ('y') and proper noun('Michael') were removed; noun and verbs were then adjusted to their standard forms ('acercara', 'acercarnos' and 'acercarte' to 'acercar'). At the end, we reduced the number of words from 4739 to 3563.

## 3. Models and Evaluation

### 3.a Video Features

The features extracted from the video provided some separation without any complex analysis, as shown in Fig. 4, a mapping of Genre and MPAA Rating by percent of time in the trailer faces appeared ("FaceTime"), and the average brightness of the trailer ("Brightness").

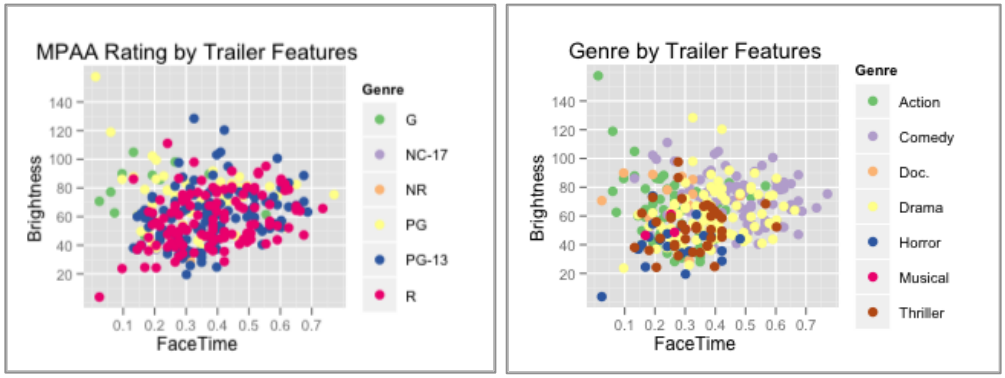


Fig. 4: MPAA Rating and Genre mapped to FaceTime and Brightness.

The most easily seen results are that: G rated films tend to skew left, which turns out to be because they heavily involve animated features with non-human characters; Drama and Comedy feature the most human content; Horror films use the least amount of brightness in the trailers.

To attempt classification, we applied 3 different main classification algorithms: naïve Bayes, Multi-Class Support Vector Machine, and Random Forest. Although the original feature space included the entire histogram of RGB values stemming from each film's Mean Frame, we found that results were improved (even over tuning) by only using the mean values of each color as features, and that using the entire vector of information for each color led to overfitting. The Support Vector machine was tested over linear, radial basis, sigmoid, and polynomial kernels, as well as tuned by γ and cost. The Random Forest was only tuned to include enough trees to reach convergence, and naïve Bayes was run without tuning.

Shown below, the Multi-Class Support Vector Machine has the highest accuracy in predicting genre, and Random Forest has the highest accuracy in predicting MPAA Rating.

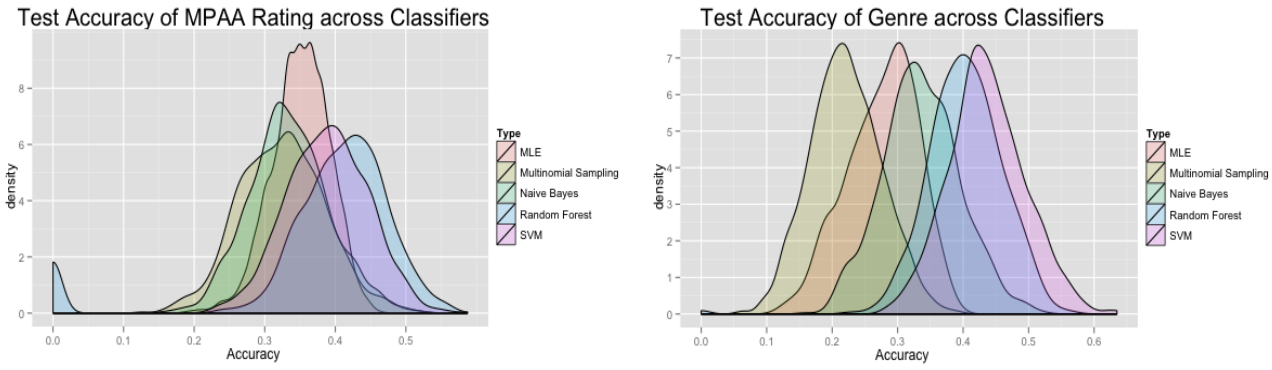


Fig. 5: Classification Accuracy Histograms of Test Accuracy (1000 iterations, 80% training) on MPAA and Genre

Table 1: Mean Accuracy Rate across Classifiers Predicting MPAA Rating by Video Features

| Naive Bayes | SVM | Random Forest | MLE | Multinomial |
|---|---|---|---|---|
| 0.3341 | 0.3887* | 0.3897* | 0.351 | 0.3244 |

* - p-value < $10^{-20}$ comparing difference in mean accuracy to the MLE classification.

Table 2: Mean Accuracy Rate across Classifiers Predicting MPAA Rating by Video Features

| Naive Bayes | SVM | Random Forest | MLE | Multinomial |
|---|---|---|---|---|
| 0.3341 | 0.3887* | 0.3897* | 0.351 | 0.3244 |

* - p-value < $10^{-20}$ comparing difference in mean accuracy to the MLE classification.

The classification techniques were compared to sampling randomly from a multinomial distribution (as parameterized by the training set distributions), and compared to an MLE estimate (which is selecting the mode of the training tags). The algorithms were run selecting an 80%/20% training/testing split over 1000 iterations, and all classification algorithms showed statistical significance, except naïve Bayes in predicting MPAA.

Table 3: Mean Accuracy Rate across Classifiers Predicting Genre by Text Features

| Naive Bayes | SVM | Random Forest | MLE | Multinomial |
|---|---|---|---|---|
| 0.3359* | 0.4366* | 0.4032* | 0.2760 | 0.2208 |

* - p-value < $10^{-20}$ comparing difference in mean accuracy to the MLE classification.

### 3.b Text Features

For subtitle features, we cleaned the data set as described above. Then, as in video data, we ran the same three classifiers on subtitle dataset, and compared over 100 iterations with an 80%/20% training/testing split on the data. The Random Forest accuracy rate on testing was 33.95%. Using a linear Kernel SVM, we gained a testing accuracy rate of 17%, a mean improvement over MLE, but without statistical significance. And using naïve Bayes, we did not beat the MLE estimate. We also compared results to using the bag of words approach, without cleaning the data first, but found that the cleaned data set had more accuracy, and was computationally far more efficient.

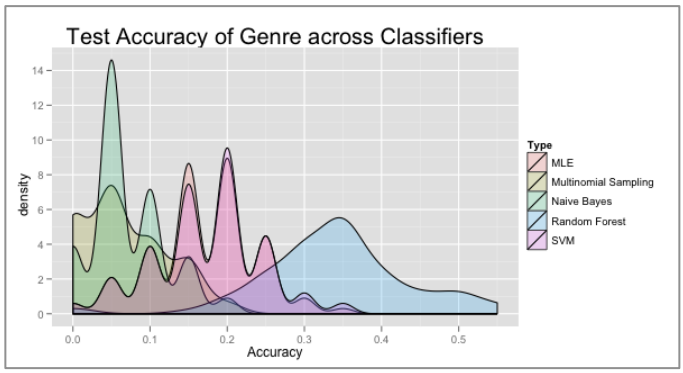


Fig. 6: Classification Accuracy Histograms

of Test Accuracy (1000 iterations, 80% training)

| Naive Bayes | SVM | Random Forest | MLE | Multinomial |
|---|---|---|---|---|
| 0.0071 | 0.1745 | 0.3395* | 0.17 | 0.066 |

* - p-value < $10^{-20}$ comparing difference in means to the MLE classification.

We also generated the top 1-gram that was most indicative of the each class, for each word w, given a certain class, c.

$$P(x = w|y = c) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_k = w\}1\{y_i = c\} + 1}{\sum_{i=1}^{m} 1\{y_i = c\}n_i + n}$$

| Action | Adventure | Animation | Comedy | Drama | Horror | Musical | Sci-Fi | Thriller |
|---|---|---|---|---|---|---|---|---|
| Bien | Solo | Vida | Solo | Nuevo | Sexo | Hacer | Bien | Pronto |

# 4. Further Work

### 4.a Larger Dataset

The selection of trailers from the-numbers only presented 312 observations, and the number of trailers found with Spanish Subtitles only numbered 100. A more complex scraping tool would allow us to download more trailers for video processing, and as closed-captioning becomes a higher priority, it may become easier to find trailers with subtitles (in Spanish or in English).

### 4.b More Features and Ensemble Learning

Other features from the trailer such as audio, including volume, length and key tune of the background music could be included into the model for better results.

Another feature we considered was mapping trailer cuts to the time sample from the original movie. The core idea is that some trailers "spoil" more of a movie than others, which when quantified may divide films in a meaningful way, either by genre or MPAA rating, or by other potential taggings such as critical reception.

Finally, an ensemble of the subtitle and video classification could be achieved if the datasets were generated with more or complete overlap.

# 5. References

1 - Zang, Qi, and Reinhard Klette. "Robust background subtraction and maintenance." *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*. Vol. 2. IEEE, 2004.

2 - Liu, Lijie, and Guoliang Fan. "Combined key-frame extraction and object-based video segmentation." *Circuits and Systems for Video Technology, IEEE Transactions on* 15.7 (2005): 869-884.