Identifying Genre of Movie Posters

Katie Foster
Emma Mack

Abstract

When artists make movie posters, they intentionally use color and structural elements to convey certain characteristics of the film, in order to market to people who would be interested in watching it. In order to explore whether computers can identify these graphic design elements, and attach them to meaning, we experimented with three different methods of classifying movie posters based off genre, specifically differentiating between Romance and Science Fiction movie posters. Our methods included simple cutoff based off average brightness of the image, using and eigenfaces approach on an image matrix including color, and using linear regression on an image matrix including color. The simple cutoff approach yielded 92% accuracy, the eigenfaces approach had 88% accuracy, and linear regression gave us 78% accuracy.

Introduction

There is a famous saying to never judge a book by it's cover, but books, movies, and other pieces of media have art made to help customers judge whether or not they would be interested in that media. These pieces of art have a lot of subconscious messaging in them, which helps to make the piece of media look interesting in order to sell more of that media. Using a computer algorithm on marketing images can help show us what sorts of subconscious signals they are giving us, which will make us more aware of our motivations. There have been several instances of people using a computer to classify movie posters or other marketing images, but most of the approaches have involved machine learning. By doing this project, we were able to find several simpler, but still reasonably accurate approaches to classifying movie posters.

Methods

We compared three different methods of differentiating between Romance and Science Fiction movie posters. First, we collected the dataset from IMDB's lists of top Romance and Science Fiction movies. This ensured that our training set was of a uniform image size, since these pages have standardized images of movie posters. We collected 74 total images and split the set into 20 training images and 17 testing images for each category.

The simplest method we used was a simple brightness cutoff method, in which we identified an average brightness threshold, and classified images solely by whether they were above or below that threshold. The first step here was to import the images to matlab, and convert the images into a matrix of color values. Then, we found the mean of every value of every red, green, and blue value in the matrix, and averaged those values. Then, we plugged that number into an if statement, which would classify the image as a Romance poster if it was above the threshold, and as a Sci-Fi poster if it was below that threshold. We chose the thresholds through trial and error, but they could also be chosen by making a graph of the mean values of all the images, and using the graph to find the point which will separate the most images into their correct categories.

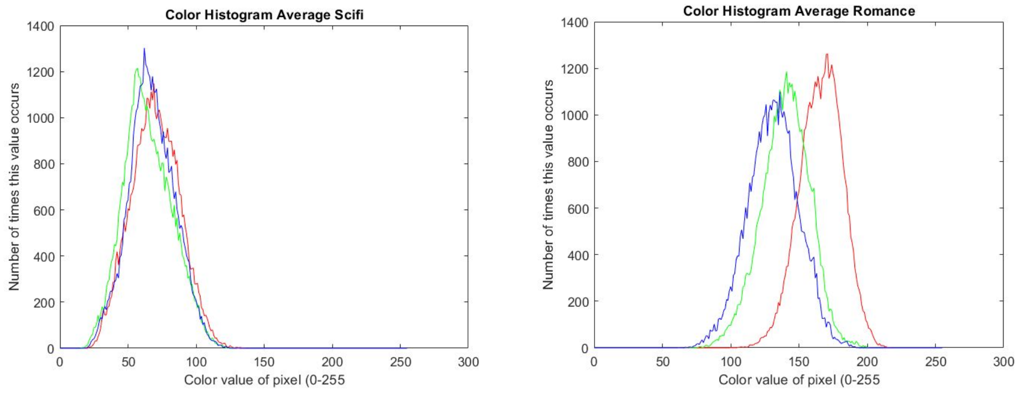
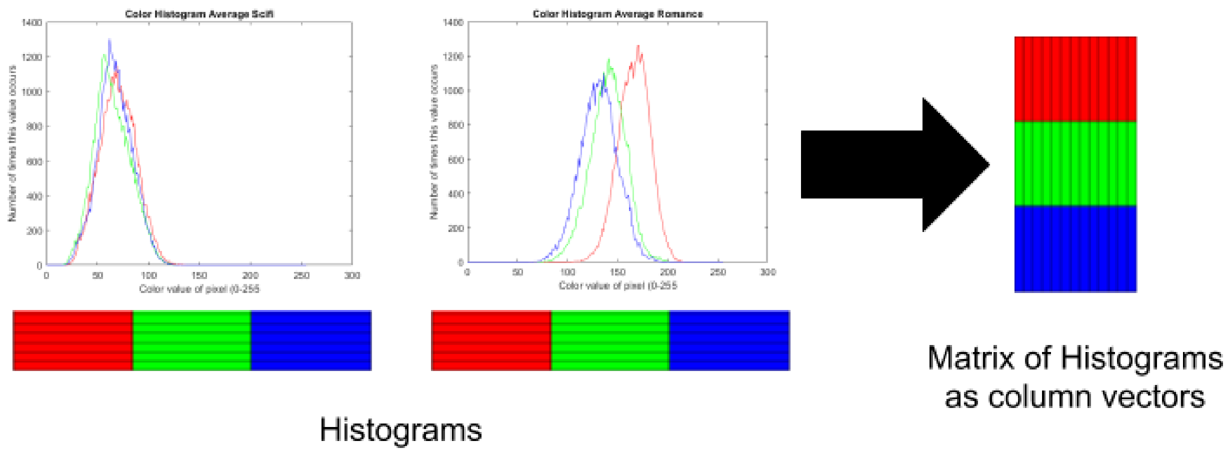


Figure 1: Color histograms of average Sci-Fi and average Romance poster respectively

Both of the next two approaches used a color histogram as a column vector in a matrix, which took some preprocessing to set up. First, our program went through all the images in each folder: Sci-Fi train, Sci-Fi test, Romance train, and Romance test, and made all of the images into matrices, and then split them into individual matrices of red, green, and blue values. Those matrices are then made into histograms, where each color value from 0 to 255 has an associated value of how many times that value occurs. And that is how we get a representation of color spread over the image. Then, the histogram data is put into a matrix where each column vector of the matrix is the histogram values of r, g, and b in a row. So there would be three column vectors that were each 255 elements long, stacked on top of each other to make a 765 element long column vector. These column vectors were put together into a matrix that has the same number of columns as there were number of images. We made two of these matrices: one of testing data and one of training data.



Color Histogram Average Scifi
Color Histogram Average Romance
Histograms
Matrix of Histograms as column vectors

Figure 2: Representation of how histograms are turned into column vectors in a matrix

Once we had a matrix of histogram data as column vectors, it was fairly simple to implement the second approach of Linear Regression. Linear regression is a method of finding equations to describe data by finding the weights, ***v***, which minimize Mean Squared Error. We generalized the MSE equation to multiple dimensions and came up with the equation:

$$(1) \qquad MSE(v) \; = \; \frac{1}{N}\Sigma_{i=1}^{N}(y_1 - (v_1x_{1i} + v_2x_{2i} \ldots))^2$$

in which ***x*** represents the input values, ***v*** represents the weights, ***y*** represents the target values. To get the weights, we simply used Matlab to solve between the training data and the transpose of training tags (which were a matrix of either 0 or 1, 0 for Romance and 1 for Sci-Fi). Then, to get the results, we just had to multiply the test data by the weights. The rest of the algorithm was just going through the matrix of results, classifying anything under 0.5 as a 0 and anything over 0.5 as a 1, and then checking for accuracy.

The last, and most complicated approach, was to apply the concept of Eigenfaces[1] to our algorithm. We followed almost all of the steps of the paper, but instead of applying the program to an image of a face, we applied it to the histogram matrix. The first step of Eigenposters™[2] is to normalize the data by subtracting out the mean histogram from each histogram. Then, we found the covariance matrix, aka A^TA, where A is the matrix of histograms. The second step is to find the eigenvectors of the covariance matrix. Once we had done the eigen decomposition of the covariance matrix, we selected the most important eigenvectors, which were usually the ones with the highest associated eigenvalues, and discarded the rest. The next step was to find out how much of each eigenvector is used to represent each image (these are the weights). We did this by multiplying the transpose of the matrix of eigenvectors by the histogram matrix. Now that we had the weights, the last step was to apply the algorithm to our test set, and evaluate the algorithm's accuracy. Whenever the algorithm is given a new image, it will find how much of each eigenvector is used to represent that image, and compare those constants to all the constants of the eigenvectors for the already existing set of images, and find the most similar ones. Our code simply subtracts the test vector from each training vector, then finds the absolute value of that vector, and sums the result. This approach is slightly different from the Eigenfaces paper, where they found the Euclidean distance, but we found our approach to be more accurate with our data. Then, once the code goes through all the training vectors, it finds the minimum of the sum of the absolute value, and selects that image as the one that most closely matches the test image.

[1] M. Turk, A. Pentland (1991). "Eigenfaces for recognition". Journal of Cognitive Neuroscience. (p. 71-76), http://www.cs.ucsb.edu/~mturk/Papers/jcn.pdf
[2] Not actually trademarked

Results and Discussion

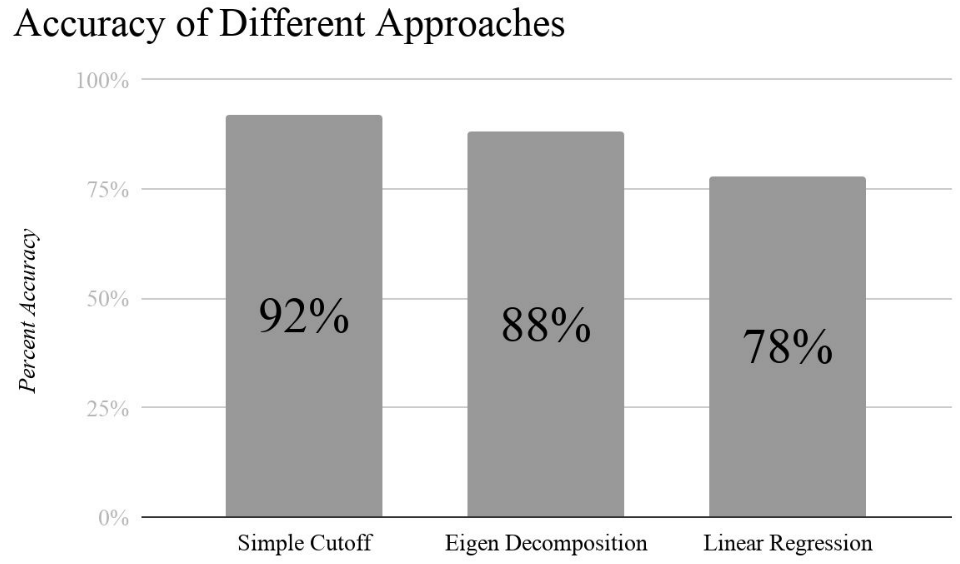


Figure 3: Bar Chart illustrating accuracy of our three different approaches

When differentiated between Romance and Sci-Fi posters, the most accurate method was the most simple one. The simple brightness cutoff method had a 92% accuracy rate, meaning that it only classified two images incorrectly in the whole testing set. This was a surprising result for such a simple method, but we hypothesized that it would not work well when we included another genre of poster in our dataset. To test that hypothesis, we made a dataset of action movies, and put those into this algorithm. Although it took some trial and error to find the right thresholds (we had to use two thresholds this time), the results for three genres were also surprisingly accurate, at 85% (especially considering how the random chance of getting it right went from 50% to 33%). So this method worked very well for datasets with two or three classes, but it has yet to be tested on datasets with more classes. We hypothesize that with more classes, the more complicated approaches will have higher accuracy.

The linear regression approach had the lowest accuracy, at 78%, which was a bit surprising but it is a simple approach. The Eigenposters approach had a slightly higher accuracy at 88%, but is still much less accurate than the simple cutoff method.

Both of these approaches may have worked better with different preprocessing of data, for example if they had been fed the actual images instead of the histogram data, since the actual images have so much more data contained in them than the histograms do. However, the extra data contained in the images might have not helped at all, as the images include data about where certain objects were in the image, which would differ more from poster to poster than the histograms would.

Conclusion

In this particular situation of differentiating between Romance and Sci-Fi posters, the simplest solution (simple brightness cutoff) is unquestionably the best method, as it is the most time efficient, and the most accurate. However, it probably won't be the best method when trying to differentiate between many different genres. However, we would need to make more datasets and run the algorithms on them in order to be sure. We have shown that there are simpler methods than machine learning to differentiate between different genres of marketing art, and that the main difference between genres is color.

References

"Feature Film, Romance (Sorted by Number of Votes Descending)", imdb.com, https://www.imdb.com/search/title?title_type=feature&genres=romance&sort=num_votes,desc&ref_=adv_prv (accessed March 30, 2019)

"Feature Film, Rating Count at least 1,000, Sci-Fi (Sorted by IMDb Rating Descending)", imdb.com https://www.imdb.com/search/title?genres=sci_fi&num_votes=1000,&sort=user_rating,desc&title_type=feature (accessed March 30, 2019)

"TOP ACTION MOVIES: 2000-2019", imdb.com, https://www.imdb.com/list/ls027328830/ (accessed March 30, 2019)

M. Turk, A. Pentland (1991). "Eigenfaces for recognition". Journal of Cognitive Neuroscience. (p. 71-76), http://www.cs.ucsb.edu/~mturk/Papers/jcn.pdf

Letter to Editor (Emily):

I tried to implement all the suggestions as best I could. There are one or two things that I don't remember what they meant, even though I took notes, so if you see something that I forgot to change, it is probably just my bad improvised notes, and not any intentional disregarding of feedback. I added a summary of results to my abstract, moved parts in my methods where I started talking about results to my results section, and elaborated on them more, briefly explained linear regression, added an equation in that explanation, took out explanation including specific matlab functions, reworded how we normalized our data, added a bar chart of results, and discussed how our method of finding the closest eigenposter was different from finding the euclidean distance.