Department of Informatics, University of Zürich

MSc Basismodul

Measuring the Similarity of Movies Based on Multidimensional Features

Luka Popovic

Matrikelnummer: 18-747-667

Email: luka.popovic@uzh.ch

April 30, 2019

supervised by Dr. Sven Helmer and Prof. Dr. Michael Böhlen

University of Zurich[UZH]

Department of Informatics

Abstract

Movie recommender systems propose movies to the users based on history of their search or based on textual descriptors or tags. All this information is generated by the users and in many cases mismatch with the real content of the movies. Our novel idea in this area is to calculate similarity measure for movies based on their visual features. This measure can then be used to cluster movies according to their similarity and classify them. This MSc Basismodul is an extension of BSc thesis done by a student at the Free University of Bozen-Bolzano. The main goal of this work is to use different similarity measure algorithm and further analyze movie keyframes and conclude what visual features of keyframes contribute to proper classification of movies.

Contents

1 Introduction

1.1 Background

In 2018 at the Free University of Bozen-Bolzano a project of analysis and classification of movies based on their visual features was started. That project brought inconclusive result that visual features of movie keyframes could be used for their classification. The project at the Free University of Bozen-Bolzano used a crossparing algorithm for calculating the similarity measure between movies. This work has the goal to further analyze what features are useful and contribute positively to correct classification of movies. For the purpose of this work, a movie dataset that consists of more than 11,500 movies has been used. This dataset was collected by a prior work [1]. The algorithm that was used for calculating similarity measure between movies is the improved version of the crossparsing algorithm from a journal paper that can be found in [2].

1.2 Crossparsing algorithm

As already mentioned crossparsing algorithm is a method for calculating similarity between two sequences. Let say we have two sequences: x=$x_1, x_2, ..., x_n$ and y=$y_1, y_2, ..., y_m$. It works by searching for the largest integer `k` such that $x_1, x_2, ..., x_k$ = $y_i, y_{i+1}, ..., y_{i+k-1}$ for some `i`. After the first `k` has been found, the algorithm searches for the longest prefix of x starting with x_{k+1} with respect to y. This process is repeated until the end of the sequence x has been reached. To make a practical example, take sequence x = $aababbbaca$ and sequence y = $babbaacabb$. The crossparsing of x with respect to y in this case is the set of phrases $s(x|y)$ = {aa, $babb$, ba, ca}. In order to determine the crossparsing distance (CPD) we repeat the same process, making the crossparsing of sequence y with respect to sequence x in order to obtain the set of phrases $s(y|x)$ [3]. The formula for the crossparsing distance that is used in this work is [2]:

$$dist_{CPD}(x, y) = \frac{\frac{|s(x|y)\backslash\{y\}|}{|x|} + \frac{|s(y|x)\backslash\{x\}|}{|y|}}{2}$$

The value of the CPD will always be between 0 and 1. The value of 0 denotes that there is no difference between two sequences and 1 denotes the existence of the largest difference between two sequences.

2 Analyzing the existing data

2.1 Existing features

All features that were used in this work belong to the class of MPEG-7 visual features [4]. The first goal of this work is to analyze the existing features that were collected in the BSc thesis that is described in [3]. Those features are Scalable Color Descriptor (SCD), Color Structure Descriptor (CSD) and Color Layout Descriptor (CLD) features. They all belong to the class of Color Descriptors. In a nutshell, SCD provides information about basic color distribution, CSD tells about local spacial distribution of colors and CLD denotes global spatial distribution of colors. The three kind of features for every keyframe were extracted and combined together in one array that consisted of 182 elements. Out of total 182 features, SCD contributed with 16 features, CSD with 32 and CLD with 134 features. The example of a feature array for the movie with n keyframes is shown on `Figure 1.`

#	1	2	3	4	..	180	181	182
$keyframe_1$	31	6	9	14	..	78	148	94
..	..	..	..	..	..	..	..	..
$keyframe_i$	21	15	25	1	..	39	202	109
..	..	..	..	..	..	..	..	..
$keyframe_n$	29	10	17	22	..	60	284	103

Figure 1.

2.2 Analysis of the existing features

2.2.1 Experiment setup

The array of 182 elements represents raw values of features and they could not be used as an input for crossparsing algorithm in that form. Main reason is that the raw values comprise of a huge range of numbers and when input like that is used in crossparing algorithm prefixes that have length greater than 1 are hard to find and as a result all sequences are different. In order to solve this problem, categorization of all features in two states, low (0) and high (1), was performed. Categorization was done by finding the median of all values that belong to

one feature in all keyframes of a movie. If the original integer value of the feature was equal or below the median, it would have a state "0", in the case the value was above the median, it would have a state "1". The example of a feature array for the movie with the n keyframes with the mapped values is shown on `Figure 2.` These arrays were used as an input to the crossparsing algorithm.

#	1	2	3	4	..	180	181	182
$keyframe_1$	1	0	0	1	..	1	1	0
..	..	..	..	..	..	..	..	..
$keyframe_i$	0	0	1	0	..	0	1	1
..	..	..	..	..	..	..	..	..
$keyframe_n$	1	0	1	1	..	1	1	0

Figure 2.

Having all keyframe feature values mapped to "0" and "1" it is suitable to use crossparsing algorithm on sequences like this. Now, instead of using row values from `Figure 2.`, its column values were used as an input to the crossparsing algorithm. The reason is that every column represents single feature in all keyframes of that movie and the goal is calculate similarity based on the corresponding features. `Figure 3.` and `Figure 4.` depict input to the crossparing algorithm for two movies, X and Y.

#	1	2	3	4	..	180	181	182
X	X^1	X^2	X^3	X^4	..	X^{180}	X^{181}	X^{182}
$keyframe_1$	1	1	0	0	..	0	1	0
..	..	..	..	..	..	..	..	..
$keyframe_i$	1	0	0	0	..	0	1	1
..	..	..	..	..	..	..	..	..
$keyframe_n$	0	0	1	0	..	1	1	1

Figure 3.

#	1	2	3	4	..	180	181	182
Y	Y^1	Y^2	Y^3	Y^4	..	Y^{180}	Y^{181}	Y^{182}
$keyframe_1$	1	1	0	1	..	1	1	1
..	..	..	..	..	..	..	..	..
$keyframe_i$	0	0	1	0	..	0	0	1
..	..	..	..	..	..	..	..	..
$keyframe_m$	1	0	1	0	..	1	1	1

Figure 4.

So we have, for a movie X, 182 integer arrays $X^1, X^2, \ldots, X^{182}$ (with values 0 and 1) of length equal to the amount of keyframes associated with that movie. Movie X is subsequently compared to every other movie Y in the dataset, with its own arrays $Y^1, Y^2, \ldots, Y^{182}$ by running the crossparsing algorithm on each pair of arrays and computing the CPD between them, ultimately returning a new array of size 182 containing all CPD values, one for each feature. For the sake of simplicity, the variables were all assigned the same weight [3]. The resulting array for each pair of movies X and Y has the following form:

$$\{CPD(X^1, Y^1), CPD(X^2, Y^2), ...CPD(X^{182}, Y^{182})\}$$

In order to further simplify this data, the values of the CPDs in each of these arrays were averaged and used as a value for similarity between two movies. As a result the process of calculating CPD between two movies X and Y can be represented as:

$$CPD(X, Y) = \frac{1}{182} \sum_{i=1}^{182} CPD(X^i, Y^i)$$

2.2.2 Actual experiment

As an input to crossparsing algorithm a subset of 20 movies (4 genres with 5 movies each) was used in this iteration. The main reasons for this decision are high computational complexity and time that takes to calculate CPD. Calculating CPD on the subset of 20 movies took 5 hours. The approach with the mapping real values was the same as one performed in the BSc thesis and results obtained were very similar to the ones found in [3], although the newer version of the crossparsing algorithm was used. Out of 5 comedy movies classification algorithm did not classify any of those 5 movies as comedy, same applies for 5 drama movies, 4/5 romance movies were classified as romance, and for the western movies again classification algorithm classified 0 movies to its belonging genre, western movies. For the presentation of results confusion matrix was used. Confusion matrix consists of columns that represent actual genres of movies while the rows represent genres of movies assigned by classification algorithm. The

values in the matrix are calculated by taking a movie from one genre and finding the smallest CPD value between movie being observed and all other movies. The smallest CPD value actually represents the similarity measure that tells to what movie observed one is most similar to. Finally, the genre of similar movie denotes the genre in which the observed movie was classified. We are most interested in values that appear on the main diagonal of the confusion matrix. The greater number of movies that appear on the main diagonal indicate a higher accuracy of the classification algorithm since in that case genres assigned by classification algorithm match real genres of movies. Since in our experiment the subset of 20 movies divided in 4 genres was used, every row of confusion matrix has total sum of 5 movies. The confusion matrix result of previous experiment is presented in the `Figure 5.`

	$Comedy$	$Drama$	$Romance$	$Western$
$Comedy$	0	0	5	0
$Drama$	1	0	4	0
$Romance$	0	1	4	0
$Western$	0	0	5	0

Figure 5.

The classification of movies using three visual features descriptors and assigning two states to original integer values showed rather moderate but promising potential to classify movies based on visual features since classification for romance movies was pretty well. In order to get better results another approach was used. Instead of using low (0) and high (1) mapping values next focus was to use four mapping states low low (0), low high (1), high low (2) and high high (3). This was done because in a case of only two states mapping all movies, regardless of their belonging to the same genre or not, had low CPD value. The border values of new sets were obtained by finding median of all values that belong to one feature in all keyframes of a movie, then on two sets of values (one sets consists of all values that are below the median, and second of all values that are above the median) median is found again. Then crossparsing algorithm was performed and crossparsing distances were obtained. Surprisingly, this method showed no improvements in comparison when two mapping values were used.

2.2.3 Further improvements

Since the beginning of the work, there was an idea that it is likely that no all features contribute positively to the calculation of the CPD. Hence, a next step was to determine what features should be kept and used for calculation of similarity between movies. The following implementation was used: first per genre CPD for the subset of 20 movies was calculated. That means that average of all CPDs between movies that belong to the same genre was found. Then one feature is omitted from the features dataset and again per genre CPD was calculated. If the newly calculated per genre CPD was higher that the original one (when all features

were used) then that means that omitted feature contributed positively to the per genre CPD (when feature is omitted per genre CPD goes up, meaning that movies that actually belong to the same genre are more disjointed, but when the feature is present, per genre CPD is lower, which should be the case since movies belonging to the same genre should have CPD value as low as possible). When this method was performed and only features that contribute positively to the per genre CPD (122 features) were used in calculation of CPD the results obtained were slightly better but still far away from perfect.

Last thing that was done was keeping features that contribute to the per genre CPD above some threshold. In other words, if the difference between per genre CPD with omitted feature and per genre CPD with all features is above some threshold then it is considered that feature has a significant impact on CPD, otherwise it makes insignificant contribution. Finally, when features that have high contribution to the increase of per genre CPD were used (11 features), the classification result was still the same as one without the threshold (when all features that contribute positively to the per genre CPD were used).

It is important to mention that the initial idea with keeping features was to keep only features that positively contribute to both per genre CPD and non per genre CPD (this values is calculated by finding average of all CPDs of movies that do not belong to the same genre). Positive contribution in a case of non per genre CPD means that after feature omission, non per genre CPD drops, because omitted feature contributed to keeping CPD of movies from different genres high. When this method was applied, only one feature that contributed positively to both per genre and non per genre CPD was found. In many further experiments this method yielded zero features so this idea was rejected and only positive contribution to the per genre CPD was observed . The results of this method just confirmed the notion noticed so far and that is that CPDs between movies that belong to the different genres actually behave very similarly to the CPDs of movies from the same genres. This is the reason why there was only one feature which omission made per genre CPD to raise and non per genre CPD to drop, since in most cases non per genre CPD raised as well.

The bad results of classification algorithm also lead to conclusion that the three descriptors: Scalable Color Descriptor (SCD), Color Structure Descriptor (CSD) and Color Layout Descriptor (CLD) do not provide enough information for determining their similarity in terms of genres. Next idea was to collect more features from the keyframes and expand analysis using them. The next chapter is focused on analyzing the new extracted features.

3 Analyzing new features data

3.1 New features dataset

The new features that were extracted are Dominant Color Descriptor (DCD), Edge Histogram Descriptor (EHD) and Homogeneous Texture Descriptor (HTD) features. Dominant Color Descriptor belong to the class of Color Descriptors while Edge Histogram Descriptor and Homogeneous Texture Descriptor belong to the class of Texture Descriptors. DCD comprises of the dominant colors values, their percentage value and variance and the spatial coherency, EHD captures the spatial distribution of edges and represents local-edge distribution in the image while HTD provides a quantitative characterization of texture. The new keyframe feature array had 163 elements. Out of total 163 features, DCD contributed with 6 features, EHD with 97 and HTD with 60 features.

3.2 Analysis of the new features

On the new features dataset we perfomed the same methodology as on the dataset collected in the BSc thesis. Firstly, raw values of the features were mapped to four states low low (0), low high (1), high low (2) and high high (3) as already described using median values. After that crossparsing algorithm was performed and crossparsing distances were obtained using all newly extracted features. Out of 5 comedy movies classification algorithm classified only 1 movie as comedy, out of 5 drama movies only 1 as drama, 4/5 romance movies were classified to their real genre, romance, and for the western movies classification algorithm classified 0 movies to western genre. This result is also shown in a form of confusion matrix in the `Figure 6.` It is easily noticeable that the result is not close to correct classification but it is better than result obtained when SCD, CSD and CLD were used.

	$Comedy$	$Drama$	$Romance$	$Western$
$Comedy$	1	0	4	0
$Drama$	0	1	4	0
$Romance$	0	1	4	0
$Western$	0	0	5	0

Figure 6.

Similarly to the analysis of the existing features, the idea to keep features that positively contribute to the per genre CPD was performed. When this approach was used, with no threshold the classification algorithm performed same as when all features were used, since all features that positively contribute to the per genre CPD actually were all 163 features. After this, the case with threshold (157 features) was used and better results were acquired. Instead classifying 1/5 movies as drama in this case 3/5 movies were classified as drama but for the other genres results were the same, and still no proper classification for western movies was obtained.

Using new features yielded better results, but that is still not enough for correct classification of all movie genres based on their similarity measure obtained from their visual features. Next step was to combine all features together and to see how this will affect CPDs and classification.

4 Analyzing all features

4.1 All features dataset

The new dataset that was used consisted of all features obtained so far. Those features are Scalable Color Descriptor (SCD), Color Structure Descriptor (CSD), Color Layout Descriptor (CLD), Dominant Color Descriptor (DCD), Edge Histogram Descriptor (EHD) and Homogeneous Texture Descriptor (HTD) features. The keyframe feature array had 345 elements.

4.2 Analysis of the all features

The same principle was performed again. The mapping to four states and calculation of the CPDs was done. Firstly, we considered all 345 features for CPD calculation. The results were the same as when existing features were used (SCD, CSD and CLD), which confusion matrix can be found in the `Figure 5`. The reason behind this lies in fact that SCD, CSD and CLD together contribute with 182 out of 345 features and that their CPD values prevail over DCD, EHD and HTD CPD feature values. Next, the features that contribute positively (no threshold case, 146 features) to the per genre CPD were kept for CPD calculation. In this case, as in the next case when threshold (37 features) was used no significant improvement was seen. For the drama genre, 1/5 instead previous 0/5 was classified correctly but this is still not proper classification. The result of no threshold and threshold case experiment is shown in a form of confusion matrix in the `Figure 7.`

	$Comedy$	$Drama$	$Romance$	$Western$
$Comedy$	0	0	5	0
$Drama$	1	1	3	0
$Romance$	0	1	4	0
$Western$	0	0	5	0

Figure 7.

5 Conclusion and future work

In this work, the main idea was to calculate similarity measure of movies based on their visual features. For the similarity measure calculation crossparsing algorithm was used. Furthermore, in order to obtain better results different features and their combination as well as different approaches of feature selection was used but no significant improvement to the proper classification of movies was obtained. Nevertheless, this does not mean that it is infeasible to classify movies based on their visual features, just that further research is needed.

Some suggestions for the future work will be to try to visualize the data using principal component analysis [5] and singular value decomposition [6]. This would bring another look at the movie dataset and provide new insights about movie features. Furthermore, another interesting approach would be to interpret the movie keyframe sequences as multivariate time series and measure their distance or cluster them. Last but not least, new ideas are arising, like instead of using features, movies tags might provide useful information that would help to more precisely calculate movies similarity.

Bibliography

[1] Elahi, M., Deldjoo, Y., Bakhshandegan Moghaddam, F., Cella, L., Cereda, S., & Cremonesi, P. (2017, August). *Exploring the semantic gap for movie recommendations*. In Proceedings of the Eleventh ACM Conference on Recommender Systems(pp. 326-330). ACM.

[2] Helmer, S., Augsten, N. & Böhlen, M. *Measuring structural similarity of semistructured data based on information-theoretic approaches*. VLDB J., 2012.

[3] Burgio, G. *Video Similarity Analysis based on MPEG-7 Visual Features*. Unibz, BSc Thesis, 2018.

[4] MPEG-7:*Visual descriptors*, `https://mpeg.chiariglione.org/standards/mpeg-7/visual`, last visit April 24th 2019

[5] Setosa: *Principal Component Analysis*, `http://setosa.io/ev/principal-component-analysis`, last visit April 25th 2019

[6] Towards Data Science: *PCA and SVD explained with numpy*, `https://towardsdatascience.com/pca-and-svd-explained-with-numpy-5d13b0` last visit April 25th 2019