Automatically Creating and Comparing Features and Contexts

Pablo Bermejo[1], Frank Hopfgartner[2], and Joemon M. Jose[2]

[1] University of Castilla-La Mancha, Albacete, Spain,
pbermejo@dsi.uclm.es
[2] University of Glasgow, Glasgow G12 8RZ, United Kingdom,
{hopfgarf,jj}@dcs.gla.ac.uk

Abstract. Representation of multimedia documents is a major problem in multimedia information retrieval due to the semantic gap. In this work, we present a tool to automatically create and evaluate three different kind of features for a database to decide what kind of representation is better suited for representing multimedia documents. Our tool provides the option of splitting interaction log files into different contexts to study their influence on classifiers' performance.

1 Introduction

A common approach to predict the value of the class a document belongs to is to select representative document features. In the text domain, predestined features are vocabulary features, as they can express the semantics of the documents. In the multimedia domain however, selecting representative features becomes a challenging task. Many videos for instance are not enriched with textual annotations [6], relying on vocabulary features is hence not adequate. Moreover, widely used low-level features such as colour histograms or textures are not sufficient enough to describe the content of the document they are meant to represent. The reason for this is the semantic gap [9]: the difference between (computable) low-level representations of a multimedia document and the actual semantic meaning of the document. Identifying representative features and feature combinations can be a key breakthrough in the challenge of bridging the semantic gap, as the quality of the used set of features is of great importance for a classifier to achieve a good performance [2].

In this paper, we present our work on automatically transforming the representation of a set of video documents into one of three different feature types: behavior, object and vocabulary features. The software is based on Weka [12] and can be used to evaluate a dataset based on extracted features.

Our tool allows to split the features into different contexts that can be found in log files of user experiments, where users test a video retrieval system by searching for different topics and under different conditions. Thus it can be investigated which contexts improve the classifier's performance and which ones make it perform worse. Some example contexts our tool can handle are:

1. *Topic.* Splitting the database with respect to the topic a shot was accessed by the user.
2. *Condition.* User might perform the search under different conditions, for example different interfaces.
3. *User Experience.* Database is split according to the experience of users accessing shots.
4. *Topic Difficulty.* Database is split according to the difficult (as perceived by user) of each search task.
5. *Mix of contexts.* Contexts can be mixed in pairs.

This paper is structured as follows: In the next section, we introduce the three types of features which can be analysed by our tool. In Section 3 we provide an overview how to use our tool. Finally, in Section 4 we present our main conclusions for this work.

2 Feature Types

So far, research on feature construction and document classification has been concentrated on three different categories: behaviour features, object features and vocabulary features. In the remainder of this section, we will introduce these features.

2.1 Behaviour Features

User Behavior Features [5, 1] give information about how the user interacts with a document. In retrieval systems, this information is related to the actions the user performs on retrieved documents. Hence, behaviour features can be extracted from user log files. We consider the behavior features shown in Table 1. They can be split into three groups: *Click-Through features*, which represent information about clicks the user performed on retrieved documents; *Browsing features*, which show different metrics about time spent with a result and *Query-Text features*, which count words in the current text query and make comparisons with other text queries.

2.2 Object Features

Object Features are extracted from the data collection. Typical features are *low-level features* and additional *metadata*. Common visual features used in the video domain are for example colour layout, dominant colour, textures or edge histograms which have been extracted from representative keyframes. Audio features can be rhythm histograms or spectrum descriptors of a video. In the video domain, *metadata* keeps information such as the length of retrieved results or information related to the textual transcript of the retrieved videos.

Feature name	Description
ClickFreq	Number of mouse clicks on shot
ClickProb	*ClickFreq* divided by total number of clicks
ClickDev	Deviation of *ClickProb*
TimeOnShot	Time the user has been performing any action on shot
CumulativeTimeOnShots	*TimeOnShot* added to time on previous shots
TimeOnAllShots	Sum of time on all shots
CumulativeTimeOnTopic	Time spent under current topic
MeanTimePerShotForThisQuery	Mean of all values for *TimeOnShot*
DevAvgTimePerShotForThisQuery	Deviation of *MeanTimePerShotForThisQuery*
DevAvgCumulativeTimeOnShots	Deviation of *CumulativeTimeOnShots*
DevAvgCumulativeTimeOnTopic	Deviation of *CumulativeTimeOnTopic*
QueryLength	Number of words in current text query
WordsSharedWithLastQuery	Number of equal words in current query and last query

Table 1. Behavior Features used to predict shots relevance

2.3 Vocabulary Features

Vocabulary Features are a bag of words created from the transcript of multimedia data. It is expected that video relevance classification based on textual transcripts is promising due to the fact that text has more descriptive power than, for example, low-level visual features. Nevertheless, transcripts do not always relate to the content of a video [13] and shots do not have a long transcript to base an analysis on. Therefore, we consider not only the transcript of one single video element, but also from the neighboured n shots with respect to the time of the video. In our case, text is not used to compute statistics, but to create a vocabulary of words to perform the typical task of classification based on text. It is expected that n-Windowed Vocabulary features perform better than creating a bag of words from the transcript of a single shot only. Our tool filters text through a stop-word list and performs Porter stemming [8] for further analysis. When we use text to create a bag of words and evaluate using a bayesian classifier, we do not use Naive Bayes but the Naive Bayes Multinomial, which is recommended for text classification [7].

3 Tool Usage

A first condition for using the tool is to format the user log files in the Attribute-Relation File Format (.arff) as introduced for the Weka machine learning software [12]. After tuning all parameters in the configuration files, the tool can be executed by running the following Java classes:

1. *CreateFeatures.* This class converts the input file to another .arff file where instances are represented by the created features. The output file contains a projection of the previous log file where the only features are the ones created, besides the tuple descriptor features (userID, shot name,...). Parameters to tune are: type of features to create, kind of relevance (official or user-defined) to use, order of the output features.

2. *SplitContexts*. This program splits the constructed database in as many parts as the chosen context can be split. Several contexts can be chosen from the configuration file. For example, if in the experiment there were four search topics and the tool's user wants to split the .arff file in contexts based on search topic, then four .arff files will be created. Each of these files will contain instances with the same topic attribute value in the original file. If no context differentiation is desired, then no split is done. The descriptor tuple features are removed in any context case as the aim is to evaluate using just the created features (behaviour, object or vocabulary) without any knowledge of userID, topicID, or shot name.
3. *Evaluate*. The evaluation class performs a cross validation of the .arff files created by SplitContexts. Weka is used for the actual classification task. The following parameters can be defined in the configuration file: number of folders in cross validation, times to perform cross validation, classifiers to use (Naïve Bayes, Naïve Bayes Multinomial, Support Vectors Machine, k-NN, and AODE available) and balance level of training sets (from 0 for no change, to 100 to become training sets fully balanced). The class outputs several metrics computed for each classifier and possible class value: Accuracy, Precision, Recall, F1-measure.
4. *IncrementalSelection*. Finally, an incremental wrapper-based feature subset selection [3] is performed over the chosen databases and then a projection of the data using the selected features is evaluated. Evaluation results are displayed showing the chosen feature subsets and the same metrics mentioned above.

3.1 Example of work done using this took

To learn how different kinds of constructed features affect relevance prediction, Bermejo et al. [4] used data logs from two users experiments [11, 10]. Each log file contained verbose data of the actions each user performed while interacting with different video retrieval interfaces. Bermejo et al. used the presented tool to construct Behavior, Object and Vocabulary Features from these log files. They conclude that windowed Vocabulary features seem to be the best when there is text associated with the video. Behavior features were only useful if stored behavior is used in a collaborative system to influence the retrieved results. Finally, Object features return lesser relevant shots and are more consistent.

4 Conclusions and Future Work

Our tool makes it easy to create three different kind of features and thus to compare which representation is better suited for the representation of multimedia documents. Contexts splits can be created and their influence on classifiers studied. Finally, feature selection can be easily performed over databases or context splits created. As future work, we propose to improve the command line interface into a GUI and make the configuration files simpler to tune for the user.

5 Acknowledgments

This work has been partially supported by the JCCM under project PCI08-0048-8577, MEC under project TIN2007-67418-C03-01 and FEDER funds. The second author was supported by the European Commission under contract K-Space (FP6-027026).

References

1. E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–26, New York, NY, USA, 2006. ACM Press.
2. R. Bekkerman, A. McCallum, and G. Huang. Automatic categorization of email into folders: Bechmark experiments on enron and sri corpora. Technical report, Department of Computer Science. University of Massachusetts, Amherst., 2005.
3. P. Bermejo, J. Gámez, and J. Puerta. On incremental wrapper-based attribute selection: experimental analysis of the relevance criteria. In *IPMU'08: Proceedings of the 12th Intl. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2008.
4. P. Bermejo, H. Joho, R. Villa, and J. M. Jose. Comparison of feature construction methods for video relevance prediction. In *MMM '09: Proceedings of the 15th International MultiMedia Modeling Conference*, 1 2009. to appear.
5. S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve web search. *ACM Trans. Inf. Syst.*, 23(2):147–168, 2005.
6. M. J. Halvey and M. T. Keane. Analysis of online video search and sharing. In *HT '07: Proceedings of the eighteenth conference on Hypertext and hypermedia*, pages 217–226, New York, NY, USA, 2007. ACM.
7. A. McCallum and N. K. A comparison of event models for naive bayes text classification. In *AAAI/ICML-98 Workshop on Learning for Text Categorization*, pages 41–48, 1998.
8. M. F. Porter. An algorithm for suffix stripping. pages 313–316, 1980.
9. A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-Based Image Retrieval at the End of the Early Years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380, 2000.
10. R. Villa, N. Gildea, and J. M. Jose. FacetBrowser: a user interface for complex search tasks. In *ACM Multimedia 2008*, 10 2008. to appear.
11. R. Villa, N. Gildea, and J. M. Jose. A study of awareness in multimedia search. In *Joint Conference on Digital Libraries*, pages 221–230, June 2008.
12. I. H. Witten and F. Eibe. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2 edition, 2005.
13. R. Yan and A. G. Hauptmann. Co-retrieval: A boosted reranking approach for video retrieval. In *CIVR*, pages 60–69, 2004.