TUD-MMC at MediaEval 2016: Context of Experience task

Bo Wang
Delft University of Technology
Delft, The Netherlands
b.wang-6@student.tudelft.nl

Cynthia C. S. Liem
Delft University of Technology
Delft, The Netherlands
C.C.S.Liem@tudelft.nl

ABSTRACT

This paper provides a three-step framework to predict user assessment of the suitability of movies for an inflight viewing context. For this, we employed classifier stacking strategies. First of all, using the different modalities of training data, twenty-one classifiers were trained together with a feature selection algorithm. Final predictions were then obtained by applying three classifier stacking strategies. Our results reveal that different stacking strategies lead to different evaluation results. A considerable improvement can be found for the F1-score when using the label stacking strategy.

1. INTRODUCTION

A substantial amount of research has been conducted in recommender systems that focus on user preference prediction. Here, taking contextual information into account can have significant positive impact on the performance of recommender systems [1].

The MediaEval *Context of Experience* task focuses on a specific type of context: the viewing context of the user. The challenge considers predicting the multimedia content that users find most fitting to watch in a specific viewing condition, more specifically, while being on a plane.

2. DATASET DESCRIPTION AND INITIAL EXPERIMENTS

The dataset for the *Context of Experience* (CoE) task[5] contains metadata and pre-extracted features for 318 movies [6]. Features are multimodal and include textual features, visual features and audio features. The training set contains 95 labeled movies, which are labeled as 0 (bad for airplane) or 1 (good for airplane).

A set of initial experiments has been conducted in order to evaluate the usefulness of the various modalities in the *CoE* dataset [6]. A rule-based PART classifier was employed to evaluate the feature performance in terms of Precision, Recall and F1 Score, the result can be found in Table 1.

3. MULTIMODAL CLASSIFIER STACKING

Ensemble learning uses a combination of different classifiers, usually getting a much better generalization ability. This particularly is the case for *weak learners*, which can be defined as learning algorithms that perform just slightly better than random guessing by themselves, but can be jointly grouped into an algorithm with arbitrarily high accuracy [2].

MediaEval 2016 Workshop, Oct. 20-21, 2016, Hilversum, Netherlands.

Features used	Precision	Recall	F1
User rating	0.371	0.609	0.461
Visual	0.447	0.476	0.458
Metadata	0.524	0.516	0.519
Metadata + user rating	0.581	0.6	0.583
Metadata + visual	0.584	0.6	0.586

Table 1: Results obtained by applying a rule-based PART classifier to the *Right Inflight* dataset.

Therefore, we were interested in taking a multimodal classifier stacking approach to the given problem, and use a combination of multiple weak learners to 'boost' them into a strong learner.

The process can be separated into three stages: classifier selection, feature selection and classifier stacking.

3.1 Classifier Selection

First of all , we want to select base classifiers that will be useful candidates in a stacking approach. For this, we use the following classifier selection procedure:

1. Initialize a list of candidate classifiers. For each modality, we consider the following classifiers: k-nearest neighbor, nearest mean, decision tree, logistic regression, SVM, bagging, random forest, AdaBoost, gradient boosting, and naive Bayes. We do not apply parameter tuning, but take the default parameter values as offered by scikit-learn[1].

2. Perform 10-fold cross-validation on the classifiers. As input data, we use the training data set and its ground truth labels, per single modality. For the audio MFCC features, we set NaN values to 0, and calculate the average of each MFCC coefficient over all frames.

3. If Precision and Recall and F1-Score > 0.5, keep the candidate classifier on the given modality as base classifier for our stacking approach.

The selected base classifiers and their relevant modalities can be found in Table 2. It should be noted that the performance of `Bagging` and `Random forest` is not stable. This is because Bagging tries to use different subset of instances in each run and RandomForest tries to use different subsets of instances and features in each run.

3.2 Feature Selection

For each classifier and corresponding modality, a better-performing subspace of features may optimize results further. Since we have

[1]http://scikit-learn.org/

Classifier	Modality	Precision	Recall	F1
k-Nearest neighbor	metadata	0.607	0.654	0.630
Nearest mean classifier	metadata	0.603	0.579	0.591
Decision tree	metadata	0.538	0.591	0.563
Logistic regression	metadata	0.548	0.609	0.578
SVM (Gaussian Kernel)	metadata	0.501	0.672	0.574
Bagging	metadata	0.604	0.662	0.631
Random Forest	metadata	0.559	0.593	0.576
AdaBoost	metadata	0.511	0.563	0.536
Gradient Boosting Tree	metadata	0.544	0.596	0.569
Naive Bayes	textual	0.545	0.987	0.702
k-Nearest neighbor	textual	0.549	0.844	0.666
SVM (Gaussian Kernel)	textual	0.547	1.000	0.707
k-Nearest neighbor	visual	0.582	0.636	0.608
Decision tree	visual	0.521	0.550	0.535
Logistic regression	visual	0.616	0.600	0.608
SVM (Gaussian Kernel)	visual	0.511	0.670	0.580
Random Forest	visual	0.614	0.664	0.638
AdaBoost	visual	0.601	0.717	0.654
Gradient Boosting Tree	visual	0.561	0.616	0.587
Logistic Regression	audio	0.507	0.597	0.546
Gradient Boosting Tree	audio	0.560	0.617	0.587

Table 2: Base classifier performance on multimodal dataset.

multiple learners, we employed the *Las Vegas Wrapper*(LVW) [3] feature selection algorithm for a feature subset selection. For each run, LVW generate a list of random features and evaluate the learner's error rate for n times, and select the best performing feature subspace as output.

In our case, we slightly modified LVW to optimize F1 score, where the original las vegas wrapper was developed for optimize accuracy.

For each base classifier, with the exception of the random forest classifier (as it already performs feature selection), we apply the LVW method, and achieve performance measures as listed in Table 2.

3.3 Classifier Stacking

In previous research, classifier stacking (or metalearning) has been proved beneficial for predictive performance by combining different learning systems which each have different inductive bias (e.g. representation, search heuristics, search space) [4]. By combining separately learned concepts, meta-learning is expected to derive a higher-level learned model that more accurately can predict than any of the individual learners. In our work, we consider three types of stacking strategies:

1. *Majority Voting:* this is the simplest case, where we select classifiers and feature subspaces through the steps above, and assign final predicted labels through majority voting on the labels of the 21 classifiers.

2. *Label Stacking:* Assume we have n instances and T base classifiers, then we can generate an n by T matrix consisting of predictions (labels) given by each classifier. Label combining strategy tries to build a second-level classifier based on this label matrix, and return a final prediction result for that.

3. *Label-Feature Stacking:* Similar to label stacking, label-feature stacking strategy uses both base-classifier predictions and features as training data to predict output.

4. RESULTS

We considered all prediction results by the 21 selected base classifiers, and then applied the three different classifier stacking strategies to the test data using 10-fold cross-validation. As results for label stacking vs. label attribute stacking were comparable on the training data, we only consider voting vs. label stacking on the test data.

All obtained results, on the training (development) and test dataset, are given in Table 3. On the training data, we notice significant improvement can be found in terms of Precision, Recall as well as F1 score in comparison to results obtained on individual modalities. The voting strategy results in the best precision score, but has bad performance in terms of recall. On the contrary, label stacking has higher recall and the highest F1 score.

Considering results obtained on the test dataset, we can conclude that *label stacking* is more robust than the voting strategy. For voting strategy, a significant decrease can be found in terms of precision on test set. This is because majority vote (and Bayesian averaging) tendency to over-fit derives from the likelihood's exponential sensitivity to random fluctuations in the sample, and increases with the number of models considered. Meanwhile, label stacking strategy performs reasonable well on test data.

Stacking Strategy	Precision	Recall	F1
Voting (cv)	0.94	0.57	0.71
Label Stacking (cv)	0.72	0.86	0.78
Label Attribute Stacking (cv)	0.71	0.79	0.75
Voting (test)	0.62	0.80	0.70
Label Stacking (test)	0.62	0.90	0.73

Table 3: Classifier Stacking results.

5. CONCLUSIONS

In our entry for the MediaEval CoE task, we aimed to improve classifier performance by a combination of classifier selection, feature selection and classifier stacking. Results reveal that employing a ensemble approach can considerably increase the classification performance, and is suitable for treating the multimodal *Right Inflight* dataset.

The larger diversity of base classifiers is able to produce a more robust ensemble classifier. On the other hand, a blending of multiple classifiers may also have some drawbacks, e.g computational costs, and difficulty in traceable interpretation.

We expect better results for our method can still be obtained through parameter tuning, and by applying more robust classifier stacking methods, such as feature weighted linear stacking [7].

6. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.

[2] Y. Freund and R. E. Schapire. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. *Journal of computer and system sciences*, 55:119–139, 1997.

[3] H. Liu and R. Setiono. Feature selection and classification—a probabilistic wrapper approach. In *Proceedings of the 9th International Conference on Industrial and Engineering Applications of AI and ES*, pages 419–424, 1997.

[4] A. Prodromidis, P. Chan, and S. Stolfo. Meta-learning in distributed data mining systems: Issues and approaches. In

Advances in distributed and parallel knowledge discovery, pages 81–114. MIT/AAAI Press, 2000.

[5] M. Riegler, , C. Spampinato, M. Larson, P. Halvorsen, and C. Griwodz. The mediaeval 2016 context of experience task: Recommending videos suiting a watching situation. In *Proceedings of the MediaEval 2016 Workshop*, 2016.

[6] M. Riegler, M. Larson, C. Spampinato, P. Halvorsen, M. Lux, J. Markussen, K. Pogorelov, C. Griwodz, and H. Stensland. Right inflight? A dataset for exploring the automatic prediction of movies suitable for a watching situation. In *Proceedings of the 7th International Conference on Multimedia Systems*, pages 45:1–45:6. ACM, 2016.

[7] J. Sill, G. Takacs, L. Mackey, and D. Lin. Feature-weighted linear stacking. arXiv:0911.0460, 2009.