HIERARCHICAL SOUND EVENT DETECTION

Maria E. Niessen, Tim L. M. Van Kasteren, Andreas Merentitis

AGT International
Hilpertstr. 35, 64295 Darmstadt, Germany
{mniessen,tkasteren,amerentitis}@agtinternational.com

1. INTRODUCTION

Environmental sound recognition in real-world conditions is a particularly challenging topic, since it requires significant efforts on both the feature extraction and the classification modeling parts in order to achieve satisfactory results. In the presented work we propose a multi-tier method that employs best of breed techniques at all relevant tasks; initially feature extraction takes place focusing on a broad range of audio features. Following feature extraction a Hierarchical Hidden Markov Model classifier scheme with explicit modeling of the finishing of the state to better detect transitions is developed. Finally, the best result is achieved when ensemble methods are added on top of the previous scheme. Specifically, a variation of Stacking using a Random Forest and the HHMM as level-1 classifiers and a second instance of HHMM as the metaclassifier is selected. Results indicate that this final method is on one hand able to deliver the best overall performance, as well as explore different tradeoffs between classes and metrics (e.g. emphasize on specific metrics or classes that are of higher importance).

2. FEATURE EXTRACTION

We implemented a selection of standard features used in research on recognition of sounds. Because the sound events in an office environment have very variable characteristics we used not only features that are successful in speech and music processing, such as Mel-frequency coefficients (MFCCs) and zero-crossing rate (ZCR), but also features that can better describe impact sounds such as door knocking and complex sounds such as the printer. A combination of temporal, spectral, and perceptual features have been used successfully in complex real world scenarios [1, 2]. In addition, features based on the auto-correlation of the signal are used by Valenzise et al. [2] to distinguish screams from noise, that is, periodic sounds from non-periodic ones. Table 2 shows the list of features that were used in our experiments.

We use the same settings for all features: a window size of 80 ms with an overlap of 50% and two window types, Hamming and a rectangular window. For the two energy roll-off features we applied a threshold of 85% of the maximum energy. The linear prediction coefficients (LPC) are calculated with the covariance method. All the features are used in the classification model, but the floating search method [5] indicates that the spectral centroid, the 3rd and 5th Mel-frequency coefficient, and the short-term energy are the most predictive features, either because they are very predictive for one class in particular (e.g. the short-term energy for the printer class), or because they have relatively small within-class variation for some classes (e.g. the spectral centroid for the harmonic classes). In general, most of the spectral features, the lower Mel-frequency coefficients, and the lower linear predictive coefficients are most useful.

Table 1: Audio features

Temporal	Spectral	Auto-correlation	Multi-dimensional
STE	Flux	Flux	13 MFCCs [3]
ZCR	Roll-off	Roll-off	12 LPCs [4]
Flatness	Flatness		
	Brightness		
	Roll-off		

3. HIERARCHICAL HMM

Instead of classifying each frame independently, a model that takes into account the temporal dependencies within and between sound events can improve recognition performance [6]. Therefore we apply a two-layer hierarchical model that has been developed for activity recogntion to classify the audio features [7]. The top layer of the state representations are the sound events as they are annotated in the Office Live (OL) dataset, such as printing, while the bottom layer represent the sub-events of this class. For example, the sound event of printing can be segmented into a sub-event for the sound the printer makes when loading paper from a tray, one for the actual printing, and one for the printer feeding the printed paper to the output tray. Although it is possible to train such a model using data which is annotated with labels of both sound events and the sub-events, we train the model using only labels for the top layer sound events. There are two advantages to this approach: 1) Annotating the data becomes significantly less involved when only the sound events have to be annotated, 2) We do not force any structure upon the model with respect to the actions, but rather let the model find this structure in the data automatically. The automatic allocation of structure can be considered as a clustering task. The clusters found in the data do not necessarily have to be meaningful clusters that correspond to actual atomic sub-events that are intuitive to humans. We therefore distinguish between the term ‘sub-event clusters’ to refer to the sub-events found through clustering and ‘sub-events’ to refer to the sub-events intuitive to humans.

Figure 1 shows the structure of the hierarchical HMM model in which the hidden state representation is split into a top and a bottom layer. The top layer state variables y_t represent the sound

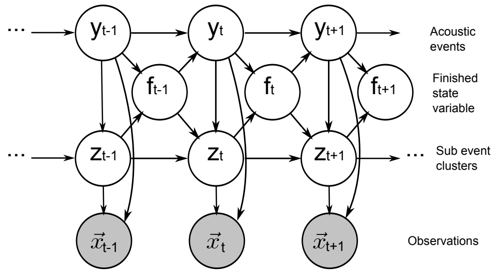

Figure 1: The graphical representation of a two-layer HHMM. Shaded nodes represent observable variables, the white nodes represent hidden states.

events and the bottom layer state variables z_t represent the sub-event clusters. Each sound event consists of a sequence of sub-event clusters and the temporal ordering of the sub-event clusters in a sequence can vary between different executions of an activity. Of particular interest to us is the last sub-event cluster that is performed at the end of an activity, because this sub-event cluster signifies the end of a sequence and announces the start of a new sequence of sub-event clusters. We therefore introduce a third variable, the finished state variable f_t, which is used as a binary indicator to indicate the bottom layer has finished its sequence. The details of this model can be found in Van Kasteren et al. [7].

The observations are modeled using the Gaussian distribution, in which each sub-event cluster is associated with a single Gaussian $p(\vec{x}_t \mid y_t = k, z_t = l) = \mathcal{N}(x_t \mid \mu_{kl}, \Sigma_k)$. Note that covariance matrix Σ_k only has a subscript k, meaning that we have a different covariance matrix for each sound event, but that the covariance matrix for different sub-event clusters is shared among the sub-event clusters for a particular sound event k. When using multiple sub-event clusters per sound event this closely corresponds to a Gaussians mixture model in which each sub-event cluster is equivalent to one of the mixtures in the mixture model. The difference with a traditional Gaussians mixture model is that our model also models the probability to transition from one sub-event to the next sub-event (i.e. from one mixture to the next mixture). This model assumes that the audio features of a sub-event are relatively constant (with some Gaussian noise added) throughout the duration of the sub-event. Significant differences in the audio features of a sound event are modeled by dividing the sound event into consecutive sub-event clusters. The ideal number of sub-event clusters to use is determined through experimentation on a validation set.

4. META-CLASSIFIER

In the context of supervised learning, algorithms are typically searching through a hypothesis space to find a hypothesis that can be used to make good predictions for the particular problem at hand. In the context of machine learning, ensemble is a technique for combining numerous weak learners in an attempt to produce a strong learner. An ensemble is also a supervised learning method, since it has the capacity to be trained and then used to perform predictions. As such, the ensemble also represents a single hypothesis in the solution space. However, this hypothesis is not necessarily contained within the space of the models used to construct the ensemble. Therefore ensembles typically have more flexibility in the functions they can represent, which can result in a reduction of model bias [8]. Considering the typical bias-variance decomposition and the bias-variance tradeoff, an increase in model complexity is often associated with an increase in variance, since the more complex model is potentially more prone to overfitting the training data. This effect is encountered to a different extent in various ensemble methods, but some of them are quite robust since they are specifically designed to avoid it (e.g. Bagging).

It can be shown empirically that having a set of models which when used independently provide good predictions and their diversity (i.e. disagreement) is high is a good basis for ensemble methods [9]. Thus many ensemble methods are often actively seeking to promote diversity among the models they combine by changing the training set of each model (i.e. through resampling) or the used sub-set

of features, or both [10]. The simpler of the ensemble methods are applied only between different models of the same type. After some experimentation with the provided datasets we have selected Bagging with decision trees as the method of choice for the basic ensemble part of the approach. Bagging, an abbreviation for Bootstrap aggregating, involves training each model in the ensemble using a randomly selected subset of the training set in order to promote model diversity and combining predictions with an equal weight voting scheme (so as to avoid overfitting). The only level-1 discriminative model selected was Random Forest, an implementation of Bagging with decision trees as basic learners (level-0). However, the considered approach does not limit itself to discriminative models, as these models are not particularly well tailored for the given type of problem since no temporal ”state” information is considered. Therefore, generative models such as HHMM were also deployed (at level-1), as described previously.

The final decision is taken by a metaclassfier (level-2) that uses as meta-features the predictions of the level-1 classifiers, both discriminative (Random Forest) and generative (one HHMM model). Since audio data have a strong temporal character a model that is designed to exploit this property is a good candidate for the metaclassfier. In our case we used a second instance of HHMM as a metaclassifier (experimentation with different methods at the level-2 which do not consider temporal information resulted in more fragmented output and poorer performance). The selected approach, a variation of Stacking, is bringing the ensemble methods to the final point of exploitation, since it can be used to combine completely different models (as opposed to e.g. Bagging and Boosting that are applied to the same basic model template, such as decision trees, neural networks, etc.). The performance of the metaclassifier is significantly better than the best level-1 model can achieve on its own but tweaking all the parameters of each model requires some effort.

5. RESULTS

Table 5 shows the results of the HHMM meta-classifier averaged over the three development files and both annotators. Our model performs best on script01 with an F-measure of 74% with annotator ’sid’ as ground truth (Figure 2). For all three development files our model performs better on annotator ’sid’ than annotator ’bdm’. There is one class that is never recognized, switch, and pendrop is also one of the more difficult classes to detect.

Our experiments show that the best results are obtained using audio features that were calculated using a rectangle window type and using a single sub-event cluster per sound event. This is surprising since with a single sub-event cluster the model does not ben-

Table 2: Results of the HHMM meta-classifier with one sub-event cluster per sound event and a rectangular window for the audio features. The metrics are averaged over development files script01, script02, and script03, and both annotators.

	Evalution method		
Metrics	Event-based	Class-wise event-based	Frame-based
R	42.21	42.38	45.26
P	55.65	43.66	71.13
***F*-Measure**	**47.53**	**41.28**	**54.41**
AEER	1.2449	1.1634	0.8784
Offset *R*	37.07	36.75	–
Offset *P*	48.93	38.82	–
Offset *F*-Meas.	41.74	36.14	–
Offset *AEER*	1.3992	1.3322	–

efit from modeling transitions between sub-events and the resulting model corresponds closely to a standard hidden Markov model (HMM). We have compared the performance of this model with an HMM using a Gaussian observation model and the hierarchical model using a single sub-event cluster significantly outperforms the HMM. The difference between the hierarchical model using a single sub-event cluster and the HMM is that the hierarchical model explicitly models the finishing of a state. We suspect the decrease in performance when using additional sub-event clusters is due to the increase in the number of parameters that need to be estimated (the number of parameters significantly increases because many of the factors of our model depend on the sub-event cluster used). Another possible explanation is that the expectation-maximization (EM) algorithm used to cluster the sub-event clusters easily gets stuck in a local maximum even though a better fitting global maximum exists. A different initialization of the model parameters when starting the EM algorithm could remedy this problem, but we were unable to find an initialization method that consistently outperforms the single sub-event cluster model.

6. ACKNOWLEDGMENT

We are grateful to Christian Debes and Roel Heremans for their contribution.

7. REFERENCES

[1] S. Chu, S. Narayanan, and C.-C. Kuo, “Environmental sound recognition with time-frequency audio features,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1142–1158, 2009.

[2] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, “Scream and gunshot detection and localization for audio-surveillance systems,” in *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007, pp. 21–26.

[3] K. Wojcicki, “HTK MFCC,” 2011.

[4] VOICEBOX, http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html.

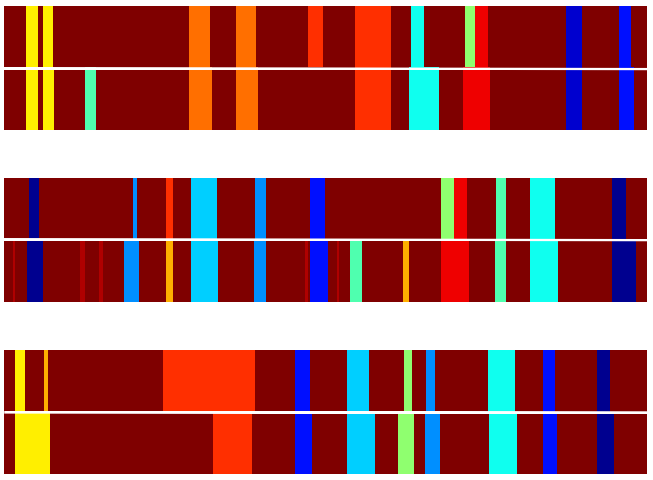

Figure 2: The top bar shows the classification result of our model for script01 and the bottom bar (seperated by the white line) the annotation of ’sid’. The various colors correspond to the different classes of sound events. The short annotations in dark red are switches and missed by our model. In addition we miss a knock at the beginning of the file and misclassify and mis a pendrop. The rest of the classes are well detected, only speech is partly confused as laughter.

[5] P. Pudil, J. Novovičová, and J. Kittler, “Floating search methods in feature selection,” *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, 1994.

[6] J. D. Krijnders, M. E. Niessen, and T. C. Andringa, “Sound event recognition through expectancy-based evaluation of signal-driven hypotheses,” *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1552–1559, 2010.

[7] T. L. M. Van Kasteren, G. Englebienne, and B. J. A. Kröse, “Hierarchical activity recognition using automatically clustered actions,” in *Proceedings of the Second International Conference on Ambient Intelligence*, 2011, pp. 82–91.

[8] D. Wolpert, “Stacked generalization,” *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[9] L. Kuncheva and C. Whitaker, “Measures of diversity in classifier ensembles,” *Machine Learning*, vol. 51, pp. 181–207, 2003.

[10] G. Brown, J. Wyatt, R. Harris, and X. Yao, “Diversity creation methods: a survey and categorisation,” *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.