An Approach for Self-Training Audio Event Detectors Using Web Data

Benjamin Elizalde[†§], Ankit Shah[*§], Siddharth Dalmia [†§],
Min Hun Lee[†§], Rohan Badlani[‡§], Anurag Kumar[†§], Bhiksha Raj [†] and Ian Lane [†]
Department of Electrical and Computer Engineering,
& Department of Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA [†]
Department of Electronics and Communication, NITK Surathkal, India [*]
Department of Computer Science, BITS Pilani, India [‡]
Email: [†]bmartin1@andrew.cmu.edu, [*]ankit.tronix@gmail.com, [†]sdalmia@andrew.cmu.edu, [†]mhlee@cmu.edu,
[‡]rohan.badlani@gmail.com, [†]alnu@andrew.cmu.edu, [†]bhiksha@cs.cmu.edu, [†]lane@cs.cmu.edu

***Abstract*—Audio Event Detection (AED) aims to recognize sounds within audio and video recordings. AED employs machine learning algorithms commonly trained and tested on annotated datasets. However, available datasets are limited in number of samples and hence it is difficult to model acoustic diversity. Therefore, we propose combining labeled audio from a dataset and unlabeled audio from the web to improve the sound models. The audio event detectors are trained on the labeled audio and ran on the unlabeled audio downloaded from YouTube. Whenever the detectors recognized any of the known sounds with high confidence, the unlabeled audio was use to re-train the detectors. The performance of the re-trained detectors is compared to the one from the original detectors using the annotated test set. Results showed an improvement of the AED, and uncovered challenges of using web audio from videos.**

I. Introduction and Related Work

Sounds are essential to how humans perceive and interact with the world. Audio content is captured in recordings and shared on the web on a minute-by-minute basis. Academia and industry exploits this acoustic information throughout multiple applications. The dominant application is multimedia video content analysis, where audio is combined with images and text [1], [2] to index, search and retrieve videos. Another task is human-robot interaction [3], [4], where sounds complement speech as non-verbal communication. Recently, a growing application is in smart cities [5], where sounds are used to detect sources of noise pollution. All of these applications rely on Audio Event Detection (AED) to recognize the occurrence of sounds within audio and video recordings.

The related work on AED has mainly focused on using available datasets to train machine learning models in a supervised manner [6], [5], [7], [8], [9]. However, the largest data set, ESC-50 [8], contains only 40 samples per class. The numbers strongly contrast with Imagenet, the computer vision counterpart, which has hundreds of samples per class. Hence, training a model which reflects the acoustic diversity of an audio event class is limited. The common solution is to have humans annotating more data. However, the process is costly and slow and thus, other solutions should be explored.

[§] First six authors contributed equally.

Another solution is to combine the small amount of labeled data with a large amount of unlabeled data. A particular method is semi-supervised self-training, which is an algorithm that iteratively re-trains a model. First, the model is trained using the labeled data set. Then, at each iteration and under a certain criteria, a portion of the unlabeled set could be labeled as any of the known classes. Lastly, using the newly labeled data, the model is re-trained. This approach has been explored for audio events in two papers [10], [11]. Particularly in [10], the authors collected 17,000 labeled audio-only recordings from *FindSounds.com*. Two thirds were used to train and test a classifier and the rest was treated as unlabeled audio for re-training. The result was an improvement of 1.4% precision over the baseline, suggesting a valid alternative to improve models. Moreover, the authors pointed out the challenges of utilizing audio-only web recordings.

In our paper, we followed a similar framework of semi-supervised self-learning, but with the following differences:

- We employed UrbanSounds8k as the labeled set for training and testing. However, we collected YouTube videos as the unlabeled set for re-training, creating mismatch conditions.
- For re-training, we used 30 times more audio files.
- The unlabeled audio is extracted from videos as opposed to audio-only recordings. Hence, posing challenges during the collection process. For instance, it is not possible to guarantee that the YouTube audio will actually contain any of the sounds.

The paper is structured as follows, in Section II we describe the flow of our self-training approach for sound detectors. Within this Section we describe the sound event dataset and YouTube video collection process and how we pre-processed the audio recordings. Then, we explain how we trained our two machine learning based detectors to compare performance. In Section III we compare the baseline performance and the self-training performance obtained with different techniques.

ISBN 978-0-9928626-7-1

II. Semi-Supervised Self-Training of Audio Event Detectors

Semi-supervised self-training is an algorithm that iteratively re-trains a model and our particular framework is illustrated in Figure 1. First, using the labeled dataset, the ten class detectors are trained and tested to compute a baseline performance. Second, the unlabeled data is run by the detectors to obtain a class label with its corresponding confidence score. Third, we applied a threshold based on the confidence score to determine candidates for self-training the detectors. Fourth, the detectors are re-trained and again tested on the labeled data to compute the new performance and compare it with the previous. Lastly, steps two, three and four are performed iteratively until the performance converges.

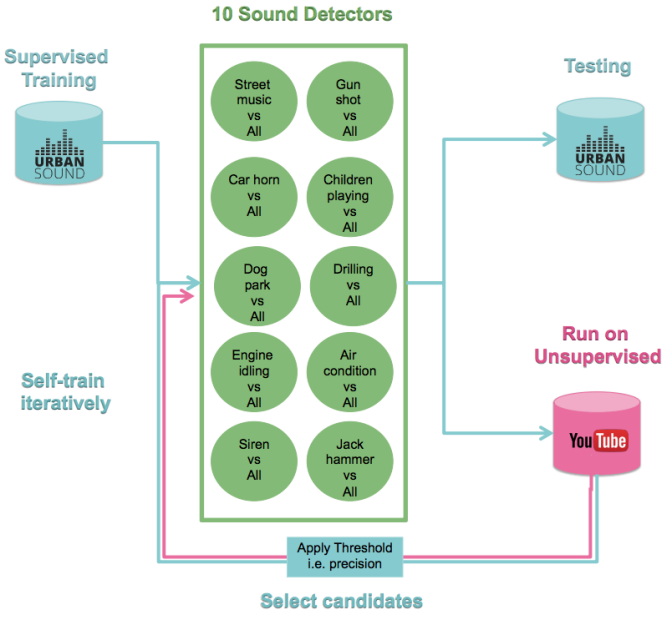

Fig. 1. Flow of the semi-supervised self-training of Audio Event Detection. Most of the tuning that improved the performance of the re-trained detectors happened in the selection of candidates.

A. Datasets: Labeled and Unlabeled

Labeled Data (Training and Testing): UrbanSound8K

The UrbanSound8K (US8K) dataset [5] has 10 classes: *air conditioner, car horn, children playing, dog bark, and street music, gun shot, drilling, engine idling, siren, jackhammer*. The content of the audio may have other overlapping sounds and the target sound may occur in the background or in the foreground. The dataset has about 8,732 audio segments of 3.5 sec average duration. These files are distributed into 10 stratified cross-validation folds.

Unlabeled Data (Self-Training): YouTube videos

The unlabeled audio comes from YouTube videos and videos are the largest source of audio. The website was chosen because it offers a wide diversity of class samples. The soundtracks of the videos were crawled and downloaded using the Pafy API[1]. The audio roughly corresponds to the 10 classes from US8k. The acoustic content is unstructured, and commonly the target sound is occluded by multiple factors such as noise, overlap with other sounds, and channel effects. The web set has 200,000 segments of 3.5 seconds. We converted all of the audio files into raw 16 bit encoding, mono-channel, and 16 kHz sampling rate.

Challenges of Unlabeled Data Collection

Audio from videos poses collection challenges. YouTube contains years of videos and in order to process and evaluate audio containing the target sound, the query should serve as a filter. Hence, the query formulation aims to filter in videos roughly matching the ten classes in US8K. Typing a query composed by a noun such as *air conditioner* will not necessarily fetch a video containing such sound event. This happens because the associated tags and metadata are mainly inspired by the video's visual content; contrary to what happens in audio-only websites such as *freesounds.org*. Therefore, we modified the query to be a combination of keywords: "*<audio event label>*sound", for example,"air conditioner sound". Although the results empirically improved, another issue was that the audio event was not guaranteed to occur and if present it most likely occurred with a short duration within whole recording. Therefore, we restricted the video length to be larger than five seconds and shorter than ten minutes to reduce the amount of irrelevant audio.

B. Data Preparation

Extracting Low-level Features: MFCCs

The Mel Frequency Cepstral Coefficients (MFCCs) have been widely used in audio event detection [12], [5], [8]. The parameters are standard, such as 10 ms shifts, window of 25 ms and 20 cepstral coefficients including delta and double-delta (time dynamics) for a total of 60 coefficients for each time window or vector.

Extracting Intermediate Features: BoAWs

An effective approach for characterizing audio events is the Bag-of-Audio-Words (BoAWs) feature representation, which is usually built over low-level features such as MFCCs. The method we followed to compute BoAWs features is broadly illustrated in Figure 2 and detailed in these papers [13], [14]. In the first step, we put all the MFCCs in the training set together. In the second step, we learn an "audio vocabulary" by grouping the features into "audio words". In contrast to conventional approaches which uses clustering for grouping words, our method adapts the MFCCs to a Gaussian Mixture Model (GMM) using Expectation Maximization where each mixture represents a word. The third step is quantization, which uses the created vocabulary to turn the MFCC matrix of a given recording into a BoAW histogram-vector of the size of the vocabulary. The conventional quantization process computes the distance of each MFCCs frame to all the audio words and sums the value of one only on the histogram bin that

[1]https://pypi.python.org/pypi/pafy

ISBN 978-0-9928626-7-1 © EURASIP 2017

corresponds to the closest word. However, our approach uses soft-quantization, which sums probabilities for all the words.

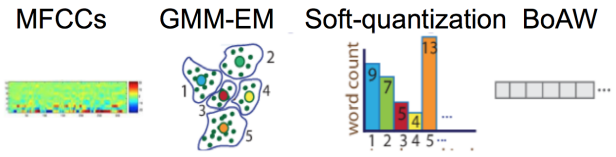

Fig. 2. Process of computing our BoW features. The idea behind GMM-based representation is to capture the distribution of MFCC vectors of a recording over the GMM components.

Formally, let the MFCC vectors of a recording be represented as $\vec{x}_t$. $\vec{x}_t$ is t^{th} D dimensional MFCC frame, $t = 1\ to\ T$. Here, the GMM $\mathcal{G} = \{w_k, N(\vec{\mu}_k, \Sigma_k), k = 1\ to\ M\}$, is learned over the MFCCs of training data. w_k, $\vec{\mu}_k$ and Σ_k are the mixture weight, mean and co-variance parameters respectively, of the k^{th} Gaussian in $\mathcal{G}$. We train GMM with diagonal co-variance matrices. To obtain the bag of audio word feature representation for any given recording, we first compute the probabilistic assignment to k^{th} Gaussian for each MFCC frame of that recording as in Equation 1. This soft assignment is then summed and normalized over all MFCC frames for k^{th} Gaussian as in Eq 2.

$$Pr(k|\vec{x}_t) = \frac{w_k N(\vec{x}_t; \vec{\mu}_k, \Sigma_k)}{\sum_{j=1}^{M} w_j N(\vec{x}_t; \vec{\mu}_k, \Sigma_k)} \quad (1)$$

$$P(k) = \frac{1}{T} \sum_{i=1}^{T} Pr(k|\vec{x}_t) \quad (2)$$

The final *soft-count* histogram feature representation, represented as $\vec{\alpha}$ is $\vec{\alpha}^M = [P(1), ..P(k)..P(M)]^T$. $\vec{\alpha}^M$ features are an M-dimensional (M=128) feature representation for any given recording. During testing, the BoAW features are computed in a similar manner however, using the created vocabulary from training.

1) Training Detectors: Positive and Negative Classes: We chose detectors–binary classifiers, because are able to recognize the presence or absence of a particular audio event in a recording. The binary setup also aims to simulate the imbalance ratio of small amount of target sound vs a large amount of non-target sounds, which is common in web retrieval tasks. The audio samples belonging to the target class are referred to as positives and those samples not belonging to the target are referred to as negatives. Each of the ten detectors is trained with both, a positive and a negative class. Positive contains class samples and negative contains samples from the rest of the classes. For instance, the detector for *jackhammer* has all the samples corresponding to *jackhammer* as positives and all the samples corresponding to the other 9 sounds as negatives.

2) Training Detectors: SVM: One round of experiments was performed with Support Vector Machines (SVMs) because SVMs have been widely explored for sound events [6]. The ten SVM-based detectors used linear decision boundaries to fit the data and were trained with the intermediate features. To relax the constraints defining the margin of the decision boundary, the parameter "C" was tuned and set to 0.01. Then, the trained detectors were evaluated using the test set. Although conventional SVMs could employ other techniques to allow non-linear decision boundaries, the problem happens when new audio segments are added for re-training. Each iteration means that the SVM has to be re-trained from scratch using all the train data. The consequence is a bottleneck, which worsens as more segments and iterations are added.

3) Training Detectors: NN: Considering the previous issue, the second round of experiments was performed with Neural Networks (NNs). The NNs are more suitable for the iterative nature of self-training. For example, in order to add a new audio segment for training, the NN does not need to be re-trained from scratch and a quick updating process suffice. The NN-based detectors are also binary classifiers. More precisely, for the NN we utilized a Multi-Layer Perceptron (MLP), with tuned hyper-parameters such as number of layers, neurons, activation function, regularization and loss function. The final architecture consisted of an input of size 128–BoW features dimensionality, one hidden layer of 100 neurons, and two output units– class or not class. The activation function was "tanh", the regularization method was dropout (p=0.5), the loss function was cross-entropy and the number of epochs was 10. Then, the trained detectors were evaluated using the labeled test set.

III. Experiments and Evaluation of Methods

A. Computing the Baseline Performance and Running Detectors on Unlabeled Data

The initial performance computed by our two machine learning algorithms defined the baseline to improve after self-training. The detectors used the labeled data from US8k, which comes divided in 10 stratified folds. We used 9 folds as training data and tested on the left-out fold. This is done in 10 different ways, resulting in 100 runs for all the 10 events classes and 10 folds. We evaluated our detectors using average precision as we wanted to detect reliable positive or negative samples. For every class, the average precision (AP) over each fold is computed, as well as the mean AP across all folds referred as Mean AP. Afterwards, the detectors where run on the unlabeled dataset to obtain confidence scores and labels for each of the 200,000 segments. Note that the unlabeled data was carefully handled to be consistent with the 10 fold cross-validation setup. For example, the detectors trained using the first 9 folds may not yield the same performance as the detectors trained with any other fold combination.

B. Selecting Candidates and Self-Training Detectors

We employed a high confidence threshold to select audio segments as candidates for self-training. The candidates were used for self-training the detectors in combination to the supervised audio segments. Once the detectors were re-trained, they were ran on the supervised test set and their performance was computed. The Mean AP value was compared with the

ISBN 978-0-9928626-7-1 © EURASIP 2017

baseline and the whole process was repeated iteratively until the Mean AP converged.

A key step in the self-training process is the selection of candidates. We tried three main approaches: Detector's output scores, precision and clarity index.

Score-based Under this approach, the output of the detector is a probability score that can be interpreted as a confidence value and has been used for self-training in the paper [10]. A score threshold of greater or equal than 0.95 was selected to filter in any segment, where 0 means the lowest confidence and 1 means the strongest confidence.

Precision High precision means that the detector returned more relevant results than irrelevant ones. A precision threshold of greater or equal than 0.95 was set. The value range is the same as the score-based.

Clarity Index Clarity Index (CI), based on the paper [15], aims to determine those segments that are the most confusing for the detector. CI is based on two losses called *relevance loss* and *irrelevance loss*. To understand these losses let us assume that the training data is $\mathcal{D} = \{(x_1, y_1), (x_2, y_2)..., (x_n, y_n)\}$ and the detector mapping function is denoted by f. Let x^u be an unlabeled data point. The Relevance Loss (RL) and the Irrelevance Loss (IL) are defined as

$$RL(x^u, f) = \frac{1}{|\mathcal{D}_0|} \sum_{x_i \in \mathcal{D}_0} I(f(x_i) - f(x^u)) \tag{3}$$

$$IL(x^u, f) = \frac{1}{|\mathcal{D}_1|} \sum_{x_i \in \mathcal{D}_1} I(f(x^u) - f(x_i)) \tag{4}$$

The relevance loss is expected to be low if x^u is relevant (positive) and irrelevance loss is expected to be low irrelevant (negative). The difference of the two losses $CI = IL - RL$ is expected to be high (close to 1) for positive instances and low (close to -1) for negative instance. Overall, the CI helps us rank unlabeled segments to choose better segments for self-training. Higher CI implies that X^u is more likely to be positive. An unlabeled point with very high CI would have outscored a large number of training points and hence is expected to be positive. Similarly, lower CI implies the instance is most likely negative.

IV. Results and Discussion

A. Baseline

The NN outperformed the SVM by an absolute 8.5% in the baseline performance. The Mean AP score was 57.8% for SVM and 66.3% for NN and are shown in Table I. One reason for to justify the better performance of the NN, is that it employed nonlinear decision boundaries to fit the data unlike the SVM, which used linear boundaries. As mentioned before, the SVM can also support nonlinear decision boundaries by using kernels, but the computation time was an issue for processing 200,000 segments for the 10 fold combinations, and the re-training.

B. Self-Training

The main results of this paper are the improvement gain by self-training shown in Table I and labeled as SVM Best and NN Best. The overall Mean AP improvement was 1.2% for both classifiers. Except for SVM's *dog barking*, all the audio events improved their performance. Particularly, *air conditioning* and *jackhammer* benefited the most with about 3%. More importantly, the performance did not degrade, which is expected to happen when audio that not belonging to the target class is added by re-training. The SVM Best and NN Best results correspond to different threshold types– CI and Pecision respectively. The performance of the three threshold types was similar (0.5%-1.4%) and we cannot say that one should be preferred.

For the three threshold types, tuning affected differently the overall selection of candidates and the detection performance. The number of candidates varied between class from 0 to 2,000 on each iteration. In general, stricter values (greater than 0.9) reduced the number of candidates to two digits. Regarding the detection performance, stricter values (greater than 0.95) and loose values (approximately 0.5) degraded Mean AP, but values close to 0.9 yielded the reported gain. The threshold also defined the number of iterations the algorithm took to converged or stop improving. For our value of 0.9, our algorithm iterated three times. Afterwards, the Mean AP performance converged and then slowly decreased. In general, most of our experiments degraded its performance after several iterations. Two possible explanation are the mismatch conditions and the lack of useful files for self-training.

Mismatch conditions are unavoidable if web audio is intended to be exploited through semi-supervised approaches. There is no control over the recording methods for unlabeled web audio and thus it will most likely be different than the control methods from labeled datasets. In our case, the dataset US8k has different collection methods and acoustic characteristics, which do not match the user-generated YouTube audio. In our experiments, the detectors were self-trained using only the newly labeled "positive" segments from YouTube. After each iteration, more and more YouTube data was added to the detectors on the positive category but not on the negative. However, the improvement was limited and often degraded. After inspecting some of the rejected files, it seemed that the detectors were discriminating YouTube vs non-YouTube audio, rather than positive vs negative. On the contrary, when "positive and negative" segments were added, the performance improved.

A manual inspection on some of the candidates helped us better understand the audio content used to re-train the detectors. Thumbnails examples are in Figure 3, illustrating interesting cases. For instance, some videos may have the presence of the sound even though the image didn't corresponded. The first thumbnail-video had the *siren* sound, but the image in the video was just a radio-like box. Another example was when sounds were acoustically similar but semantically different. The third thumbnail-video showed a scene from the

ISBN 978-0-9928626-7-1 © EURASIP 2017

Category	SVM Baseline	SVM Best	NN Baseline	NN Best
air_conditioner	39.3	45.1	49.9	53.2
car_horn	52.4	53.0	51.6	52.8
children_playing	53.8	54.3	65.1	65.2
dog_bark	76.2	75.9	81.7	82.0
drilling	56.6	57.2	63.4	63.0
engine_idling	53.8	54.1	68.0	69.8
gun_shot	67.8	69.1	80.4	81.9
jackhammer	60.2	62.3	63.7	66.2
siren	72.2	72.8	80.2	80.4
street_music	46.0	46.4	58.5	59.0
Mean AP	57.8	59.0	66.3	67.5

TABLE I
THE TABLE SHOWS THE CLASS PRECISION AND FOLD AVERAGE PRECISION (MEAN AP). THE MEAN AP OF THE SVM AND NN BASELINES WAS IMPROVED THROUGH SELF-TRAINING. THE SVM BEST CORRESPONDS TO CLARITY INDEX THRESHOLD AND NN BEST CORRESPONDS TO PRECISION THRESHOLD.

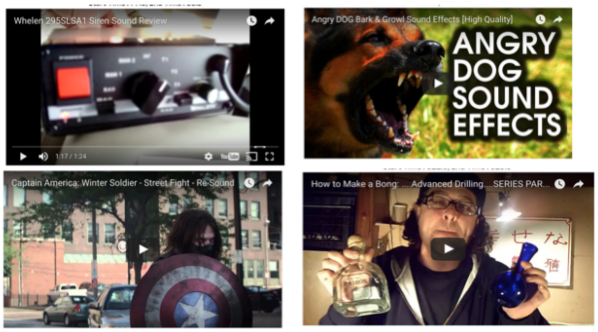

Fig. 3. Manual inspection of selected candidates from Siren, Dog bark, Air Conditioner and Drilling.

movie "Captain America", where the audio was similar to "air conditioner", but there was no such item. These examples does not necessarily degrade the quality of the detector as shown in [16]. In a similar manner, *gun shot* had, among some of the candidates, object banging sounds.

V. LIMITATIONS AND FUTURE WORK

Classifier bias Semi-supervised approaches have limitations related to what extent they can help, as discussed in [17]. Especially, self-training has an inherent detector bias issue which happens when a detector is trained with an initial set of data. The detector then, is ran on the unlabeled data and the confidence score depends on the initial model. Once we add new segments, we are enforcing the acoustic characteristics of the previous model and not necessarily making our models more robust. Addressing the issue was out of scope, but could be a reason for the fast convergence in our results.

Threshold type The set of thresholds utilized are a reasonable approach supported in the literature. However, a more elaborated objective function should be considered to better select candidates.

VI. CONCLUSIONS

In this work we proposed a framework of semi-supervised self-training of audio event detectors, where the detectors were trained with the annotated US8K dataset, and the self-training employed unlabeled audio from YouTube videos. The NN detectors yielded a higher baseline performance than SVM. Both detectors and almost all the classes benefited from self-training. Despite the audio mismatch conditions and thepossibility of having few or no target sounds to be candidates, the performance after self-training did not degrade. Further exploration to select candidates offers a valuable opportunity. Unlabeled audio from videos can help audio event detection.

REFERENCES

[1] P. Schäuble, *Multimedia information retrieval: content-based information retrieval from large text and audio databases*. Springer Science & Business Media, 2012, vol. 397.
[2] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain, "Content-based multimedia information retrieval: State of the art and challenges," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 2, no. 1, pp. 1–19, 2006.
[3] J. Maxime, X. Alameda-Pineda, L. Girin, and R. Horaud, "Sound representation and classification benchmark for domestic robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6285–6292.
[4] M. Janvier, X. Alameda-Pineda, L. Girinz, and R. Horaud, "Sound-event recognition with a companion humanoid," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE, 2012, pp. 104–111.
[5] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *22st ACM International Conference on Multimedia (ACM-MM'14)*, Orlando, FL, USA, Nov. 2014.
[6] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Transactions on Multimedia*, vol. 17, no. 10, pp. 1733–1746, 2015.
[7] M. Ravanelli, B. Elizalde, K. Ni, and G. Friedland, "Audio concept classification with hierarchical deep neural networks," in *Proceedings of EUSIPCO*, 2014.
[8] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2015, pp. 1–6.
[9] T. Virtanen, A. Mesaros, T. Heittola, M. Plumbley, P. Foster, E. Benetos, and M. Lagrange, *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016)*. Tampere University of Technology. Department of Signal Processing, 2016.
[10] W. Han, E. Coutinho, H. Ruan, H. Li, B. Schuller, X. Yu, and X. Zhu, "Semi-supervised active learning for sound classification in hybrid learning environments," *PloS one*, vol. 11, no. 9, p. e0162075, 2016.
[11] Z. Zhang and B. Schuller, "Semi-supervised learning helps in sound event classification," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 333–336.
[12] F. Metze, S. Rawat, and Y. Wang, "Improved audio features for large-scale multimedia event detection," in *Multimedia and Expo (ICME), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–6.
[13] F.-F. Li and P. Perona, "The perceived position of moving objects: Transcranial magnetic stimulation of area MT+ reduces the flash-lag effect," in *IEEE CVPR*, vol. 2, 2005.
[14] B. Elizalde, A. Kumar, A. Shah, R. Badlani, E. Vincent, B. Raj, and I. Lane, "Experiments on the dcase challenge 2016: Acoustic scene classification and sound event detection in real life recording," in *DCASE2016 Workshop on Detection and Classification of Acoustic Scenes and Events*, 2016.
[15] T. S. Huang, C. K. Dagli, S. Rajaram, E. Y. Chang, M. I. Mandel, G. E. Poliner, and D. P. Ellis, "Active learning for interactive multimedia retrieval," *Proceedings of the IEEE*, vol. 96, no. 4, pp. 648–667, 2008.
[16] B. Elizalde, M. Ravanelli, and G. Friedland, "Audio concept ranking for video event detection on user-generated content," *in Multimedia (SLAM 2013)*.
[17] A. Singh, R. Nowak, and X. Zhu, "Unlabeled data: Now it helps, now it doesn't," in *Advances in neural information processing systems*, 2009, pp. 1513–1520.

ISBN 978-0-9928626-7-1 © EURASIP 2017