CQT-BASED CONVOLUTIONAL NEURAL NETWORKS FOR AUDIO SCENE CLASSIFICATION AND DOMESTIC AUDIO TAGGING

Thomas Lidy

Vienna University of Technology
Institute of Software Technology
Vienna, Austria
lidy@ifs.tuwien.ac.at

Alexander Schindler

Austrian Institute of Technology
Digital Safety and Security
Vienna, Austria
alexander.schindler@ait.ac.at

ABSTRACT

For the DCASE 2016 challenge on detection and classification of acoustic scenes and events we submitted a parallel Convolutional Neural Network architecture for the tasks of classifying acoustic scenes and urban sound scapes (task 1) and domestic audio tagging (task 4). A popular choice for input to a Convolutional Neural Network in audio classification problems are Mel-transformed spectrograms. We, however, found that a Constant-Q-transformed input improves results. Furthermore, we evaluated critical parameters such as the number of necessary bands and filter sizes in a Convolutional Neural Network. Finally, we propose a parallel (graph-based) neural network architecture, which captures relevant audio characteristics both in time and in frequency, and submitted it to the DCASE 2016 tasks 1 and 4. For the acoustic scenes classification task our approach scored 80.25 % accuracy on the development set, a 10.7 % relative improvement of the DCASE baseline system [1], and achieved 83.3 % on the evaluation set (rank 14 of 35) in the challenge. On the domestic audio tagging task, our approach is the winning algorithm (rank 1 of 9) with 16.6 % equal error rate.

Index Terms— Deep Learning, Constant-Q-Transform, Convolutional Neural Networks, Audio Event Classification, Audio Tagging

1. INTRODUCTION

Recent advances with Deep Learning approaches in image retrieval have fueled the interest as well in audio-based tasks such as speech recognition and music information retrieval. A particular sub-task in the audio domain is the detection and classification of acoustic sound events and scenes, such as the recognition of urban city sounds, vehicles, or life forms, such as birds.[1] The IEEE AASP Challenge DCASE 2016 is a benchmarking challenge for the "Detection and Classification of Acoustic Scenes and Events". It comprises four tasks, which include acoustic scene classification in urban environments (task 1), sound event detection in synthetic and real audio (tasks 2 and 3) and audio tagging of human activity in a domestic environment (task 4).

We submitted our system (with slight differences) for tasks 1 and 4, which are focusing on classification and tagging of sound files. We did not participate in tasks 2 and 3 on detection of events in audio streams. The goal of task 1 was to classify test recordings into *one* of predefined classes that characterizes the environment in which it was recorded, for example "metro station", "beach", "bus", etc. [1]. The goal of task 4 was to classify sound snippets into multiple (none, one, or more) of given tags related to domestic environments: child speech, adult male / female speech, video game, percussive sounds, broadband noise from household appliances, etc.

A popular choice for applying Deep Learning to audio is the use of Convolutional Neural Networks (CNN). The apparent method is to use an audio spectrogram (derived from the Fast Fourier Transform and/or other transformations) as an input to a CNN and to apply convolving filter kernels that extract patterns in 2D, similar as being done for image analysis and object recognition. Yet, audio has a fundamental difference to images: The two axes in a spectrogram do not represent a spatial coherence of visual data, but exhibit two completely different semantics: time and frequency. Approaches have been reported applying convolutions directly on the wave form (i.e. time domain) data, however with not fully satisfying success so far [2]. Therefore, typically audio is transformed into the time-frequency domain, with some (optional) further processing steps, such as the Mel transform and/or a Log transform.

In an earlier publication related to our participation in the MIREX benchmarking contest ("Music Information Retrieval Evaluation eXchange") [3] we have shown the successful application of Mel-spectrogram based Convolutional Neural Networks on music/speech classification (discrimination) [4]. Our approach won the MIREX 2015 music/speech classification task with 99.73 % accuracy.[2] As our background is the recognition of semantic high-level concepts in music (e.g. genre, or mood, c.f. [5, 6]), and Mel Frequency Cepstral Coefficients (MFCCs) are used in both music and speech recognition, the use of the Mel scale was an evident choice.

However, we realized in the course of developing a solution for the task of classifying acoustic scenes from urban sounds that an adaptation was necessary to cover activity in very low and very high frequencies that may or may not be rhythmical. Our research and experimentation led us to applying the Constant-Q-Transform (CQT), which captures low and mid-to-low frequencies better than the Mel scale. We also did a number of alterations in the architecture of the Convolutional Neural Network. Earlier research [7] showed that a combination of a CNN that captures temporal information and another one that captures timbral relations in the frequency domain is a promising approach for music genre recognition, in which typically both tempo and timbre (e.g. particular instruments) play an important role. Again, this had to be adapted for the classification tasks in DCASE 2016.

This abstract accompanying our submission to DCASE 2016

[1] http://www.imageclef.org/lifeclef/2016/bird

[2] http://www.music-ir.org/mirex/wiki/2015:Music/Speech_Classification_and_Detection_Results

is an extension to our paper submitted to the DCASE 2016 workshop [8]. In the workshop paper, we focused on task 1 only. For related work, please refer to the workshop paper [8]. Section 2 describes our method in detail and provides a few alterations not described in the workshop paper. Section 3 describes task 1 on acoustic scene classification, the data set and our results on the development set. Section 4 describes task 4 on domestic audio tagging, the dataset and our approach. Finally, in Section 5 we summarize our conclusions on the presented approach.

2. METHOD AND SYSTEM

For both tasks – acoustic scene classification and domestic audio tagging – we use Convolutional Neural Networks, which we trained on CQT-transformed audio input. We describe these two parts in more detail in this section.

2.1. Audio Preprocessing: CQT

Before being input to the neural network, a few preprocessing steps are carried out on the original audio which are depicted in Figure 1. First of all, a stereo audio signal is transformed to mono by averaging the two channels. Then, we apply the Constant-Q-Transform. The Constant-Q-Transform (CQT) is a time-frequency representation where the frequency bins are geometrically spaced and the so called Q-factors (ratios of the center frequencies to bandwidths) of all bins are equal [9]. The CQT is essentially a wavelet transform, which means that the frequency resolution is better for low frequencies and the time resolution is better for high frequencies. The CQT is motivated from both musical and perceptual viewpoints: The human auditory system is approximately "constant Q" in most of the audible frequency range, and also the fundamental frequencies of the tones in Western music are geometrically spaced along the standard 12-tone scale [9]. Thus, the CQT typically captures 84 bands covering 7 octaves of 12 semi-tones each, however, it allows to set a different number of bands and also a higher number of bands per octave. In our approach, we use a total number of 80 bands, with the standard setting of 12 bands per octave, meaning that the 4 highest bands will be cut off. We use a hop length of 512 samples (similar as it is typically used when a fast Fourier transform is applied on 1024 samples long windows to calculate a spectrogram), i.e. a CQT is computed every 512 samples (11.6 milliseconds). Following the CQT, we perform a Log_{10} transform of all values derived from the CQT. This process is performed on chunks, or segments, of 41472 samples length (0.94 seconds), resulting in 82 CQT frames (analogously to FFT frames). The idea is to process a multitude of short-term segments from an audio example to be learned by the neural network. In this case, a 30 second input file results in 31 CQT excerpts of shape 80 bands × 82 frames.

In our workshop paper [8] we show that using the CQT instead of the Mel-transform has a beneficial impact. In our experiments, the best result is achieved with 80 CQT bands.

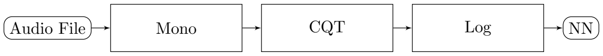


Figure 1: Preprocessing of audio before input to CNN

2.2. Convolutional Neural Network

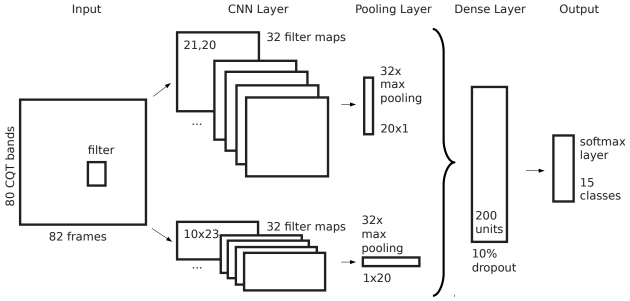


Figure 2: CNN architecture

Following [7] we created a parallel CNN architecture, which comprises a CNN Layer which is optimized for processing and recognizing relations in frequency domain, and a parallel one which is aimed at capturing temporal relations (c.f. Figure 2). Both parts of the CNN architecture use the same input, i.e. the 80 bands × 82 frames CQT matrix as output of step 1 described in Subsection 2.1. In each epoch of the training, multiple training examples, sampled from the segment-wise CQT extraction of all files in the training set, are presented to both pipelines of the neural network. Both CNN layers are followed by a Max Pooling layer, which performs a sub-sampling of the matrices that are output after applying the CNN's filter kernels. We describe this in more detail: In a Convolutional Neural Network, weights are essentially learned in a filter kernel of a particular shape. Multiple of such filter kernels – in our approach 32 in each pipeline – are applied to the input data, by convolving over the input image. Convolution means multiplication of the filter kernel with an equal sized portion of the input image. This filter kernel window is then moved sequentially over the input data (typically from left to right, top to bottom), producing an output of either equal size (when padding is used at the borders), or reduced by filter-length - 1 on each axis (when no padding is used, and the filter kernel is kept inside the borders of the input).

The particularity of this process is that the weights that are stored in each filter kernel are shared among the "input units" regardless of their input location. The filter weights are updated after each training epoch using back-propagation. Thus, by convolving over the input data, the filter kernels learn characteristic structures of the input data. The subsequent Max Pooling step serves as a data aggregation and reduction step. The pooling length in each direction determines how many "pixels" are aggregated together in the output. Max pooling thereby preserves only the maximum value from the input within its pooling window. Note that Max Pooling is applied to all 32 filter outputs (even though not visible in Figure 2).

In our CNN architecture, depicted in Figure 2, we use two pipelines of CNN Layer with 32 filter kernels each, following by a Max Pooling on all of these filter kernels. We altered the filter and pooling sizes, for further improvement after the submission of the DCASE Workshop paper [8]. Filter and pooling sizes are larger now. The lower pipeline is aimed at capturing frequency relations. Its filter kernel sizes are set to 10×23 and the Max Pooling size to 1×20. This means that the output of the filtering step is preserving more information on the frequency axis than in time. On the contrary, the upper pipeline uses filter sizes of 21×20 and pooling of 20×1, aggregating on the frequency axis and therefore retaining more information on the time axis.

In the next step, the parallel architecture is merged into a single pipeline, by flattening all the matrices from both previous pipelines, concatenating them and feeding them into a dense (fully connected) layer with 200 units. The parameters we described were found after a larger set of experiments (not described in this paper).

Recently, a number of techniques have been presented that make Deep Neural Networks generalize faster and better. One such technique is *Dropout*: it can be applied to any layer and reduces overfitting by dropping a percentage of random units at each weight update [10, 11]. Dropping means that it disregards these units in both input and output, so that they do not contribute to activation, nor to any weight updates. In terms of activation of a unit's output, the traditional Sigmoid function has been widely replaced by the ReLU: The *Rectified Linear Unit* simplifies and speeds up the learning process by using the activation function $f(x) = max(0, x)$ [12]. Due to its sparse activation (in a randomly initialized network) only about 50% of hidden units are activated, which makes the network generalize much faster [13]. The *Leaky ReLU* [14] is an extension to the ReLU that does not completely cut off activation for negative values, but allows for negative values close to zero to pass through. It is defined by adding a coefficient α in $f(x) = \alpha x$, for $x < 0$, while keeping $f(x) = x$, for $x \geq 0$ as for the ReLU.

In our architecture, we apply Leaky ReLU activation with $\alpha = 0.3$ in both Convolutional layers, and Sigmoid activation in the dense layer. We apply a Dropout value of 0.1 to the fully connected layer. The last layer is a so-called Softmax layer: It connects the 200 units of the preceding layer with as many units as the number of output classes (15), and applies the Softmax function to guarantee that the output activations to always sum up to 1 [13]. The output from the Softmax layer can be thought of as a probability distribution and is typically used for single-label classification problems. All layers are initialized with the Glorot uniform initialization [15].

For the results presented in Section 3.6 this CNN architecture was trained over 100 epochs with a constant learning rate of 0.02. The model is adapted in each epoch using Stochastic Gradient Descent (SGD) and a mini-batch-size of 40 instances.

The system is implemented in Python and using *librosa* for the CQT-transform and *Theano*-based library *Keras* for Deep Learning.

3. TASK 1: ACOUSTIC SCENE CLASSIFICATION

3.1. Data Set

For the development of the system we described in Section 2 we used the TUT acoustic scenes 2016 dataset provided by the DCASE 2016 organizers for task 1 on acoustic scene classification [1]. The goal of this task is to classify a recording into one of 15 different classes that represent urban and some non-urban environments. The 15 classes are: beach, bus, cafe/restaurant, car, city center, forest path, grocery store, home, library, metro station, office, park, residential area, train and tram. In this task 1, individual train and test files are exclusively labeled with one class.

The sounds were recorded from different locations (mostly in Finland) and use 44.1 kHz sampling rate and a 24 bit resolution. For each location, a 3-5 minute long audio recording was captured. The original recordings were then split into 30-second segments for the challenge. This imposes the need for particular attention when doing train/test set splits or cross-validation: one needs to make sure that recordings from the same location are not to be found in different sets, as it introduces a beneficial bias. Thus, the task organizers made sure that all segments from the same original recording are included in a single subset – either development dataset or evaluation dataset. They also provide a 4-fold cross-validation setup for the development set which ensures this correct splitting.

For each acoustic scene, 78 segments (39 minutes of audio) were included in the development dataset and 26 segments (13 minutes of audio) were kept for evaluation. The development set contains 1170 30-sec segments (in total 9h 45mins of audio), and the evaluation set 390 30-sec segments (3h 15mins).

Full annotations for the *development set* were available, but no annotations for the *evaluation set*, as the task was still open at the time of this writing. We therefore exclusively used the *development set* of this data set to create, improve and evaluate our methodology for acoustic scene classification described in the next section, using the 4-fold cross-validation splits provided by the organizers.

3.2. Baseline

The baseline system is a GMM classifier using MFCC audio features calculated using frames of 40 ms with a Hamming window and 50 % overlap. 40 Mel bands are extracted but only the first 20 coefficients are kept, plus delta and acceleration coefficients (60 values in total). The system learns one acoustic model per acoustic scene class (GMM with 32 components) and performs the classification using a maximum likelihood classification scheme (expectation maximization) [1]. The reported average classification accuracy over 4 folds is 72.5 %.

3.3. Evaluation Measures

As our system analyzes and predicts multiple audio segments per input audio file, there are several ways to perform the final prediction of an input instance:

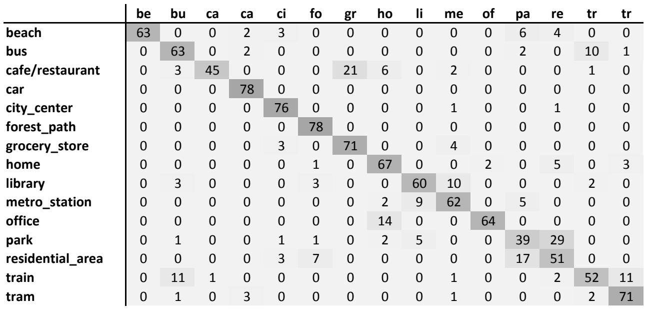
Maximum Probability: The output probabilities of the Softmax layer for the 15 classes are summed up for all segments belonging to the same input file. The predicted class is determined by the maximum probability among the classes from the summed probabilities.

Majority Vote: Here, the predictions are made for each segment processed from the audio file as input instance to the network. The class of an audio segment is determined by the maximum probability as output by the Softmax layer for this segment instance. Then, a majority vote is taken on all predicted classes from all segments of the same input file. Majority vote determines the class that occurs most often.

In both cases, the resulting accuracy is determined by comparing the file-based predictions to the groundtruth provided by the task organizers. All accuracies mentioned in this paper as well as the system submitted to the DCASE challenge use the *Maximum Probability* strategy for decision making.

3.4. Results on the Development Set

For our experimental results, we used exclusively the *development* dataset that was provided by the DCASE 2016 acoustic scene classification task organizers, as described before. The task organizers also provided a cross-validation setup for this development dataset, which consists of 4 folds distributing the 78 available segments based on location, to ensure that all files recorded in same location are placed on the same side of the evaluation, in order to prevent bias from recognizing the recording location.We used the provided fold splits in order to make results comparable to other work, including the baseline system that was also provided by the task organizers. The reported average classification accuracy of the baseline system over 4 folds is 72.5 %. With our main approach of a parallel CNN described in Section 2 we achieve 80.33 % accuracy by max probability and 80.76 % accuracy by majority vote on 4 fold cross-validation.



	be	bu	ca	ca	ci	fo	gr	ho	li	me	of	pa	re	tr	tr
beach	63	0	0	2	3	0	0	0	0	0	0	6	4	0	0
bus	0	63	0	2	0	0	0	0	0	0	0	2	0	10	1
cafe/restaurant	0	3	45	0	0	0	21	6	0	2	0	0	0	1	0
car	0	0	0	78	0	0	0	0	0	0	0	0	0	0	0
city_center	0	0	0	0	76	0	0	0	0	1	0	0	1	0	0
forest_path	0	0	0	0	0	78	0	0	0	0	0	0	0	0	0
grocery_store	0	0	0	0	3	0	71	0	0	4	0	0	0	0	0
home	0	0	0	0	0	1	0	67	0	0	2	0	5	0	3
library	0	3	0	0	0	3	0	0	60	10	0	0	0	2	0
metro_station	0	0	0	0	0	0	0	2	9	62	0	5	0	0	0
office	0	0	0	0	0	0	0	14	0	0	64	0	0	0	0
park	0	1	0	0	1	1	0	2	5	0	0	39	29	0	0
residential_area	0	0	0	0	3	7	0	0	0	0	0	17	51	0	0
train	0	11	1	0	0	0	0	0	0	1	0	0	2	52	11
tram	0	1	0	3	0	0	0	0	0	1	0	0	0	2	71

Figure 3: Confusion Matrix of the optimized model including two-class predictions (Figure 2)

3.5. Improvement with 2-class models

When we investigate the per-class accuracies by having a look at the confusion matrix in Figure 3, it can be observed that the best configuration of the proposed system excels for the classes *car*, *city center*, *grocery store* and *tram*. The largest confusions are between the classes *residential area* and *park*, *cafe/restaurant* and *grocery store* as well as *tram*, *train* and *bus*. We also noticed that the fold accuracies vary quite heavily.

To overcome the biggest mistakes in the model, we provide a 2nd approach, training additional pairwise (2 class) models on the following classes:

- park vs. residential area
- cafe/restaurant vs. train
- home vs. library

We use the original base model, which is trained on all the classes, to make initial predictions. Once a prediction is made for any of the 6 classes above, this instance will be sent to an additional model, trained only on 2 classes. This new prediction will replace the original prediction. By this, we hope to overcome the confusion between the most difficult classes. In our tests on the development set, this could improve the results by 1 to 2 %.

The architectures for these 2-class models are single layer CNNs with 15 filters each, and the following filter and pooling sizes:

- park vs. residential area: filter 21×20, pooling 1×20
- cafe/restaurant vs. train: filter 21×20, pooling 1×20
- home vs. library: filter 21×20, pooling 20×1

As in the parallel architecture, the CNN and pooling layer is followed by a full layer with 200 units and a Softmax layer, containing 2 output units. Activation is Leaky ReLU for the CNN layer, Sigmoid for the full layer, and a Dropout of 0.1 has been used. These models were trained on the respective instances of these classes in the development set, over 50 epochs with a constant learn rate of 0.02 (except cafe/restaurant vs. train, where the learn rate was 0.0002).

In the end we submitted predictions of 2 models:

- CQTCNN_1: the base model as in Figure 2
- CQTCNN_2: the base model improved by the 3 pairwise class models

3.6. Results on the Evaluation Set

In the DCASE 2016 challenge on task 1, CQTCNN_1 (the base model) achieved 81.8 % accuracy (rank 18 of 35) and CQTCNN_2 (the improved model) scored 83.3 % (rank 14 of 35) on the evaluation set. The best algorithm achieved 89.7 % accuracy.

4. TASK 4: DOMESTIC AUDIO TAGGING

4.1. Data set

This task is based on audio recordings made in a domestic environment. The objective of the task is to perform multi-label classification on 4-second audio chunks (i.e. assign zero or more labels to each 4-second audio chunk).[3]

Predictions shall be made for the following 7 classes:

- c: Child speech
- m: Adult male speech
- f: Adult female speech
- v: Video game / TV
- p: Percussive sounds, e.g. crash, bang, knock, footsteps
- b: Broadband noise, e.g. household appliances
- o: Other identifiable sounds

The dataset however contains an 8th class: silence ('S'), which is also annotated for audio chunks. This class is also predicted by our system, however, gracefully ignored by the evaluation system used by the task organizers.

4378 chunks are provided for system development, based on partitioning at the level of 5-minute recording segments. For each chunk, multi-label annotations were first obtained for each of 3 annotators [16]. However, for system development, only chunks where an agreement (by majority vote) of the annotators is present, are used, leaving 1946 such 'strong agreement' chunks for the development dataset. 816 majority vote agreed chunks are used for evaluation of the task submission.

The audio data for the development set was provided in 48kHz stereo and 16kHz mono format. However, the evaluation set is only provided with 16kHz mono (obtained by downsampling the right-hand channel of the 48kHz recordings) "with the aim of approximating typical recording capabilities of commodity hardware".

4.2. Audio Preprocessing

As the evaluation on this task is performed on 16 kHz audio, we also used only the 16 kHz mono examples provided in the development set for training. We, however, resampled the audio inputs to 22 kHz,

[3] `http://www.cs.tut.fi/sgn/arg/dcase2016/task-audio-tagging`

to ensure the compatibility with our framework that is optimized for 22 and/or 44 kHz, before feeding it to the audio preprocessing described in Section 2.1.

4.3. Approach

We used, in principle, the parallel CNN as described in Section 2.2 for this task. Due to this task being a multi-label prediction task, we altered the final output layer from a Softmax layer to a standard full layer with Sigmoid activation function. This is because Softmax optimizes the output to have probabilities summing up to 1 for all classes, which is only desirable in single-label classification.

For the predictions on the evaluation set of DCASE task 4 we trained the system on the 'refined development set', containing the 1946 instances only where the annotators had a strong agreement. For the evaluation, 816 instances, also with a strong annotator agreement, were used (with 16 kHz mono audio format).

No additional 2-class models were used in this system and task.

4.4. Evaluation Measure

As per the task organizers, performance is measured using the equal error rate (EER), which is defined as the fixed point of the graph of false negative rate versus false positive rate [17]. The EER is computed individually for each label. We did not measure the EER on the development set.

4.5. Results

Our approach won the DCASE 2016 domestic audio tagging task with 16.6% equal error rate on the evaluation set. The DCASE baseline system of this task has an equal error rate of 20.9% and the system ranked second scored at 16.8% EER, also using a Convolutional Neural Network approach. The results vary rather strongly between the classes, a detailed comparison of class-wise performance of all submitted algorithms can be found on the result web site.[4]

5. SUMMARY

We have shown how we adapted a musically inspired Convolutional Neural Network approach to recognize acoustic scenes from recordings of urban and domestic environments. The crucial adaptations were the utilization of the Constant-Q-Transform to capture essential audio information from both low and high frequencies in sufficient resolution and the creation of a parallel CNN architecture, which is capable of capturing both relations in time and frequency. The presented Deep Neural Network architecture has shown a 10.7 % relative improvement over the baseline system provided by the DCASE 2016 Acoustic Scene Classification task organizers, achieving 80.25 % on the development set and the same 4-fold cross-validation setup as provided. Moreover, it achieved 83.3 % on the evaluation set, ranking 14th of 35 in the DCASE 2016 challenge's task 1. On task 4 on domestic audio tagging, our approach was the winning algorithm (rank 1 of 9) with 16.6 % equal error rate. We conclude that this system is capable of detecting urban and domestic acoustic settings, yet there is ample room for improving the system further.

6. ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research.

7. REFERENCES

[1] T. H. Annamaria Mesaros and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *24th European Signal Processing Conference (EUSIPCO 2016)*, Budapest, Hungary, 2016.

[2] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 6964–6968, 2014.

[3] J. Downie, K. West, A. Ehmann, and E. Vincent, "The 2005 music information retrieval evaluation exchange (MIREX 2005): preliminary overview," in *6th Int. Conf. on Music Information Retrieval (ISMIR)*, 2005, pp. 320–323.

[4] T. Lidy, "Spectral convolutional neural network for music classification," in *Music Information Retrieval Evaluation eXchange (MIREX)*, Malaga, Spain, October 2015.

[5] T. Lidy and A. Rauber, "Evaluation of feature extractors and psycho-acoustic transformations for music genre classification," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, UK, September 11-15 2005, pp. 34–41.

[6] T. Lidy, C. N. S. Jr., O. Cornelis, F. Gouyon, A. Rauber, C. A. A. Kaestner, and A. L. Koerich, "On the suitability of state-of-the-art music information retrieval methods for analyzing, categorizing, structuring and accessing non-western and ethnic music collections," *Signal Processing*, vol. 90, no. 4, pp. 1032 – 1048, April 2010, Special section: ethnic music audio documents: from the peservation to the fruition.

[7] J. Pons, T. Lidy, and X. Serra, "Experimenting with musically motivated convolutional neural networks," in *Proceedings of the 14th International Workshop on Content-based Multimedia Indexing (CBMI 2016)*, Bucharest, Romania, June 2016.

[8] T. Lidy and A. Schindler, "CQT-based Convolutional Neural Networks for Audio Scene Classification," in *Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE 2016)*, Budapest, Hungary, 2016.

[9] C. Schörkhuber and A. Klapuri, "Constant-Q transform toolbox for music processing," *7th Sound and Music Computing Conference*, pp. 3–64, Jan 2010.

[10] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv: 1207.0580*, pp. 1–18, 2012.

[11] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research (JMLR)*, vol. 15, pp. 1929–1958, 2014.

[12] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," *Proceedings of the 27th International Conference on Machine Learning*, no. 3, pp. 807–814, 2010.

[4] `http://www.cs.tut.fi/sgn/arg/dcase2016/task-results-audio-tagging`

[13] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com

[14] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," *ICML 2013*, vol. 28, 2013.

[15] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 9, 2010, pp. 249–256.

[16] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "Chime-home: A dataset for sound source recognition in a domestic environment," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2015 IEEE Workshop on*. IEEE, 2015, pp. 1–5.

[17] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, 2012.