CQT-BASED CONVOLUTIONAL NEURAL NETWORKS FOR AUDIO SCENE CLASSIFICATION

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

In this paper, we propose a parallel Convolutional Neural Network architecture for the task of classifying acoustic scenes and urban sound scapes. A popular choice for input to a Convolutional Neural Network in audio classification problems are Mel-transformed spectrograms. We, however, show in this paper 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. These are non-trivial in audio tasks due to the different semantics of the two axes of the input data: time vs. frequency. Finally, we propose a parallel (graph-based) neural network architecture which captures relevant audio characteristics both in time and in frequency. Our approach shows a 10.7 % relative improvement of the baseline system of the DCASE 2016 Acoustic Scenes Classification task [1].

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

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). In this paper we focus particularly on acoustic scene classification in urban environments (task 1). The goal of this task is 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].

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 task of audio scene classification.

In Section 2 we will give a brief overview of related work. Section 3 describes the data set and the task's challenge. Section 4 describes our method in detail, while Section 5 presents preliminary results. Finally, Section 6 summarizes the paper and provides conclusions.

2. RELATED WORK

A variety of publications study the modeling of audio signals in time and/or frequency domain for the purpose of acoustic scene recognition and event detection. Mel-frequency Cepstral Coefficients (MFCCs) typically model the frequency relations very well and are

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

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

used frequently as part of an audio event detection system. However, MFCCs without any derivatives are not performing very well. Only by including derivatives of first and second order the temporal context is included to a certain extent [8]. The authors of [9] propose to use the Matching Pursuit (MP) algorithm to supplement MFCC features to yield higher accuracy. They demonstrate the effectiveness of joint MP + MFCC features for unstructured environmental sound classification, e.g. chirpings of insects and sounds of rain, which are investigated for their temporal domain signatures. Cotton and Ellis [10] propose an approach modeling acoustic events directly describing temporal context. They use convolutive nonnegative matrix factorization (NMF) to discover spectro-temporal patch bases, which correspond to event-like structures. Features are derived from the activations of these patch bases. Mesaros et al present a combination of MFCC features with a Hidden Markov Models (HMM) based audio event detection system [11]. They test it on a diverse set of 61 classes of isolated events (54 % accuracy) as well as real life recordings (23.8 % avg. accuracy).

In 2013, the IEEE AASP Challenge "Detection and Classification of Acoustic Scenes and Events" (DCASE) was organized for the first time to help move forward the research in this domain [12, 13]. The dataset used in the acoustic scene classification task comprised 10 classes of indoor and outdoor urban and office sounds, similar to the current one. The authors of [12] also provide an overview of previous approaches in literature: The two main methodologies are 1) the bag-of-frames approach using a set of low-level features (e.g. MFCCs), modeling long-term statistical distribution of the local features and 2) the use of an intermediate representation that models the scene using higher level features that are usually captured by a vocabulary of "acoustic atoms", which represent audio events that are learned in an unsupervised manner from the data (e.g. by NMF). The authors also present a NMF-based system for the event detection task, in which the constant-Q transform (CQT) is used for the time-frequency representation, with a log-frequency resolution of 60 bins per octave. The best performing system in the DCASE 2013 office live event detection task [8] laid the focus on spectro-temporal features and used a two-layer HMM. The authors compare amplitude modulation spectrograms, Gabor filterbank features and MFCCs and employ various noise reduction / signal enhancement strategies. The use of Gabor filters is motivated by their similarity to spectro-temporal patterns of neurons in the auditory cortex of mammals. Their proposed spectro-temporal features achieve a better recognition accuracy than MFCCs. Another work that explores the temporal dynamics in the audio and tackles the sensitivity of MFCCs to background noise is found in [14]. The authors present a work on unsupervised feature learning for urban sound classification, employing the spherical k-means algorithm for feature learning. It is shown that classification accuracy can be significantly improved by feature learning if the domain specificities are taken into account – in this case capturing the temporal dynamics of urban sound sources.

The authors of [15] use a framework of spectrogram image-based SIF features and human auditory system modeling SAI features (stabilized auditory image) in various configurations together with Support Vector Machines (SVM) and Deep Neural Networks (DNN). The DNN uses 5 to 6 fully connected layers with 100 to 300 hidden units each and is shown to perform better than the SVM. A comparison is done to a range of other systems on various noise levels. One finding is that the SIF features used in conjunction with DNN incorporate additional temporal context being advantageous for classification in noisy environments.

3. DATA SET

For both the development and the evaluation of the system we describe in Section 4 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.

4. METHOD

For the task of acoustic scene classification we use Convolutional Neural Networks, which we trained on CQT-transformed audio input. We describe these two parts in more detail.

4.1. 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 [16]. 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

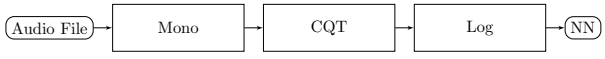

Figure 1: Preprocessing of audio before input to CNN

audible frequency range, and also the fundamental frequencies of the tones in Western music are geometrically spaced along the standard 12-tone scale [16]. 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 $\times$ 82 frames.

4.2. Convolutional Neural Network

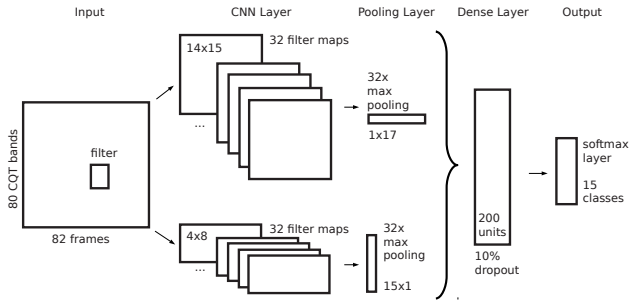

Figure 2: CNN architecture

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 $\times$ 82 frames CQT matrix as output of step 1 described in Subsection 4.1. 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. The upper pipeline is aimed at capturing frequency relations. Its filter kernel sizes are set to 14$\times$15 and the Max Pooling size to 1$\times$17. This means that the output of the filtering step is 32 matrices of shape 67$\times$68, which are then "pooled" to 32 matrices of shape 67$\times$4, preserving more information on the frequency axis than in time. On the contrary, the lower pipeline uses filter sizes of 4$\times$8 and pooling of 15$\times$1, aggregating on the frequency axis and therefore retaining more information on the time axis: Its output shape is 5$\times$75 (32 times).

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. Note that the input to this layer is 20,576 weights (the flattened output of the two previous pipelines) and with 200 fully connected units (and one bias) this layer has 4,115,400 parameters. The complete network has 4,126,223 parameters, which hints at the power of Convolutional Layers: to drastically reduce the weights that are needed to make the network learn, through the spatial weight sharing principle of the filter convolution approach. Note, however, that setting the filter and pooling parameters is less straight-forward than in image retrieval where typically quadratic shapes are used for both the filter and the pooling shapes, due to the different semantics of the two axes while in images the axes have the same semantics. 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 [17, 18]. 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 $f(x) = max(0, x)$ [19]. 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 [20]. The *Leaky ReLU* [21] 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 [20]. 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 [22].

For the experiments presented in Section 5 this CNN architec-

ture 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.

5. RESULTS

5.1. Data set and Baseline

For our experimental results, we used exclusively the *development* dataset that was provided by the DCASE 2016 Acoustic Scene Classification task organizers, which was described in Section 4.The task organizers also provide 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 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 %.

5.2. Evaluation

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. We present the result achieved by the system described in Section 4 and compare the impact of different audio transformations as an input step to the CNN. Table 1 shows the results. The Mel frequency transforms have been computing using a Fast Fourier Transform (FFT) with a Hanning window of 1024 samples and 50 % overlap. The segment size of the audio chunks has been chosen to be equal to the one used for CQT, which results in 80 frames. The FFT spectrogram frequency bands are transformed to Mel scale by applying 40 or 80 Mel filters. Subsequently a Log_{10} transform is applied. From the results table we see that the approach with 80 Mel filters performed only slightly better. Yet, we also see that using the CQT instead of the Mel-transform has a beneficial impact. The best result is achieved with 80 CQT bands. It is 80.25 % accuracy with the Maximum Probability strategy and 80.07 % with Majority Vote. Applying the full standard CQT of 7 octaves with 12 semi-tones each performed worse. Extending the 12 semi-tones to 18 bands per octave, with 126 CQT bands in total (covering the same 7 octaves) did also not improve the results.

Transform	Bands / Frames	$A_{maxprob}$	$A_{majvote}$
Mel	40×80	76.23%	75.62%
Mel	80×80	76.55%	76.38%
CQT	80×82	**80.25%**	**80.07%**
CQT	84×82	78.11%	77.59%
CQT	126×82	79.39%	79.14%

Table 1: Different input transformations to the parallel CNN

	li	ci	tr	pa	fo	gr	re	ca	tr	of	be	me	bu	ho	ca
library	61	0	0	1	0	3	0	0	3	0	0	10	0	0	0
city_center	0	76	0	0	0	0	1	0	0	0	0	1	0	0	0
tram	0	0	70	1	0	1	0	3	2	0	0	0	1	0	0
park	5	0	0	35	0	0	26	0	0	0	7	1	2	2	0
forest_path	0	0	0	0	75	0	2	0	0	0	0	0	0	0	1
grocery_store	0	3	0	0	0	73	0	0	0	0	0	2	0	0	0
residential_area	0	3	0	20	7	0	48	0	0	0	0	0	0	0	0
car	0	0	3	0	0	0	0	75	0	0	0	0	0	0	0
train	0	0	11	3	0	0	2	0	51	0	0	1	9	0	1
office	0	0	0	0	0	0	0	0	0	68	0	0	0	10	0
beach	0	1	0	5	0	0	6	2	0	0	64	0	0	0	0
metro_station	6	0	0	5	0	0	0	0	0	0	0	66	0	1	0
bus	0	0	2	2	0	0	1	1	11	0	0	0	61	0	0
home	0	0	3	0	1	0	2	0	0	1	0	0	0	71	0
cafe/restaurant	0	0	0	0	0	20	0	0	1	0	0	3	3	6	45

Figure 3: Confusion Matrix of the best model of Table 1.

Next, 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 *city center*, *forest path*, *grocery store*, *car* and *home* with accuracies ranging from 91% to 97.4%. The largest confusions are between the classes *residential area* and *park*, *cafe/restaurant* and *grocery store* as well as *tram*, *train* and *bus*.

6. SUMMARY

We have shown how we adapted a musically inspired Convolutional Neural Network approach to recognize acoustic scenes from recordings of urban 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 same 4-fold cross-validation setup as provided. With a few additional optimizations, described in our submission abstract [23], our approach achieved 83.3 % accuracy on the evaluation set in the challenge (rank 14 of 35). On the domestic audio tagging task, our improved approach is the winning algorithm (rank 1 of 9) with 16.6 % equal error rate. We conclude that this system is capable of detecting urban acoustic settings, yet there is ample room for improving the system further.

7. ACKNOWLEDGMENTS

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

8. 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] J. Schröder, N. Moritz, M. R. Schadler, B. Cauchi, K. Adiloglu, J. Anemuller, S. Doclo, B. Kollmeier, and S. Goetze, "On the use of spectro-temporal features for the IEEE AASP challenge 'detection and classification of acoustic scenes and events'," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, USA.

[9] S. Chu, S. Narayanan, and C. C. J. 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.

[10] C. V. Cotton and D. P. W. Ellis, "Spectral vs. spectro-temporal features for acoustic event detection," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2011, pp. 69–72.

[11] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real life recordings," in *European Signal Processing Conference*, 2010, pp. 1267–1271.

[12] D. Giannoulis, D. Stowell, E. Benetos, M. Rossignol, M. Lagrange, and M. D. Plumbley, "A database and challenge for acoustic scene classification and event detection," in *European Signal Processing Conference*, 2013.

[13] 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.

[14] J. Salamon and J. P. Bello, "Unsupervised feature learning for urban sound classification," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. April, South Brisbane, Australia, 2015, pp. 171–175.

[15] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao, "Robust Sound Event Classification Using Deep Neural Networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 540–552, Mar 2015.

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

[17] 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.

[18] 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.

[19] 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.

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

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

[22] 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.

[23] T. Lidy and A. Schindler, "Cqt-based convolutional neural networks for audio scene classification and domestic audio tagging," IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2016), Budapest, Hungary, Tech. Rep., September 3 2016.