Interspeech 2018
2-6 September 2018, Hyderabad

Temporal transformer networks for acoustic scene classification

Teng Zhang[1], *Kailai Zhang*[2], *Ji Wu*[3]

[123]Multimedia Signal and Intelligent Information Processing Lab
Department of Electronic Engineering, Tsinghua University, Beijing, P.R.China
zhangteng1887@gmail.com, zhang-kl13@tsinghua.org.cn, wuji_ee@mail.tsinghua.edu.cn

Abstract

Neural networks have been proven to be powerful models for acoustic scene classification tasks, but are still limited by the lack of ability to be temporally invariant to the audio data. In this paper, a novel temporal transformer module is proposed to allow the temporal manipulation of data in neural networks. This module is composed of a Fourier transform layer for feature maps and a learnable feature reduction layer, and can be inserted into existing convolutional neural network (CNN) and Long short-term memory (LSTM) models. Experiments on LITIS Rouen dataset and DCASE2016 dataset show that the proposed method leads to a significant improvement when compared with the existing neural networks. Our approach is able to perform significantly better than the state-of-the-art result on LITIS Rouen dataset, obtaining a relative reduction of 23.6% on classification error.

Index Terms: acoustic scene classification, long short-term memory, convolutional neural network, Fourier transform

1. Introduction

Acoustic scene classification (ASC) is a task of classifying environments from the sounds they produce [1][2], with applications in devices where the environment can be defined based on physical or social context, e.g., park, office, meeting, etc [3].

Influenced by traditional speech and music processing methods, early works on ASC focused on modeling the time-frequency characteristic of audio features. Eronen [4] employed Mel-frequency cepstral coefficients (MFCCs) as features, and constructed Gaussian mixture and hidden markov models (GMM-HMM) to get knowledge about acoustic categories. Eronen and co-authors [5] further developed on this work by considering a larger group of features, obtaining an overall 58% accuracy in the classification of 18 different acoustic scenes. In recent years, more specific features are motivated by the fact that the environmental sound is different from speech and music in time and frequency structures. New features are often inspired by other research fields such as image processing features [6][7], matrix factorization features [8][9], unsupervised learning features [10] and so on. Meanwhile, Deep neural networks [11][12] are also used to learn feature representations. Supported by large amounts of training data, deeper architectures significantly improve the performance of many tasks in speech processing and computer vision. However, the performance of acoustic scene classification methods based on deep neural networks is relatively poor and asks for more efforts.

Some researchers have treated audio spectrograms as natural images in audio processing tasks. Fig.1 shows two spectrograms of "restaurant" scene, the collision sound of dishes is a typical sound in"restaurant" scene, which occurs in time t_1 and t_2 separately, arising from the random segmentation during the actual recording process. Intuitively, shifting the spectrograms in time direction according to t_1 and t_2 does not affect the semantic representation of these spectrograms. A desirable property of an ASC system which is able to reason about audios is to eliminate the temporal shifting interference. In image processing field, natural images are often taken as equivalent in each direction, the small object pose and part deformation can be disentangled using local max-pooling layers in convolutional neural network (CNN) [13][14][15]. Unlike pooling layers where the receptive fields are fixed and local, spatial transformer networks [16] was proposed to actively spatially transform an image or a feature map by producing an appropriate transformation for each input sample. Furthermore, reinforcement learning driven selective attention networks [17] was introduced to model selective attention in deep CNN. However, for audio signals, the spectrogram structure is not equivalent in time and frequency direction. Thus these mechanisms are not invariant to long-term temporal transformations of audio data.

In this paper, we introduce a temporal transformer module, which can be inserted into existing neural networks to provide temporal transformation capabilities. This module is composed of a Fourier transform layer for feature maps and a learnable feature reduction layer. Unlike pooling layers and other spatial transformer mechanisms in image processing field, the temporal transformer module is able to deal with the arbitrary length of temporal shifting. The transformation can be performed not only on audio spectrograms but also on all feature maps produced by neural networks. Notably, this module can be trained with standard back-propagation, allowing for the end-to-end training of the models they are injected in.

The rest of the paper is organized as follows. In Section 2, we describe our motivation and implementation details of temporal transformer module. Next, we discuss how to insert the temporal transformer module to existing CNN and LSTM models in Section 3. Then we conduct several experiments and evaluate the performance of the proposed method in Section 4. At last, we conclude this paper and present our future work in Section 5.

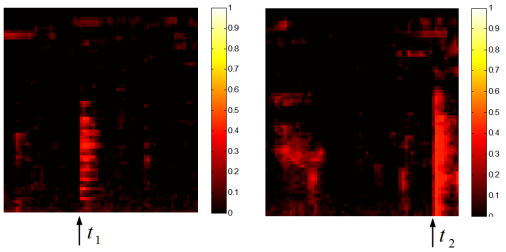


Figure 1: *Spectrogram examples of restaurant scene.* t_1 *and* t_2 *are the start points of collision sound of dishes.*

10.21437/Interspeech.2018-1152

2. Temporal Transformer Module

In this section, we describe the implementation details of a temporal transformer module. The input audio signal is first transformed to a sequence of vectors using Short-time Fourier Transform (STFT) [18], the output spectrogram can be represented as $\boldsymbol{X}_{1...T} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$. T is determined by the frameshift in STFT, corresponding to the time resolution in frame theory [19]. The dimension of each vector $\boldsymbol{x}$ can be labeled as N, which is determined by frame length. Temporal transformer module is a differentiable module which applies a temporal transformation to spectrograms or feature maps such as $\boldsymbol{X}$, producing a single output vector $\boldsymbol{y} = \{y_1, y_2, ..., y_N\}$. For simplicity, we first consider single transforms for each row of the spectrogram, which can be represented as $\tilde{\boldsymbol{x}}_{1...T} = \{x_{i,1}, x_{i,2}, ..., x_{i,T}\}$, where i is the row index of $\boldsymbol{X}$. Then we can generalize to the whole spectrogram, as shown in experiments.

A temporal transformer module consists of two parts, a Fourier transform layer and a feature reduction layer. The Fourier transform method is injected here to eliminate the temporal shifting. The feature reduction layer is used to produce a single output per transformer. The combination of these two parts forms a temporal transformer module and will now be described in more details in the following sections.

2.1. Fourier Transform Layer

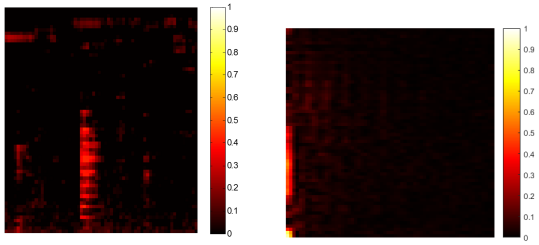
We start with the simplified situation that the temporal shifting of a discrete time series $\tilde{\boldsymbol{x}}$ can be shown as $\tilde{\boldsymbol{x}}_t = \{x_{i,(t+1)_T}, x_{i,(t+2)_T}, ..., x_{i,(t+T)_T}\}$, where t is the temporal shifting and represent unwanted temporal variations, $(n)_T$ represents the modulo operation that $(n)_T = (n \ mod \ T) + 1$. We desire to eliminate the temporal shifting and obtain the unified expression for both $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}_t$.

The Fourier transform [20] and related techniques are of importance in signal processing filed. For our case, the discrete Fourier transform (DFT) representations of $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}_t$ can be given as Eq.1, where k is the frequency index and j is the imaginary unit. Then we compute the absolute value and get the expression that $\|\tilde{\boldsymbol{D}}(k)\| = \|\tilde{\boldsymbol{D}}_t(k)\|$. Now the temporal shifting t has been eliminated from $\tilde{\boldsymbol{x}}_t$.

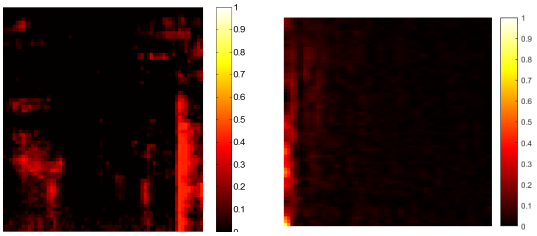
$$\tilde{\boldsymbol{D}}(k) = \frac{1}{T}\sum_{n=1}^{T} x_{i,n} e^{-j\frac{2\pi nk}{T}}$$
$$\tilde{\boldsymbol{D}}_t(k) = \frac{1}{T}\sum_{n=1}^{T} x_{i,(t+n)_T} e^{-j\frac{2\pi nk}{T}} \quad (1)$$

We generalize to the whole spectrogram, each row of the spectrogram is operated using DFT as Eq.1, then the absolute value can be shown as Fig.2. Fig.2b and Fig.2d are respectively the transformed representation of Fig.2a and Fig.2c. The result shows that the temporal shifting between Fig.2a and Fig.2c has been almost eliminated. For our classification task, Fig.2b and Fig.2d can simplify the subsequent process, and lead to superior classification performance.

The fast Fourier transform (FFT) algorithm [21] has been used for a long time to implement DFT in many signal processing applications because of its high efficiency. However, when we insert this module into standard neural networks, it is difficult to implement parallel FFT operations on GPU devices, and the deep structure of FFT will block the standard back-propagation in the network. These limitations of FFT make us trace back to the original definition of DFT as Eq.1. The definition is rewritten in the following form for convenience of explanation:



(a) *Original-spec-1* (b) *Fourier-spec-1*



(c) *Original-spec-2* (d) *Fourier-spec-2*

Figure 2: *Spectrograms after Fourier transform layer. (a) and (c) are the original spectrograms produced by STFT algorithm. (c) and (d) are the correspondingly spectrograms processed after Fourier transform layer.*

$$\boldsymbol{W^1} = [cos(\frac{2\pi nk}{T})]_{nk}$$
$$\boldsymbol{W^2} = [sin(\frac{2\pi nk}{T})]_{nk}$$
$$\tilde{\boldsymbol{S}}(ik) = \sqrt{(\sum_{n=1}^{T} x_{in}\boldsymbol{W^1}_{nk})^2 + (\sum_{n=1}^{T} x_{in}\boldsymbol{W^2}_{nk})^2} \quad (2)$$

where $1 \le n \le T$, $1 \le k \le T$, $1 \le i \le N$, $\tilde{\boldsymbol{S}}(ik) = \|\tilde{\boldsymbol{D}}(k)\|$ as Eq.1.

In this form, $\boldsymbol{W^1}$ and $\boldsymbol{W^2}$ are pre-set parameters and do not need training in neural networks. DFT operation is now simplified as a parameterless fully connected layer in neural networks, which makes standard back-propagation procedure efficient and practical. Some speech recognition systems benefited from using Mel-frequency scale instead of real frequency, whereas in our case, this mapping has not provided any notable increase in performance.

2.2. Feature Reduction Layer

To perform feature reduction for $\tilde{\boldsymbol{S}}$ in Eq.2, we apply an additional feature reduction layer to produce the output feature vector. For audio spectrograms, each row represents a frequency bin. From our earlier investigation, we know that the energy distribution and coherence vary tremendously in different frequency bins. Thus the feature reduction is applied to each row of $\tilde{\boldsymbol{S}}$ respectively. The concatenation of all these reduced features forms the output feature vector of the whole temporal transform module. This layer takes $\tilde{\boldsymbol{S}} \in \mathbb{R}^{N\times T}$ as input and output feature vector $\boldsymbol{y} \in \mathbb{R}^N$ as Eq.3, where $\Theta \in \mathbb{R}^{N\times T}$ is the trainable parameters and can be trained with standard back-

propagation method.

$$\tilde{\boldsymbol{y}}(i) = \sum_{k=1}^{T} \tilde{\boldsymbol{S}}_{ik} \Theta_{ik}$$
$$\boldsymbol{y} = [\tilde{\boldsymbol{y}}_1, \tilde{\boldsymbol{y}}_2, ..., \tilde{\boldsymbol{y}}_N] \quad (3)$$

3. Temporal Transformer Networks

As described in Section 2, the proposed temporal transformer module can be inserted into many existing neural network structures. In this section, we introduce three commonly used structures including deep neural network (DNN), CNN and LSTM to verify the applicability of this module.

Fig.3(a) is the DNN structure consisting of several fully connected layers and a softmax layer. When temporal transformer module is inserted, the input spectrogram $\boldsymbol{X}$ is processed using Eq.2 and Eq.3 to produce a feature vector $\boldsymbol{y}$ for the following fully connected layers. However, for DNN structure without temporal transformer module, an average-overtime pooling operation over $\boldsymbol{X}$ is applied and the average value $\boldsymbol{y}_i = mean(\tilde{\boldsymbol{x}})$ is taken as the feature vector, where $\tilde{\boldsymbol{x}}$ has been defined in Section 2.

Fig.3(b) is the CNN structure as described in [22]. In general, $\boldsymbol{x}_{i:i+j}$ refers to the concatenation of frames $[\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, ...\boldsymbol{x}_{i+j}]$. The convolution operation involves a filter $\boldsymbol{w} \in R^{hm}$, which is applied to a window of h frames to produce a new feature. For example, a feature c_i is generated from a window of frames $\boldsymbol{x}_{i:i+h-1}$ by Eq.4, where $b \in R$ is a bias term and f is a non-linear function. This filter is applied to each possible window of frames to produce a feature map $\boldsymbol{c} = [c_1, c_2, ...c_{T-h+1}]$. For CNN structure with temporal transformer module, $\boldsymbol{c}$ is processed using Eq.2 and Eq.3 to produce a feature value $\boldsymbol{y}_i$. Otherwise, a max-overtime pooling operation[23] over the feature map is applied and the maximum value $\boldsymbol{y}_i = max(\boldsymbol{c})$ is taken as the feature corresponding to this filter. For both cases, one feature is extracted using one filter. This model uses multiple filters with varying window sizes to obtain multiple features. The features extracted here are then passed to several fully connected layers and a softmax layer, whose structures are the same with Fig.3(a).

$$c_i = f(\boldsymbol{w} \cdot \boldsymbol{x}_{i:i+h-1} + b) \quad (4)$$

Fig.3(c) is the LSTM structure as described in [24]. The LSTM layer takes the spectrogram $\boldsymbol{X}$ as an input sequence with length T and outputs $\boldsymbol{H}_{1...T} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_T\}$. When temporal transformer module is inserted, the LSTM outputs $\boldsymbol{H}$ is processed using Eq.2 and Eq.3 to produce a feature vector $\boldsymbol{y}$ for the following fully connected layers. Otherwise, the final hidden state is used as the feature vector. The following fully connected layers and softmax layer are the same with Fig.3(a).

The classification loss of these three structures is given by Eq.5, where n is the number of audios, k is the number of categories, l is the number of layers, $\boldsymbol{o}$ is the category labels and $\boldsymbol{p}$ is the probability distribution produced by neural networks. In this case, neural networks in Fig.3 can be optimized using standard back-propagation method.

$$\epsilon = \sum_{i=1}^{n} \sum_{j=1}^{k} \boldsymbol{o}_{ij} \cdot log(\boldsymbol{p}_{ij}) + \lambda \sum_{l} \parallel \boldsymbol{w}_l \parallel^2 \quad (5)$$

4. Experimental Evaluation

In this section, we employ LITIS ROUEN dataset [6] and DCASE2016 dataset [3] to conduct acoustic scene classification experiments.

Details of these datasets are listed as follows.

- *LITIS ROUEN dataset*: This is the largest publicly available dataset for ASC to the best of our knowledge. The dataset contains about 1500 minutes of acoustic scene recordings belonging to 19 classes. Each audio recording is divided into 30-second examples without overlapping, thus obtain 3026 examples in total. The sampling frequency of the audio is 22050 Hz. The dataset is provided with 20 training/testing splits. In each split, 80% of the examples are kept for training and the other 20% for testing. We use the mean average accuracy over the 20 splits as the evaluation criterion.
- *DCASE2016 dataset*: The dataset is released as Task 1 of the DCASE2016 challenge. We use the development data in this paper. The development data contains about 585 minutes of acoustic scene recordings belonging to 15 classes. Each audio recording is divided into 30-second examples without overlapping, thus obtain 1170 examples in total. The sampling frequency of the audio is 44100 Hz. The dataset is divided into 4 folds. Our experiments obey this setting, and the average performance will be reported.

4.1. Audio Pre-procession

For both datasets, the audio signal is first transformed using Short-time Fourier Transform with a frame length of 1024 and a frameshift of 220, the number of frequency filters is set to be 64. For both datasets, the examples are 30 seconds long. In the data preprocessing step, we first divide the 30-second examples into 1-second clips with 50% overlap. Then each clip is processed using neural networks in Fig.3. The classification results of all these clips will be averaged to get an ensemble result for the 30-second examples.

4.2. Hyper-parameters and Evaluation

The size of audio spectrograms is 64×128. For DNN structure in Fig.3(a), the fully connected layers can be summarized as $128 \times 128 \times 19(15)$. For CNN structure in Fig.3(b), the window sizes of convolutional layers are $64 \times 2 \times 64$, $64 \times 3 \times 64$ and $64 \times 4 \times 64$, the fully connected layers are $196 \times 128 \times 19(15)$. For LSTM structure in Fig.3(c), we use the number of LSTM cells as 128, LSTM layers as 1, the fully connected layers as $128 \times 128 \times 19(15)$. For DCASE2016 dataset, we use dropout rate of 0.5. For all these methods, the learning rate is 0.001, l_2 weight is $1e^{-4}$, training is done using the Adam [25] update method and is stopped after 100 training epochs.

In order to compute the results for each training-test split, we use the classification error over all classes. The final classification error is its average value over all splits.

4.3. Experiments without Temporal Transformer Module

We begin with experiments where we train different neural network models without temporal transformer module on both datasets. We train vanilla DNN, CNN and LSTM whose details have been given in Section 3 and Section 4.2. All networks have approximately the same number of parameters and are trained with identical optimization schemes.

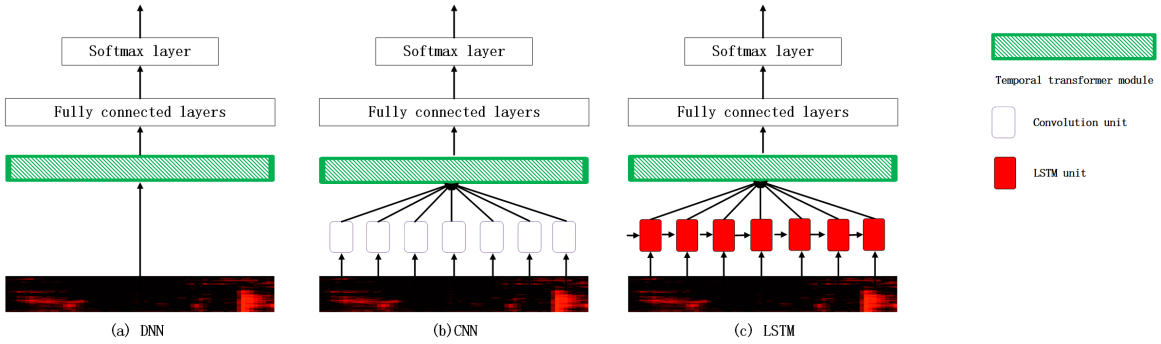


Figure 3: *Neural networks with temporal transformer module.*

The results of these experiments are shown in Table 1. Comparing with related works, our approaches on LITIS Rouen dataset achieve gains in accuracy, among which LSTM model performs much better than other models such as CNN, DNN and Nonnegative Matrix Factorization (NMF) [26] and results in state-of-the-art performance. However, on DCASE2016 dataset, LSTM model is the worst model when compared with CNN, DNN and NMF, this can be attributed to the lack of training data for the DCASE2016 dataset, where in the case of smaller training sets, matrix factorization methods such as NMF can be a good alternative to learn meaningful representations. Actually, our approach of DNN model obtains poorer performance than [27] on both datasets, mainly because of the stability of Constant Q-transform (CQT) [28] feature representations. In conclusion, our approaches of neural networks obtain excellent results on both datasets. Some results are worse on DCASE2016 dataset because of the feature extraction method, but these results do not affect our following testing of the temporal transformer module.

Table 1: *Acoustic scene classification errors using vanilla NN structures without temporal transformer module.*

Model	LITIS Rouen (%)	DCASE2016 (%)
vanilla DNN	5.30	24.3
vanilla CNN	3.21	23.1
vanilla LSTM	**2.54**	27.4
RNN-Gam [29]	3.4	-
CNN-Gam [30]	4.2	-
MFCC-GMM [3]	-	27.5
DNN-CQT [27]	3.4	21.9
Sparse-NMF [27]	5.4	**17.3**
DNN-Mel [31]	-	23.6
CNN-Mel [32]	-	24.0

4.4. Experiments with Temporal Transformer Module

We now test our temporal transformer networks on both datasets. We extend our baseline DNN, CNN and LSTM in Section 4.3 by inserting a temporal transformer module as shown in Fig.3. The number of parameters increases less than 30% for all these models.

The results of these experiments are shown in Table 2. The temporal transformer modules on DNN, CNN and LSTM all achieve performance gains. On LITIS Rouen dataset, the temporal transformer CNN obtains a new state-of-the-art result, which is even better than temporal transformer LSTM. This is because that the LSTM model itself has some capacity to deal with temporal shifting in audio data, thus the effect of temporal transformer module is less than CNN model. The temporal transformer CNN achieves an error of 1.94%, outperforming the former state-of-the-art result obtained using baseline LSTM by relatively 23.6%. On DCASE2016 dataset, all these three temporal transformer networks outperform the corresponding baseline models. Notably, the performance of temporal transformer CNN on DCASE2016 dataset reaches DNN model using CQT features in [27], meaning that the temporal transformer module makes up for the lack of feature extractions.

Table 2: *Acoustic scene classification errors using temporal transformer NN structures. TT represents the temporal transformer module.*

Model	LITIS Rouen (%)	DCASE2016 (%)
TT-DNN	2.60	22.4
TT-CNN	**1.94**	**21.8**
TT-LSTM	2.14	24.2
vanilla DNN	5.30	24.3
vanilla CNN	3.21	23.1
vanilla LSTM	2.54	27.4

5. Conclusions

In this paper, we introduce a new temporal transformer module for neural networks. This module is able to perform temporal transformations for an arbitrary length of temporal shifting in audio data and can be inserted into many existing neural networks, and can be learned in an end-to-end fashion. For commonly used DNN, CNN and LSTM structures, we see consistent gains in accuracy using temporal transformers on two acoustic scene classification datasets. On LITIS ROUEN dataset, our approach of temporal transformer CNN is able to perform significantly better than the state-of-the-art result and obtains 1.94% on classification error. This module is useful for other sequence modeling tasks and is possible to be extended to text and video classification tasks.

6. Acknowledgment

This work is partly funded by National Natural Science Foundation of China (Grant No: 61571266)

7. References

[1] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, 2015.

[2] T.-H. Lin, S.-H. Fang, and Y. Tsao, "Improving biodiversity assessment via unsupervised separation of biological sounds from long-duration recordings," *Scientific Reports*, vol. 7, no. 1, p. 4547, 2017.

[3] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *24th European Signal Processing Conference*, vol. 2016, 2016.

[4] A. Eronen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context awareness-acoustic modeling and perceptual evaluation," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, vol. 5. IEEE, 2003, pp. V–529.

[5] A. J. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 321–329, 2006.

[6] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time-frequency representations for audio scene classification," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 1, pp. 142–153, 2015.

[7] V. Bisot, S. Essid, and G. Richard, "Hog and subband power distribution image features for acoustic scene classification," in *Signal Processing Conference (EUSIPCO), 2015 23rd European*. IEEE, 2015, pp. 719–723.

[8] E. Benetos, M. Lagrange, and S. Dixon, "Characterisation of acoustic scenes using a temporally constrained shit-invariant model," in *DAFx*, 2012.

[9] V. Bisot, R. Serizel, S. Essid *et al.*, "Acoustic scene classification with matrix factorization for unsupervised feature learning," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 6445–6449.

[10] 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.

[11] Y. Petetin, C. Laroche, and A. Mayoue, "Deep neural networks for audio scene recognition," in *Signal Processing Conference (EUSIPCO), 2015 23rd European*. IEEE, 2015, pp. 125–129.

[12] N. Takahashi, M. Gygli, B. Pfister, and L. Van Gool, "Deep convolutional neural networks and data augmentation for acoustic event detection," *arXiv preprint arXiv:1604.07160*, 2016.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[14] T. S. Cohen and M. Welling, "Transformation properties of learned visual representations," *arXiv preprint arXiv:1412.7659*, 2014.

[15] K. Lenc and A. Vedaldi, "Understanding image representations by measuring their equivariance and equivalence," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 991–999.

[16] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[17] M. F. Stollenga, J. Masci, F. Gomez, and J. Schmidhuber, "Deep networks with internal selective attention through feedback connections," in *Advances in Neural Information Processing Systems*, 2014, pp. 3545–3553.

[18] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[19] P. Balazs, M. Dörfler, F. Jaillet, N. Holighaus, and G. Velasco, "Theory, implementation and applications of nonstationary gabor frames," *Journal of computational and applied mathematics*, vol. 236, no. 6, pp. 1481–1496, 2011.

[20] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications*. McGraw-Hill New York, 1986, vol. 31999.

[21] P. Welch, "The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms," *IEEE Transactions on audio and electroacoustics*, vol. 15, no. 2, pp. 70–73, 1967.

[22] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[23] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[24] W. Zaremba and I. Sutskever, "Learning to execute," *arXiv preprint arXiv:1410.4615*, 2014.

[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[26] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[27] V. Bisot, R. Serizel, S. Essid, and G. Richard, "Feature learning with matrix factorization applied to acoustic scene classification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1216–1229, 2017.

[28] J. C. Brown, "Calculation of a constant q spectral transform," *The Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991.

[29] H. Phan, P. Koch, F. Katzberg, M. Maass, R. Mazur, and A. Mertins, "Audio scene classification with deep recurrent neural networks," *arXiv preprint arXiv:1703.04770*, 2017.

[30] H. Phan, L. Hertel, M. Maass, P. Koch, R. Mazur, and A. Mertins, "Improved audio scene classification based on label-tree embeddings and convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1278–1290, 2017.

[31] Q. Kong, I. Sobieraj, W. Wang, and M. Plumbley, "Deep neural network baseline for dcase challenge 2016," *Proceedings of DCASE 2016*, 2016.

[32] D. Battaglino, L. Lepauloux, N. Evans, F. Mougins, and F. Biot, "Acoustic scene classification using convolutional neural networks," *DCASE2016 Challenge, Tech. Rep.*, 2016.