LANGUAGE MODELLING FOR SOUND EVENT DETECTION WITH TEACHER FORCING AND SCHEDULED SAMPLING

Konstantinos Drossos, Shayan Gharib, Paul Magron, and Tuomas Virtanen

Audio Research Group, Tampere University, Tampere, Finland
{firstname.lastname}@tuni.fi

ABSTRACT

A sound event detection (SED) method typically takes as an input a sequence of audio frames and predicts the activities of sound events in each frame. In real-life recordings, the sound events exhibit some temporal structure: for instance, a "car horn" will likely be followed by a "car passing by". While this temporal structure is widely exploited in sequence prediction tasks (e.g., in machine translation), where language models (LM) are exploited, it is not satisfactorily modeled in SED. In this work we propose a method which allows a recurrent neural network (RNN) to learn an LM for the SED task. The method conditions the input of the RNN with the activities of classes at the previous time step. We evaluate our method using F_1 score and error rate (ER) over three different and publicly available datasets; the TUT-SED Synthetic 2016 and the TUT Sound Events 2016 and 2017 datasets. The obtained results show an increase of 6% and 3% at the F_1 (higher is better) and a decrease of 3% and 2% at ER (lower is better) for the TUT Sound Events 2016 and 2017 datasets, respectively, when using our method. On the contrary, with our method there is a decrease of 10% at F_1 score and an increase of 11% at ER for the TUT-SED Synthetic 2016 dataset.

Index Terms— sound event detection, language modelling, sequence modelling, teacher forcing, scheduled sampling

1. INTRODUCTION

Sound event detection (SED) consists in detecting the activity of classes (onset and offset times) in an audio signal, where the classes correspond to different sound events. (e.g., "baby cry", "glass shatter"). This task finds applications in many areas related to machine listening, such as audio surveillance for smart industries and cities [1, 2], smart meeting room devices for enhanced telecommunications [3, 4], or bio-diversity monitoring in natural environments [5, 6]. SED is a challenging research task since the sound events are of very diverse nature, which might be unknown a priori in real-life recordings. Besides, they often overlap in time, a problem termed as polyphonic SED. Significant advances in SED were made recently thanks to the advent of deep learning [7]. The recurrent neural network (RNN) have proven particularly promising [8, 9] as they are able to model the temporal discriminant representations for sound events. More recently, these have been stacked with convolutional layers, resulting in convolutional recurrent neural networks (CRNN) which yield state-of-the-art results [10, 11].

In real-life recordings, the various sound events likely temporal structures within and across events. For instance, a "footsteps" event might be repeated with pauses in between (intra-event structure). On the other hand, "car horn" is likely to follow or precede the "car passing by" sound event (inter-events structure). Although these temporal structures vary with the acoustic scene and the actual sound events classes, they exist and can be exploited in the SED task. Some previous studies focus on exploiting these temporal structures. For example, in [9], the authors propose to use hidden Markov models (HMMs) to control the duration of each sound event predicted with a deep neural network (DNN). Although the results show some improvement with the usage of HMMs, the approach is a hybrid one and it requires a post processing step, which might be limited compared to an non-hybrid, DNN-based approach. In [12] and [13], the connectionist temporal classification (CTC) [14] loss function is used for SED: the output of the DNN is modified in order to be used with the CTC. Although the usage of CTC seems to be promising, CTC needs modification in order to be used for SED, it is a complicated criterion to employ, and it was developed to solve the problem where there is no frame-to-frame alignment between the input and output sequences [14]. Thus, there might be the case that using a different method for SED language modelling could provide better results than CTC. Finally, in [13], the authors also employ N-grams, which require pre and post processing stages, and use the class activities as extra input features. However, the latter approach did not perform better than a baseline which did not employ any language model.

In this paper we propose an RNN-based method for SED that exploit the temporal structures within and across events of audio scenes without the aforementioned drawbacks of the previous approaches. This method is based on established practices from other scientific disciplines that deal with sequential data (e.g., machine translation, natural language processing, speech recognition). It consists in using the output of the classifier as an extra input to the RNN in order to learn a model of the temporal structures of the output sequence (referred to as language model), a technique called *teacher forcing* [15]. Besides, this extra input of the RNN is chosen as a combination of the ground truth and predicted classes. This strategy, known as *schedule sampling* [16], consists in first using the ground truth activities and further replacing them by the predictions. This allows the RNN to learn a robust language model from clean labels, without introducing any mismatch between the training and inference processes.

The rest of the paper is organized as follows. In Section 2 we present our method. Section 3 details the experimental protocol and Section 4 presents the results. Section 5 concludes the paper.

2. PROPOSED METHOD

We propose a system that consists of a DNN acting as a feature extractor, an RNN that learns the temporal structures withing and across events (i.e. a language model), and a feed-forward neural network (FNN) acting as a classifier. Since we focus on designing

https://doi.org/10.33682/1dze-8739

an RNN that is able to learn a language model over the sound events, the RNN takes as inputs the outputs of both the DNN and the FNN. The code for our method can be found online[1].

2.1. Baseline system

The DNN takes as an input a time-frequency representation of an audio signal denoted $\mathbf{X} \in \mathbb{R}_{\geq 0}^{T \times F}$, where T and F respectively denote the number of time frames and features. It outputs a latent representation:

$$\mathbf{H} = \text{DNN}(\mathbf{X}), \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{T \times F'}$ is the learned representation with F' features. Then, the RNN operates over the rows of $\mathbf{H}$ as

$$\mathbf{h}'_t = \text{RNN}(\mathbf{h}_t, \mathbf{h}'_{t-1}), \tag{2}$$

where $t = 1, 2, \ldots, T$, $\mathbf{h}'_0 = \{0\}^{F''}$, $\mathbf{h}'_t \in [-1, 1]^{F''}$, and F'' is the amount of features that the RNN outputs at each time-step. Finally, the FNN takes $\mathbf{h}'_t$ as an input and outputs the prediction $\hat{\mathbf{y}}_t$ for the time-step t as:

$$\hat{\mathbf{y}}_t = \sigma(\text{FNN}(\mathbf{h}'_t)), \tag{3}$$

where σ is the sigmoid function, and $\hat{\mathbf{y}}_t \in [0, 1]^C$ is the predicted activity of each of the C classes.

The DNN, the RNN, and the FNN are simultaneously optimized by minimizing the loss $\mathcal{L}(\hat{\mathbf{Y}}, \mathbf{Y}) = \sum_t \mathcal{L}_t(\hat{\mathbf{y}}_t, \mathbf{y}_t)$ with:

$$\mathcal{L}_t(\hat{\mathbf{y}}_t, \mathbf{y}_t) = \sum_{c=1}^{C} y_{t,c} \log(\hat{y}_{t,c}) + (1 - y_{t,c}) \log(1 - \hat{y}_{t,c}), \tag{4}$$

where $y_{t,c}$ and $\hat{y}_{t,c}$ are the ground truth and predicted activities, respectively, of the c-th class at the t-th time-step.

2.2. Teacher forcing

The modeling in Eq. (2) shows that the RNN learns according to its input and its previous state [15, 16]. In order to allow the RNN to learn a language model over the output (i.e. the sound events), we propose to inform the RNN of the activities of the classes of the sound events at the time step $t - 1$. That is, we condition the input to the RNN as:

$$\mathbf{h}'_t = \text{RNN}(\mathbf{h}_t, \mathbf{h}'_{t-1}, \mathbf{y}'_{t-1}), \tag{5}$$

where $\mathbf{y}'_{t-1}$ is the vector with the activities of the classes of the sound events at time step $t - 1$, and $\mathbf{y}'_0 = \{0\}^C$. This technique is termed as teacher forcing [15], and is widely employed in sequence prediction/generation tasks where the output sequence has an inherent temporal model/structure (e.g., machine translation, image captioning, speech recognition) [17, 18, 19]. By using this conditioning of the RNN, the RNN can learn a language model over the output tokens of the classifier [15, 16]. In SED, this results in letting the RNN learn a language model over the sound events, e.g., which sound events are more likely to happen together and/or in sequence, or how likely is a sound event to keep being active, given the previous activity of the sound events. Teacher forcing is different from what was proposed in [13], as the latter approach conditioned the DNN (not the RNN) with the class activities: such an approach yielded poor results, intuitively explained by having $\mathbf{y}'_{t-1}$ dominated by the information in $\mathbf{X}$ through the sequence of the CNN blocks.

[1]`https://github.com/dr-costas/SEDLM`

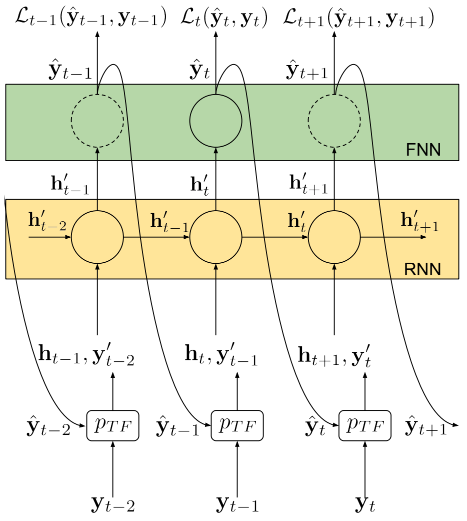

Figure 1: Proposed method of teacher forcing with scheduled sampling.

2.3. Scheduled sampling

The activity vector $\mathbf{y}'_{t-1}$ can be either the ground truth data (i.e., $\mathbf{y}_{t-1}$), or the predictions of the classifier (i.e., $\hat{\mathbf{y}}_{t-1}$). In the former case, the RNN is likely to start learning the desired language model from the first updates of the weights. At each time step t, the RNN will take as input the ground truth activities of the classes, thus being able to exploit this information from the very first weight updates. However, these ground truth values are not available at the inference stage: these would be replaced by the estimates $\hat{\mathbf{y}}_{t-1}$, which would create a mismatch between the training and testing processes. Besides, an RNN trained using only the true class activities is very likely to be sensitive to the prediction errors in $\hat{\mathbf{y}}_{t-1}$. Finally, we empirically observed that using $\mathbf{y}_{t-1}$ with the SED datasets, which are of relatively small size, results in a very poor generalization of the SED method.

A countermeasure to the above is to use the predictions $\hat{\mathbf{y}}_{t-1}$ as $\mathbf{y}'_{t-1}$, which allows the RNN to compensate for the prediction errors. However, during the first weight updates, the predicted $\hat{\mathbf{y}}_{t-1}$ is very noisy and any error created at a time step t is propagated over time, which results in accumulating more errors down the line of the output sequence. This makes the training process very unstable and is likely to yield a poor SED performance.

To exploit the best of both approaches, we propose to use the scheduled sampling strategy [16]: the ground truth class activities are used during the initial epochs, and they are further gradually replaced by the predicted class activities. This gradual replacement is based on a probability p_{TF} of picking $\mathbf{y}_{t-1}$ over $\hat{\mathbf{y}}_{t-1}$ as $\mathbf{y}'_{t-1}$ that decreases over epochs. Different functions can be used for the calculation of p_{TF} (e.g., exponential, sigmoid, linear). Here, we employ a model of exponential decrease of p_{TF}:

$$p_{\text{TF}} = \min(p_{\text{max}}, 1 - \min(1 - p_{\text{min}}, \frac{2}{1 + e^{\beta}} - 1)), \tag{6}$$

where $\beta = -i\gamma N_b^{-1}$, i is the index of the weight update (i.e., how many weight updates have been performed), N_b is the amount of batches in one epoch, and p_{max}, p_{min}, and γ are hyper-parameters to be tuned. p_{max} and p_{min} are the maximum and minimum probabilities of selecting $\hat{\mathbf{y}}_t$, and γ controls the slope of the curve of p_{TF} for a given N_b and as i increases. We use a minimum probability $p_{\min}$ because we experimentally observed that if we solely use $\mathbf{y}_{t-1}$ as $\mathbf{y}'_{t-1}$ even in the first initial weight updates, then the SED method overfits. The usage of $p_{\min}$ counters this fact. On the other hand, we use a maximum probability $p_{\max}$ in order to allow the usage of $\mathbf{y}_{t-1}$ as $\mathbf{y}'_{t-1}$ at the later stages of the learning process. We do this because the length of a sequence in SED is usually over 1000 time-steps and any error in $\hat{\mathbf{y}}_t$ is accumulated in this very long sequence, resulting in hampering the learning process. The usage of $p_{\max}$ offers a counter measure to this, by allowing the usage of ground truth values $\mathbf{y}_t$ instead of predicted and noisy values. This method is illustrated in Figure 1.

3. EVALUATION

We evaluate our method using the CRNN from [10], and we employ synthetic and real-life recordings datasets to illustrate the impact of the language model learned by our method.

3.1. Data and feature extraction

The synthetic dataset is the TUT-SED Synthetic 2016, used in [10], and consisting of 100 audio files which are synthetically created out of isolated sound events of 16 different classes. These classes are: *alarms and sirens*, *baby crying*, *bird singing*, *bus*, *cat meowing*, *crowd applause*, *crowd cheering*, *dog barking*, *footsteps*, *glass smash*, *gun shot*, *horse walk*, *mixer*, *motorcycle*, *rain*, and *thunder*. Each audio file contains a maximum of N number of randomly selected target classes, where N is sampled from the discrete uniform distribution $U(4, 9)$, and the maximum number of simultaneously active (polyphony) sound events is 5. The audio files do not contain any background noise. The audio files amount to a total of 566 minutes of audio material, and according to the splits introduced by [10], roughly 60% of the data are dedicated to training, 20% to validation, and 20% to testing split. More information about the dataset can be found online[2].

We employ two real-life recording datasets, which were part of the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge datasets for SED in real life audio task: the TUT Sound Events 2016 and the TUT Sound Events 2017 [20]. The TUT Sound Events 2016 dataset contains sound events recorded in two environments: home (indoor), which contains 11 classes, and residential area (outdoor), which contains 7 classes. The classes for the home environment are: *(object) rustling*, *(object) snapping*, *cupboard*, *cutlery*, *dishes*, *drawer*, *glass jingling*, *object impact*, *people walking*, *washing dishes*, and *water tap running*. The classes for the residential area environment are: *(object) banging*, *bird singing*, *car passing by*, *children shouting*, *people speaking*, *people walking*, and *wind blowing*. The TUT Sound Events 2017 dataset contains recordings in a street environment and contains 6 different classes. These classes are: *brakes squeaking*, *car*, *children*, *large vehicle*, *people speaking*, and *people walking*. For both datasets, we use the cross-fold validation split proposed in the DCASE 2016 and 2017 challenges. More information about the classes, the cross-fold setting, and the recordings of the datasets can be found online[3,4].

[2]http://www.cs.tut.fi/sgn/arg/taslp2017-crnn-sed/tut-sed-synthetic-2016

[3]http://www.cs.tut.fi/sgn/arg/dcase2016

[4]http://www.cs.tut.fi/sgn/arg/dcase2017/challenge

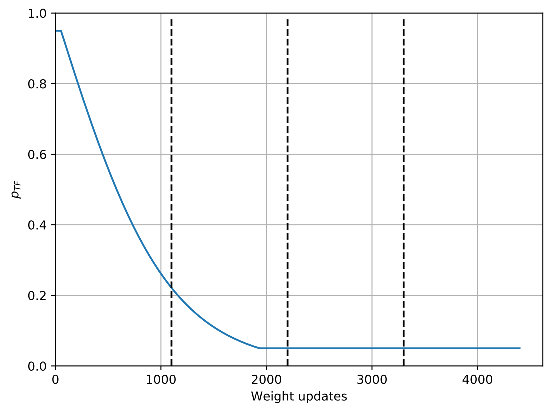

Figure 2: The value of p_{TF} with consecutive weight updates with $p_{\min} = 0.05$, $p_{\max} = 0.95$, and $N_b = 44$. The vertical dashed lines indicate steps of 25 epochs (i.e. 25, 50, 75 epochs).

The synthetic dataset has randomly selected and placed sound events, therefore not exhibiting any underlying temporal structure of sound events. We thus expect the performance of our method to be similar to a method without language modeling on the synthetic dataset. Contrarily, the real-life recording datasets exhibit some underlying temporal structures in the sound events, therefore we expect our method to perform better than a method without language modelling on these datasets.

As input features $\mathbf{X}$ we use non-overlapping sequences of $T = 1024$ feature vectors. These consist of $F = 40$ log mel-bands, extracted using a short-time Fourier transform using a 22 ms Hamming window, 50% overlap and no zero padding. We normalize the extracted feature vectors from each dataset to have zero mean and unit variance, employing statistics calculated on the training split of each corresponding dataset.

3.2. System and hyper-parameters

As our DNN we use the three convolutional neural network (CNN) blocks from the system in [10], each consisting of a CNN, a batch normalization function, a max-pooling operation, a dropout function, and a rectified linear unit (ReLU). The kernels of the CNNs are square with a width of 5, a stride of 1, and a padding of 2 in both directions. There are 128 filters for each CNN. The kernel and the stride for the first max-pooling operation are $\{1, 5\}$, for the second $\{1, 4\}$, and for the third $\{1, 2\}$. These result in $F' = 128$ for $\mathbf{H}$. All CNN blocks use a dropout of 25% at their input, and the last CNN block also uses a dropout of 25% at its output. As our RNN we use a gated recurrent unit (GRU) with $F'' = 128$ and our FNN is a single-layer feed-forward network with the output size defined according to the amount of classes in each dataset: $C = 16$ for TUT-SED Synthetic 2016, $C = 11$ and $C = 7$ for the home and residential area scenes of the TUT Sound Events 2016, and $C = 6$ for the TUT Sound Events 2017. To optimize the weights we employed the Adam optimizer [21] with default values. We employ a batch size of 8 and we stop the training when the loss for the validation data is not decreasing for 50 consecutive epochs. Finally, we set the hyper-parameters for p_{TF} at $\gamma = 12^{-1}$, $p_{\min} = 0.05$, and $p_{\max} = 0.9$. In Figure 2 is the value of p_{TF} for consecutive weight updates of $N_b = 44$ and for 100 epochs.

Empirically we observed that when using the TUT Sound

Events 2017, there are some irregular spikes of relatively high gradients in different batches during training. To alleviate this issue, we clipped the ℓ_2-norm of the gradient of all weights in each layer of our system to a value of 0.5. Additionally, we also observed that for the TUT Sound Events 2017 and TUT-SED Synthetic 2016 datasets, our method performed significantly better when we decreased the learning rate of the optimizer to $5e-4$. Therefore, we employed the above mentioned gradient clipping and modified learning rate for our method, when using the aforementioned datasets. Finally, for the TUT Sound Events 2016, we employed a binarized version of $\mathbf{y}'$ denoted $\mathbf{y}''$, such that $y''_{t,c}=1$ if $y'_{t,c}\geq 0.5$, and $y''_{t,c}=0$ otherwise.

All the above hyper-parameters were tuned using the cross validation set up for the TUT Sound Events 2016 and 2017 datasets provided by DCASE challenges, and the validation split provided in [10] for the TUT-SED Synthetic 2016 dataset.

3.3. Metrics

We measure the performance of our method using the frame based F_1 score and the error rate (ER), according to previous studies and the DCASE Challenge directions [10, 13]. For the real-life datasets, the F_1 and ER are the averages among the provided folds (and among the different acoustic scenes for the 2016 dataset), while for the synthetic dataset the F_1 and ER are obtained on the testing split. Finally, we repeat four times the training and testing process for all datasets, in order to obtain a mean and standard deviation (STD) for F_1 and ER.

3.4. Baseline

As a baseline we employ the system presented in [10], that does not exploit any language model. We do not apply any data augmentation technique during training and we use the hyper-parameters presented in the corresponding paper. This system is referred to as "Baseline".

When using our method with the TUT Sound Events 2017 and TUT-SED Synthetic 2016 datasets, we employ a modified learning rate for the optimizer and we clip the ℓ_2-norm of the gradient for all weights. To obtain a thorough and fair assessment of the performance of our method, we utilize a second baseline for this dataset: we use again the system presented in [10], but we employ the above-mentioned gradient clipping and modified learning rate. We denote this modified baseline as "modBaseline".

Finally, we compare our method to the best results presented in [13] which are obtained by employing N-grams as a post-processing to learn a language model. We report the results of this method on the TUT Sound Events 2016 datasets, as these are the only ones in the corresponding paper that are based on a publicly available dataset. It must be noted that in [13] was proposed the usage of $\mathbf{y}'_{t-1}$ as extra input features and the usage of CTC, but the results were inferior to the N-grams approach. Specifically, the per frame F_1 score was 0.02 and 0.04 lower and ER was 0.02 and 0.15 higher with the usage of $\mathbf{y}'_{t-1}$ as an extra input feature and the usage of CTC, respectively, compared to the N-grams approach.

4. RESULTS & DISCUSSION

In Table 1 are the obtained results for all the employed datasets. We remark that using the proposed language model improves the performance of SED in the real-life datasets. Specifically, for the TUT Sound Events 2016 dataset there is an improvement of 0.09 in the F_1 score and 0.07 for the ER. For the TUT Sound Events 2017, there is a 0.02 improvement in F_1 and 0.02 improvement in ER. These results clearly show that the employment of language modelling was beneficial for the SED method, when a real life datset was used. This is expected, since in a real life scenario the sound events exhibit temporal relationships. For example, "people speaking" and "people walking" or "washing dishes" and "water tap running" are likely to happen together or one after the other.

Table 1: Mean and STD (Mean/STD) of F_1 (higher is better) and ER (lower is better). For the method [13] only the mean is available.

	Baseline	**modBaseline**	**[13]**	**Proposed**
	TUT Sound Events 2016 dataset			
$\mathbf{F_1}$	0.28/0.01	–	0.29	0.37/0.02
ER	0.86/0.02	–	0.94	0.79/0.01
	TUT Sound Events 2017 dataset			
$\mathbf{F_1}$	0.48/0.01	0.49/0.01	–	0.50/0.02
ER	0.72/0.01	0.70/0.01	–	0.70/0.01
	TUT-SED Synthetic 2016 dataset			
$\mathbf{F_1}$	0.58/0.01	0.62/0.01	–	0.54/0.01
ER	0.54/0.01	0.49/0.01	–	0.61/0.02

On the contrary, from Table 1 we observe that there is a decrease in performance with our method on the synthetic data. Specifically, there is a 0.04 (or 0.08 when compared to modBaseline) decrease in F_1 and 0.07 (or 0.12 when compared to modBaseline) increase in ER. This clearly indicates that using a language model has a negative impact when the synthetic dataset is used. The sound events in the synthetic dataset do not exhibit any temporal relationships and, thus, the language model cannot provide any benefit to the SED method. We suggest that in such a scenario, the network focuses on learning a language model that does not exist in the data instead of solely trying to accurately predict the events on a frame-wise basis: this explains the drop in performance compared to the baseline method. Overall, this difference in performance between the two types of datasets strongly suggests that our method learns a language model over the activities of the sound events.

Finally, our system significantly outperforms the previous method [13] on the TUT Sound Events 2016 dataset. This shows that learning a language model is more powerful than crafting it as a post-processing.

5. CONCLUSIONS

In this paper we presented a method for learning a language model for SED. Our method focuses on systems that utilize an RNN before the the last layer of the SED system, and consists of conditioning the RNN at a time step t with the activities of sound events at the time step $t-1$. As activities for $t-1$ we select the ground truth early on the training process, and we gradually switch to the prediction of the classifier as the training proceeds over time. We evaluate our method with three different and publicly available datasets, two from real life recordings and one synthetic dataset. The obtained results indicate that with our method, the utilized SED system learned a language model over the activities of the sound events, which is beneficial when used on real life datasets.

In future work, we will conduct a more in-depth analysis of the learned language model and of the SED performance per class.

6. REFERENCES

[1] M. Crocco, M. Cristani, A. Trucco, and V. Murino, "Audio surveillance: A systematic review," *ACM Comput. Surv.*, vol. 48, no. 4, pp. 52:1–52:46, Feb. 2016.

[2] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: A system for detecting anomalous sounds," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 279–288, Jan 2016.

[3] T. Butko, F. G. Pla, C. Segura, C. Nadeu, and J. Hernando, "Two-source acoustic event detection and localization: Online implementation in a smart-room," in *Proc. European Signal Processing Conference*, Aug 2011.

[4] C. Busso, S. Hernanz, Chi-Wei Chu, Soon-il Kwon, Sung Lee, P. G. Georgiou, I. Cohen, and S. Narayanan, "Smart room: participant and speaker localization and identification," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, March 2005.

[5] B. J. Furnas and R. L. Callas, "Using automated recorders and occupancy models to monitor common forest birds across a large geographic region," *The Journal of Wildlife Management*, vol. 79, no. 2, pp. 325–337, 2015.

[6] T. A. Marques, L. Thomas, S. W. Martin, D. K. Mellinger, J. A. Ward, D. J. Moretti, D. Harris, and P. L. Tyack, "Estimating animal population density using passive acoustics," *Biological Reviews*, vol. 88, no. 2, pp. 287–309, 2013.

[7] E. Benetos, D. Stowell, and M. D. Plumbley, *Approaches to Complex Sound Scene Analysis*. Springer International Publishing, 2018, pp. 215–242.

[8] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016.

[9] T. Hayashi, S. Watanabe, T. Toda, T. Hori, J. Le Roux, and K. Takeda, "Duration-controlled LSTM for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 11, pp. 2059–2070, November 2017.

[10] E. Çakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, June 2017.

[11] S. Adavanne, P. Pertilä, and T. Virtanen, "Sound event detection using spatial features and convolutional recurrent neural network," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017.

[12] Y. Wang and F. Metze, "A first attempt at polyphonic sound event detection using connectionist temporal classification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017.

[13] G. Huang, T. Heittola, and T. Virtanen, "Using sequential information in polyphonic sound event detection," in *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, Sep. 2018, pp. 291–295.

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: ACM, 2006, pp. 369–376. [Online]. Available: http://doi.acm.org/10.1145/1143844.1143891

[15] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, June 1989.

[16] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 1171–1179. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969239.2969370

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3104–3112. [Online]. Available: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

[19] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3156–3164.

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

[21] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, May 2015.