CONVOLUTIONAL RECURRENT NEURAL NETWORKS FOR WEAKLY LABELED SEMI-SUPERVISED SOUND EVENT DETECTION IN DOMESTIC ENVIRONMENTS

Technical Report

Janek Ebbers, Reinhold Haeb-Umbach

Paderborn University, Department of Communications Engineering, Paderborn, Germany
{ebbers, haeb}@nt.upb.de

ABSTRACT

In this report we present our system for the *Detection and Classification of acoustic scenes and events (DCASE) 2020 Challenge Task 4: Sound event detection in domestic environments*. We present a convolutional recurrent neural network (CRNN) with two recurrent neural network (RNN) classifiers sharing the same preprocessing convolutional neural network (CNN). Both recurrent networks perform audio tagging. One is processing the input audio signal in forward direction and the other in backward direction. The networks are trained jointly using weakly labeled data, such that at each time step an active event is tagged by at least one of the networks given that the event is either in the past captured by the forward RNN or in the future captured by the backward RNN. This way the models are encouraged to output tags as soon as possible. After training, the networks can be used for sound event detection by applying them to smaller audio segments, e.g. 200 ms. Further we propose a tag conditioned CNN as a second stage which is supposed to refine sound event detection. Given its receptive field and file tags as input it performs strong label prediction trained using pseudo strong labels provided by the CRNN system. By ensembling four CRNNs and four CNNs we obtain an event-based F-score of 48.3% on the validation set, which is 13.5% higher than the baseline. We are going to release the source code at `https://github.com/fgnt/pb_sed`.

Index Terms— audio tagging, event detection

1. CONVOLUTIONAL RECURRENT NEURAL NETWORKS FOR AUDIO TAGGING

Our systems input $\mathbf{X}$ is a 128 dimensional log-mel spectrogram using a short-time Fourier transform (STFT) with a hop-size of 20 ms, a frame length of 60 ms and a sampling rate of 16 kHz. Waveforms are initially normalized to be within the range of -1 and 1: $x(t) = s(t)/\max(|s(t)|)$. Each mel bin of the log-mel spectrogram is globally normalized to zero mean and unit variance.

During training various data augmentation techniques are used, namely random scaling, mixup, frequency warping, blurring, time masking, frequency masking and random noise. Random scaling and mixup [1] are performed on the waveform similar as in [2] by shifting and superposing signals as follows:

$$x_i'(t) = \sum_{j=0}^{J-1} \lambda_j x_j(t - \tau_j)$$

with τ_j being uniformly sampled such that $x_i'(t)$ is not longer than 15 s, $\lambda_j \sim \text{LogNormal}(0, 1)$ and

$$\Pr(J) = \begin{cases} 1/3 & J = 1, \\ 2/3 & J = 2, \\ 0 & \text{else.} \end{cases}$$

Note the difference to original mixup [1] as we do not do an interpolation of signals but a superposition. Therefore, we also do not interpolate the targets but combine them into a single multi-hot target vector. Frequency warping and time- and frequency masking are performed exactly as in [2]. Blurring is performed with a gaussian blur kernel of size 5×5 with $\sigma_b \sim \text{Exp}(0.5)$. Finally random gaussian noise with a standard deviation of $\sigma_n \sim \text{Uniform}(0, 0.2)$ is added to the log-mel spectrogram.

The input is forwarded through the CNN $\mathbf{H} = f_{\text{cnn}}(\mathbf{X})$. The CNN architecture is shown in Tab. 1.

Table 1: CNN architecture as in [2] but without temporal pooling. Each ConvXd uses a kernel size of three and a stride of one and includes BatchNorm [3] and ReLU.

Block	output shape
LogMel(128)	$1\times128\times N$
GlobalNorm	$1\times128\times N$
$2\times$Conv2d(16)	$16\times128\times N$
Pool2d(2×1)	$16\times64\times N$
$2\times$Conv2d(32)	$32\times64\times N$
Pool2d(2×1)	$32\times32\times N$
$2\times$Conv2d(64)	$64\times32\times N$
Pool2d(2×1)	$64\times16\times N$
$2\times$Conv2d(128)	$128\times16\times N$
Pool2d(2×1)	$128\times8\times N$
Conv2d(256)	$256\times8\times N$
Pool2d(2×1)	$256\times4\times N$
Reshape	$1024\times N$
$3\times$Conv1d(256)	$256\times N$

Table 2: Recurrent classifier architecture as in [2].

Block	output shape
$2\times$GRU(256)	$256\times N$
$\text{fc}_{\text{ReLU}}(256)$	$256\times N$
$\text{fc}_{\text{Sigmoid}}(10)$	$10\times N$

This work has been supported by Deutsche Forschungsgemeinschaft under contract no. HA 3455/15-1 within the Research Unit FOR 2457.

We then perform recurrent forward tagging $\mathbf{Y}^{\text{fwd}} = f_{\text{rnn}}^{\text{fwd}}(\mathbf{H})$ and backward tagging $\widetilde{\mathbf{Y}}^{\text{bwd}} = f_{\text{rnn}}^{\text{bwd}}(\widetilde{\mathbf{H}})$, with $\widetilde{\mathbf{H}}$ denoting time flipped $\mathbf{H}$. The recurrent architecture is shown in Tab. 2. During training tag predictions are computed at each time frame as $\mathbf{Y} = \max(\mathbf{Y}^{\text{fwd}}, \mathbf{Y}^{\text{bwd}})$ with $\max(\cdot)$ denoting point wise maximum operation. We hypothesize that forward and backward classifiers jointly have to be able to perform tagging at each time frame as the forward classifier has seen all sound events between start and current frame while the backward classifier has seen all sound events between end and current frame. This way we encourage the classifiers to tag sound events as soon as they appear. At inference time tag predictions are obtained as $\hat{\mathbf{y}} = f_{\text{tag}}(\mathbf{X}) = (\mathbf{Y}^{\text{fwd}}[N-1] + \mathbf{Y}^{\text{bwd}}[0])/2$. Event-specific thresholds λ_k are used to get binary predictions

$$\hat{z}_k = \begin{cases} 1 & \hat{y}_k > \lambda_k, \\ 0 & \text{else.} \end{cases}$$

During training on-the-fly tagging of unlabeled data is performed. If each event score either exceeds a event-specific detection threshold or falls below a exclusion-threshold, the audio file is included in training. Detection- and exclusion-thresholds are newly determined on the validation set every 1000-th update step. The threshold with the best F-score constrained to precisions $>90\%$ is chosen as detection threshold and the threshold with the best harmonic mean of specificity and negative predictive value (the F-score of negatives so to say) constrained to negative predictive values $\text{NPV} > 98\%$ is chosen as exclusion threshold. At test-time we use decision thresholds which give the best (unconstrained) F-scores on the validation set.

To get frame predictions we apply tagging to a small context around each frame: $\hat{\mathbf{y}}_n = f_{\text{tag}}(\mathbf{X}_{n-C:n+C}) \cdot \hat{\mathbf{z}}$. We use an ensemble of four independently trained CRNNs, a context of $C = 20$ and a median filter size of $M = 41$ frames to compute pseudo strong labels for the weakly labeled and unlabeled portions of the training set using decision thresholds which give best frame F-scores on the validation set. At test time different contexts $C \in \{5, 10\}$ and median filter sizes $M \in \{21, 41\}$ are used for different events.

2. TAG CONDITIONED CONVOLUTIONAL NEURAL NETWORK FOR SOUND EVENT DETECTION

Given the predicted pseudo strong labels from the CRNN-ensemble we further train a CNN $\hat{\mathbf{y}}_n = f_{\text{sed}}(\mathbf{X}_{n-R:n+R}, \hat{\mathbf{z}})$ to perform frame-wise sound event detection (SED) with $R = 13$ being the one-sided receptive field of the CNN. The CNN architecture is similar as in Tab. 1 with the last Conv1d layer outputting $K = 10$ scores, one for each event class. Additionally to the (augmented) log-mel input spectrogram, this second-stage CNN is conditioned on the file tags by concatenating the multi-hot tag encoding $\hat{\mathbf{z}}$ to each time-frequency bin along the channel dimension of the log mel spectrogram as well as to the hidden representation between the reshape operation and the first Conv1d layer. We hypothesize that the CNN can do better event detection if it is aware about the active events within the file, i.e. when it has to predict $\Pr(\text{event active in frame}|\text{event active in file})$ rather than $\Pr(\text{event active in frame})$. However, we add some noise to the concatenated tags to prevent the CNN from simply forwarding the tags. Note that we have not yet explored the effectiveness of the conditioning. For more insides you may have a look at our workshop paper we are going to submit.

We independently train four tag conditioned CNNs to perform frame wise event detection. Final scores are obtained as the average of the four CRNNs and four tag conditioned CNNs masked by the file tag predictions and subsequent median filtering with event-specific filter size $M \in \{21, 41\}$, which are determined on the validation set.

3. TRAINING DETAILS

The training criterion in all trainings is the binary cross entropy between the model predictions $\hat{\mathbf{y}}$ and the multi-hot target vector $\mathbf{z}$:

$$L(\hat{\mathbf{y}}, \mathbf{z}) = -\sum_{k=0}^{K-1} \Big(z_k \log(y_k) + (1 - z_k) \log(1 - y_k) \Big)$$

with $K = 10$ denoting the number of target event classes.

We randomly sample mini batches of size 24 from the training data such that no signal in the mini batch is padded by more than 5%. We compute gradients of the average loss in the mini batch with gradient clipping at a threshold of 20. Adam [4] is used for optimization with a learning rate of $5 \cdot 10^{-4}$. Training is performed for 20000 update steps, with validation every 1000-th update step where the checkpoint with the best validation F-score is chosen as final model.

4. RESULTS

Tab. 3 shows final validation performance as well as C and M for each event class.

Table 3: Validation performance.

Model	Event	C	M	F-score [%]
Baseline	macro-average	-	-	34.8
Ours	macro-average	-	-	48.3
	Alarm bell ringing	5	21	45.9
	Blender	5	41	54.4
	Cat	5	21	49.3
	Dishes	5	21	22.8
	Dog	5	21	27.0
	Electr. shaver/toothbrush	10	41	57.9
	Frying	5	41	52.6
	Running water	5	41	41.2
	Speech	5	21	50.5
	Vacuum cleaner	10	41	80.9

5. CONCLUSIONS

In this report we presented our system for the DCASE 2020 Challenge: Sound event detection in domestic environments. We proposed a weakly labeled sound event detection method trained to perform forward as well as backward tagging which encourages the model to predict labels with low latency. This makes the model suitable to also be applied to small segments of only a few hundred milliseconds. Hence, the proposed approach presents a competitive alternative/complement to the widespread use of attention methods. We further suggest to use a second stage of CNNs which are conditioned on the file tags to refine sound event detection. The overall system is able to outperform the baseline by 13.5%.

6. REFERENCES

[1] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[2] J. Ebbers and R. Hb-Umbach, "Convolutional recurrent neural network and data augmentation for audio tagging with noisy labels and minimal supervision," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, New York University, NY, USA, October 2019, pp. 64–68.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

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