SOUND SOURCE DETECTION, LOCALIZATION AND CLASSIFICATION USING CONSECUTIVE ENSEMBLE OF CRNN MODELS

Sławomir Kapka, Mateusz Lewandowski*

Samsung R&D Institute Poland
Artificial Intelligence
Warsaw, 00-844, Poland
{s.kapka, m.lewandows4}@samsung.com

ABSTRACT

In this paper, we describe our method for DCASE2019 task 3: Sound Event Localization and Detection (SELD). We use four CRNN SELDnet-like single output models which run in a consecutive manner to recover all possible information of occurring events. We decompose the SELD task into estimating number of active sources, estimating direction of arrival of a single source, estimating direction of arrival of the second source where the direction of the first one is known and a multi-label classification task. We use custom consecutive ensemble to predict events' onset, offset, direction of arrival and class. The proposed approach is evaluated on the TAU Spatial Sound Events 2019 - Ambisonic and it is compared with other participants' submissions.

***Index Terms*—** DCASE 2019, Sound Event Localization and Detection, CRNN, Ambisonics

1. INTRODUCTION

Sound Event Localization and Detection (SELD) is a complex task which naturally appears when one wants to develop a system that possesses spatial awareness of the surrounding world using multi-channel audio signals. This year, the task 3 from the IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2019) [1] concerned the SELD problem. SELDnet introduced in [2] is a single system of a good quality designed for the SELD task, and the slight modification of SELDnet was set as the baseline system [3] during the DCASE 2019 Challenge. Solely based on [2] and [3], we develop a novel system designed for the task 3 from the DCASE 2019 Challenge.

In our work, we follow the philosophy that if a complex problem can be split into simpler ones, one should do so. Thus we decompose the SELD task with up to 2 active sound sources into the following subtasks:

- estimating the number of active sources (*noas*),
- estimating the direction of arrival of a sound event when there is one active sound source (*doa1*),
- estimating the direction of arrival of a sound event when there are two active sound sources and we posses the knowledge of the direction of arrival of one of these sound events, which we will call an *associated event* (*doa2*),
- multi-label classification of sound events (*class*).

*Corresponding author.

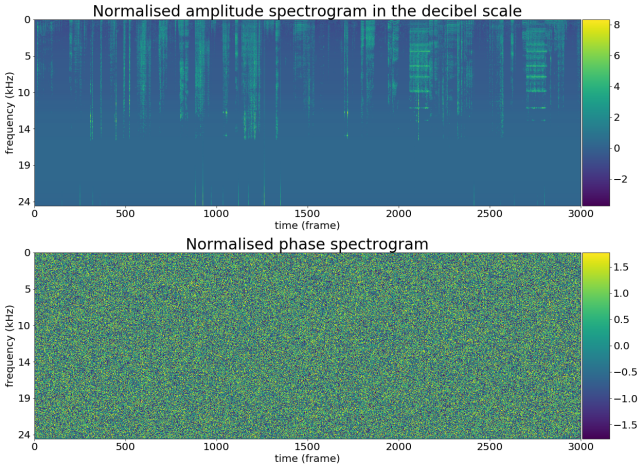

Figure 1: An example of the normalised amplitude spectrogram in the decibel scale and the normalised phase spectrogram obtained from the first *foa* channel from some randomly selected recording. The horizontal and vertical axes denote frame numbers and frequencies respectively obtained from the STFT. Note that the values from the legends on the right are dimensionless due to the normalization used in the preprocessing.

For each of this subtasks, we develop a SELDnet-like convolutional recurrent neural network (CRNN) with a single output. We discuss it in detail in section 3. Given such models, we develop a custom consecutive ensemble of these models. This allows us to predict the events' onset, offset, direction of arrival and class, which we discuss in detail in section 4. Due to the sequential nature of generating predictions in our system, errors in models' predictions may cascade, and thus an overall error may cumulate. Despite this drawback, our system acquire very good results on the TAU Spatial Sound Events 2019 - Ambisonic database. We discuss the results in detail in section 5.

2. FEATURES

The DCASE 2019 task 3 provides two formats of the TAU Spatial Sound Events 2019 dataset: first order ambisonic (*foa*) and 4 channels from a microphone array (*mic*) [3]. In our method we only use the ambisonic format.

https://doi.org/10.33682/9f2t-ab23

Each recording is approximately 1 minute long with sampling rate of 48k. We use the short time Fourier transform (STFT) with Hann window. We use the window of length 0.4s and hop of length 0.2s in STFT to transform a raw audio associated to each *foa* channel into the complex spectrogram of size 3000x1024. If audio is longer than 1 minute, we truncate spectrograms. If an audio is shorter than 1 minute, we pad them with zeros.

From each complex spectrogram we extract its module and phase point-wise, that is amplitude and phase spectrograms, respectively. We transform amplitude spectrograms to the decibel scale. Finally, we standardize all spectrograms frequency-wise to zero mean and unit variance, to obtain spectrograms as in Figure 1.

In summary, from each recording we acquire 4 standardized amplitude spectrograms in the decibel scale and 4 standardized phase spectrograms corresponding to 4 *foa* channels.

3. ARCHITECTURE

As mentioned in the introduction, each of the subtasks (*noas*, *doa1*, *doa2* and *class*) has its own SELDnet-like CRNN. Each of these models is a copy of a single SELDnet node with just minor adjustments so that it fits to the specific subtask and for the regularization purpose.

Each of these models takes as an input a fixed length subsequence of decibel scale amplitude spectrograms (in case of *noas* and *class* subtasks) or both decibel scale amplitude and phase spectrograms (in case of *doa1* and *doa2* subtasks) from all 4 channels.

In each case, the input layers are followed by 3 convolutional layer blocks. Each block is made of a convolutional layer, batch norm, relu activation, maxpool and dropout. The output from the last convolutional block is reshaped so that it forms a multivariate sequence of a fixed length. In the case of *doa2*, we additionaly concatenate directions of arrivals of associated events with this multivariate sequence. Next, there are two recurrent layers (GRU or LSTM) with 128 units each with dropout and recurrent dropout. Next layer is a time distributed dense layer with dropout and with the number of units depending on subtask.

Lastly, depending on a subtask, the model has a different output. For *noas*, the model has just a single time distributed output that corresponds to the number of active sources (0, 1 or 2). For *doa1* and *doa2*, the models have 3 time distributed outputs that corresponds to cartesian xyz coordinates as in [2]. Cartesian coordinates are advantageous over spherical coordinates in this task due to their continuity. Lastly, for *class*, the model has 11 time distributed outputs corresponding to 11 possible classes. We present the detailed architecture in Table 1.

Depending on a subtask, we feed the network with the whole recordings or just their parts. For *noas*, we feed all the data. For *doa1*, we extract only those parts of the recordings where there is just one sound source active. For *doa2*, we extract only those parts of the recordings where there are exactly two active sound sources. For *class*, we extract those parts of the recordings where there are at least one active source.

As for the learning process, we used mean square error loss for the *noas*, *doa1*, *doa2* subtasks and binary cross-entropy loss for the *class* subtask. For all subtasks we initialised learning process using Adam optimizer with default parameters [4]. The *noas* and *class* subtasks were learned for 500 epochs with exponential learning rate decay; every 5 epochs the learning rate were multiplied by 0.95. In *doa1* and *doa2* subtasks, we run learning process for 1000 epochs without changing the initial learning rate.

As for complexity, the *noas*, *doa1*, *doa2* and *class* have 572,129, 753,603, 591,555 and 572,299 parameters respectively, making total of 2,651,634 parameters.

4. CONSECUTIVE ENSEMBLE

In this section, we introduce and describe the idea of the consecutive ensemble which is the core of our approach. This custom binding of our four models allows us to predict the events' onset, offset, direction of arrival and class.

4.1. The algorithm

We assume that recordings have at most 2 active sound sources at once and the sound events occur on a 10 degrees resolution grid. In our setting, the audios after feature extraction have exactly 3000 vectors corresponding to the time dimension. Henceforth we will call these vectors as frames. The algorithm itself goes as follows:

1. We feed the features to the *noas* network to predict the number of active sources (NOAS) in each frame.

2. We transform the predicted NOAS so that each recording starts and ends with no sound sources and the difference of NOAS between each frames is no greater than 1.

3. From the predicted NOAS we deduce the number of events, their onsets and the list of possible offsets for each event. If NOAS in two consecutive frames increases, then we predict that a new event happened at the second frame. If in two consecutive frames NOAS decreases, then we append the first frame to all events since last time NOAS was 0 as a possible offset.

4. In order to determine which offset corresponds to which event we use the *doa1* network. We extract chunks (intervals of equal NOAS) of audio where the predicted NOAS equals 1 and we feed it to *doa1* network. For each chunk where NOAS was 1 we predict the average azimuth and elevation, and we round it to the closest multiple of 10. If two consecutive chunks have the same azimuth and elevation then we conclude that the first event covered two chunks and the second event started and ended between those chunks. If two consecutive chunks have a different azimuth or elevation, then we conclude that the first event ended when the second chunk started and the second event continued in the second chunk.

5. To determine the remaining information about angles we need to predict the direction of arrival (DOA) of events that start and end while the associated event is happening. We feed the chunks where NOAS is 2 to the *doa2* network with the second input being DOA of the associated event in cartesian xyz coordinates. Similarly as in step 4, we average the predicted results from chunks and round it to the closest multiple of 10.

6. Lastly, we predict the events' classes. If an event has chunks where the event is happening in an isolation (NOAS = 1), then all such chunks are feed to the *class* network and the most probable class (using soft voting among frames) is taken as a predicted class. If an event has no such chunks, i.e. the event is only happening with an associated event, then such chunk (NOAS = 2) is fed to the network and two most probable classes are extracted. We choose the first one which does not equal to the class of the associated event.

4.2. An example

The algorithm itself may seem quite complex at first glance. Hence, we investigate here a concrete example.

Table 1: The architecture and the parameters of the networks

Layer Type	**Parameters**	*noas*	*doa1*	*doa2*	*class*
Input	Shape	$256 \times 1024 \times 4$	$128 \times 1024 \times 8$	$128 \times 1024 \times 8$	$128 \times 1024 \times 4$
*ConvBlock**	Pool	8	8	8	8
*ConvBlock**	Pool	8	8	8	8
*ConvBlock**	Pool	4	4	4	4
Reshape	Sequence length $\times$ features	256×-1	128×-1	128×-1	128×-1
Doa2 input	Is used	False	False	True	False
Concatenate	Is used	False	False	True	False
*RecBlock***	Unit type	GRU	LSTM	GRU	GRU
*RecBlock***	Unit type	GRU	LSTM	GRU	GRU
TD Dense	Number of units	16	128	128	16
Dropout	Dropout rate	0.2	0.2	0.2	0.2
TD Dense	Number of units	1	3	3	11
Activation	Function	linear	linear	linear	sigmoid
**ConvBlock*(P)*					
Conv2D	64 filters, 3×3 kernel, 1×1 stride, same padding				
BatchNorm	—				
Activation	ReLu function				
MaxPooling2D	$1 \times P$ pooling				
Dropout	0.2 dropout rate				
***RecBlock*(U)*					
Recurrent	128 recurrent units of type U, 0.2 recurrent dropout rate				
Activation	$\tanh$ function				
Dropout	0.2 dropout rate				

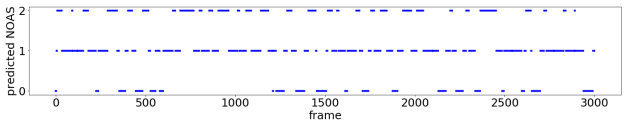

Figure 2: The plot visualising the predicted number of active sources for some randomly selected recording.

Given a recording constituting of 3000 vectors, we predict its NOAS in each frame as in Figure 2. For the sake of clarity we constrain only to a part of the recording. Consider a block with predicted NOAS as in the top plot from Figure 3. According to the step 3 from the algorithm, we predict that 3 events happened here: E_1, E_2, E_3 with 3 corresponding onsets On_1, On_2, On_3. Events E_1 and E_2 may end at Off_1, Off_2 or Off_3 and event E_3 may end at Off_2 or Off_3 (see the bottom plot from Figure 3). According to the step 4 from the algorithm, we predict DOA using *doa1* in chunks from On_1 to On_2, from Off_1 to On_3 and from Off_2 to Off_3. Based on that we deduce the events' offsets as in Figure 3. Based on step 5 from the algorithm, we predict the DOA of chunk from On_3 to Off_2 using *doa2* where the associated DOA is the DOA of E_2. Lastly we deduce classes of the events E_1, E_2 and E_3. According to the step 6 form the algorithm, we predict class of E_1 based on the chunk from On_1 to On_2, predict the class of E_2 based on chunks from Off_1 to On_3 and from Off_2 to Off_3. Finally, we predict the class of E_3 based on the chunk from On_3 to Off_2. If the predicted class of E_3 is the same as the class of E_2 then we predict it to be the second most probable class from the *class* network.

Table 2: The average results from all 4 splits.

	Error rate	F-score	DOA error	Frame recall	Seld score
Train	0.03	0.98	2.71	0.98	0.02
Val.	0.15	0.89	4.81	0.95	0.08
Test	0.14	0.90	4.75	0.95	0.08
Baseline	0.34	0.80	28.5	0.85	0.22

5. RESULTS

We evaluate our results on TAU Spatial Sound Events 2019 - Ambisonic dataset. This dataset constitutes of two parts: the development and evaluation sets. The development part consists of 400 recordings with predefined 4-fold cross-validation and the evaluation part consists of 100 recordings. The results from this section relate to our submission `Kapka_SRPOL_task3_2`.

5.1. Development phase

As for the development part, we used 2 splits out of 4 for training for every fold using the suggested cross-validation even though validation splits do not influence the training process.

We show in Table 2 the averaged metrics from all folds for our setting and metrics for the baseline [3]. In order to demonstrate the variance among folds, we present in Table 3 the detailed results on the test splits from each fold. The development set provides the distinction for the files where there is up to 1 active sound source at once (ov1) and where there are up to 2 (ov2). In Table 4 we compare metrics for the ov1 and ov2 subsets.

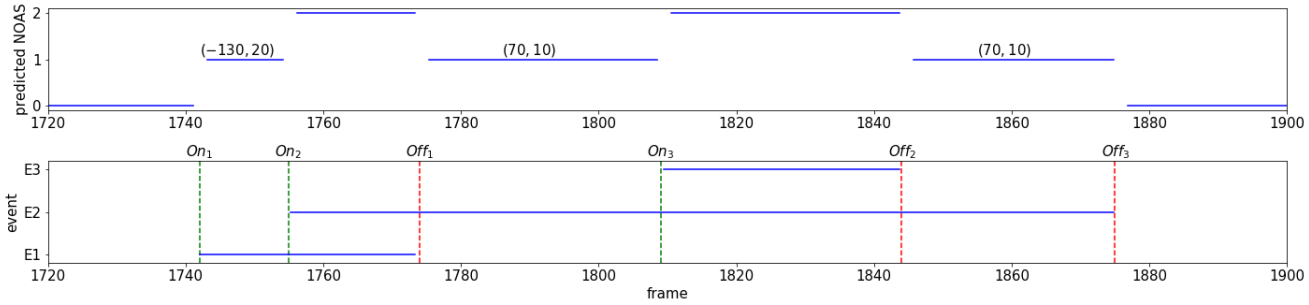

Figure 3: Given the predicted NOAS from the part of some recording as in the top plot, we deduce that there are 3 events E_1, E_2 and E_3 with corresponding onsets denoted by the green lines in the bottom plot. Based on the predicted DOA, which we placed in the top plot above the segments, we deduce the events' offsets denoted by the red lines in the bottom plot.

Table 3: The results on the test splits from each fold.

	Error rate	F-score	DOA error	Frame recall	Seld score
Split 1	0.13	0.91	6.01	0.95	0.07
Split 2	0.16	0.88	6.01	0.95	0.09
Split 3	0.11	0.93	4.93	0.96	0.06
Split 4	0.17	0.86	5.89	0.96	0.10

Table 4: The results on the ov1 and ov2 subsets.

	Error rate	F-score	DOA error	Frame recall	Seld score
ov1	0.07	0.94	1.28	0.99	0.04
ov2	0.18	0.87	7.96	0.93	0.11

5.2. Official results

For the evaluation part, we used all 4 splits for training from the development set. We compare our final results with the selected submissions in Table 5.

The idea of decomposing the SELD task into simpler ones proved to be a very popular idea among contestants. The recent two-stage approach to SELD introduced in [5] was used and developed further by many. The best submission using two-step approach `Cao_Surrey_task3_4` [6] obtained results very similar to ours. `He_THU_task3_2` [7] and `Chang_HYU_task3_3` [8] outperform our submission in SED metrics and DOA error respectively. However, our approach based on estimating NOAS first allows us to outperform all contestants in frame recall.

6. SUBMISSIONS

Overall, we created 4 submissions for the competition:

- ConseqFOA (`Kapka_SRPOL_task3_2`),
- ConseqFOA1 (`Kapka_SRPOL_task3_3`),
- ConseqFOAb (`Kapka_SRPOL_task3_4`),
- MLDcT32019 (`Lewandowski_SRPOL_task3_1`).

Table 5: The comparison of the selected submissions.

Rank	Submission name	Error rate	F-score	DOA error	Frame recall
1	`Kapka_SRPOL_task3_2`	0.08	94.7	3.7	**96.8**
4	`Cao_Surrey_task3_4`	0.08	95.5	5.5	92.2
6	`He_THU_task3_2`	**0.06**	**96.7**	22.4	94.1
19	`Chang_HYU_task3_3`	0.14	91.9	**2.7**	90.8
48	`DCASE2019_FOA_baseline`	0.28	85.4	24.6	85.7

The first three submissions use the approach described in the above sections. The only difference is that ConseqFOA is trained on all four splits from development dataset. ConseqFOA1 is trained on splits 2,3,4. ConseqFOAb is trained on all splits but the classifier in this version was trained using categorical cross-entropy instead of binary cross-entropy loss.

Our MLDcT32019 submission uses a different approach. It works in the same way as the original SELDnet architecture but with the following differences:

- We implemented the Squeeze-and-Excitation block [9] after the last convolutional block. We pass the output from the last convolutional block through two densely connected neural layers with respectively 1 and 4 neurons, we multiply it with the output of the last convolutional block and we pass it further to recurrent layers.
- We set all dropout rates to 0.2.
- We used SpecAugment [10] as an augmentation technique to double the training dataset.
- We replaced recurrent layer GRU units with LSTM units.

7. CONCLUSION

We conclude that decomposing the SELD problem into simpler tasks is instinctive and efficient. However, we are aware that our solution has some serious limitations and it fails when one wants to consider a more general setup. For example when there are more than 2 active sources at once or when the grid resolution is more refined. Thus, we claim that the pursuit for universal and efficient SELD solutions is still open.

8. ACKNOWLEDGEMENT

We are most grateful to Zuzanna Kwiatkowska for spending her time on careful reading with a deep understanding the final draft of this paper.

9. REFERENCES

[1] http://dcase.community/challenge2019/task-sound-event-localization-and-detection.

[2] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–1, 2018.

[3] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and uetection," in *Submitted to Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019. [Online]. Available: https://arxiv.org/abs/1905.08546

[4] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[5] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. D. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy," 2019. [Online]. Available: https://arxiv.org/abs/1905.00268

[6] Y. Cao, T. Iqbal, Q. Kong, M. Galindo, W. Wang, and M. Plumbley, "Two-stage sound event localization and detection using intensity vector and generalized cross-correlation," 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Cao_74.pdf

[7] J. Zhang, W. Ding, and L. He, "Data augmentation and prior knowledge-based regularization for sound event localization and detection," 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_He_97.pdf

[8] K. Noh, C. Jeong-Hwan, J. Dongyeop, and C. Joon-Hyuk, "Three-stage approach for sound event localization and detection," 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Chang_81.pdf

[9] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018, pp. 7132–7141.

[10] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," 2019. [Online]. Available: https://arxiv.org/abs/1904.08779