Deep Neural Network for 3D Audio

Andrea Carlesimo 1618599, Simone Faricelli 1647406

Neural Networks 2018-2019

May 13, 2019

Dataset analysis

- Binaural audio, TUTSED 2017
- street context audio at 24 bit and 44.1 kHz as sampling rate
- manual annotation of 6 sound event classes: brakes squeaking, car, children, large vehicle, people speaking, and people walking

Feature Extraction

- Log mel-band energy (**MBE**)
  Intra-Channel Features

- Generalized cross correlation with phase transform (**GCC**)
  Inter-Channel Features

Feature Extraction - MBE

- 40 ms windows with 50% overlap, so the hop-lenght parameter is half of nfft.
- 40 mel-bands in the frequency range of 0-22050 Hz.
- For a sequence length of T frames, the mbe feature has a general dimension of Tx40xChannels

Feature Extraction - MBE

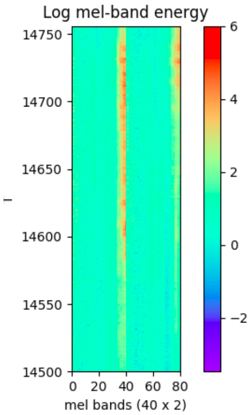


Figure: Log mel-band energy (Tx40x2)

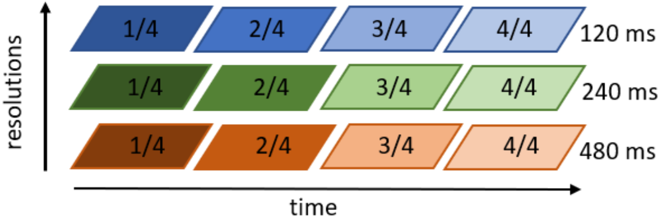
Feature Extraction - GCC

The SED methods can benefit with gcc for overlapping sound events. We extract gcc in three resolutions, 120, 240, and 480 ms as:

$$R(\Delta_{12}, t) = \sum_{k=0}^{K-1} \frac{X_1(k,t) \cdot X_2^*(k,t)}{|X_1(k,t)||X_2(k,t)|} \exp \frac{i2\pi k \Delta_{12}}{K} \tag{1}$$

- $X_1(k,t)$ and $X_2(k,t)$ are the FFT coefficients of the two channels.
- K the total frequency bin.
- $\Delta_{12} \in [-29, 30]$ is the range of travel delay between the two microphones.
- For a sequence length of T frames, the gcc feature has a general dimension of Tx60xResolutions

Feature Extraction - Multiprocessing GCC



- 12 parallel processes
- full CPU power
- reduced GCC extraction time

Feature Extraction - GCC

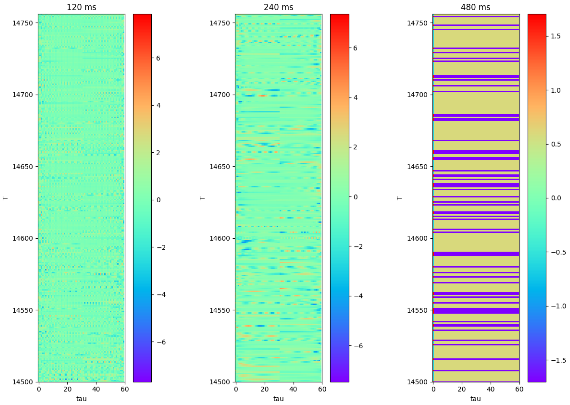


Figure: Generalized cross-correlation (Tx60x3)

Neural Network

- Kernels size: 64
- Filters dimension: 2x3x3, 3x3 and 3x3 mbe
- Filters dimension: 3x3x3, 3x3 and 3x3 gcc
- Max pooling on mel-bands: 5,2,2
- Max pooling on delay: 5,3,2
- GRU units: 64

Neural Network - CONV 3D

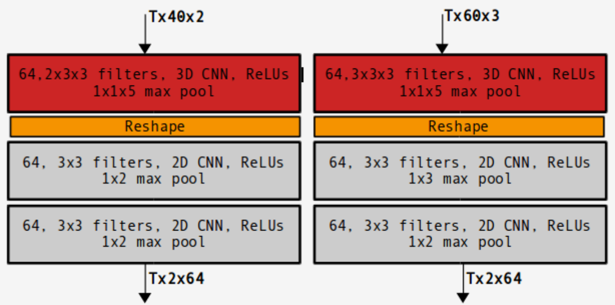


Figure: Convolutional layers architecture.

Neural Network - RNN

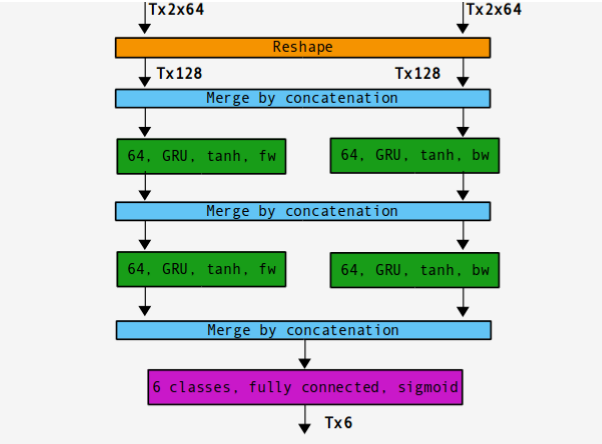


Figure: RNN architecture.

Neural Network - Preprocessing

- Training data Normalization and testing data scaling using its training data weights.
- Re-sampling in subset of 256 frames.
- Reshape to fit the 3D Convolutional input requirement (4D tensor).

This particular input of features along channel-time-frequency enables the network to learn both inter- and intra-channel features simultaneously.

Training

GPU	Training epoch ETA
NVIDIA 940MX 2 GB	$\sim$ 120 s
NVIDIA RTX 2080 8 GB	$\sim$ 32 s
NVIDIA Tesla K80 12 GB	$\sim$ 29 s

Table: Training epoch ETA for the tested hardware with the gcc and mbe branches.

Training - Custom Class Metric

Development of a *Keras* custom class to compute a personalized metric for each epoch in order to evaluate the model and perform Early stopping.
All computations are done using callbacks:

- *on_train_begin* to initialize some variables.
- *on_epoch_end* to compute our frame-wise metric and storage all the data for future analysis.

```
class Metrics(keras.callbacks.Callback):
    def on_train_begin(self, logs={}):
        self._er= 0
        self._f1 = 0
        #    ...
        #other variables
```

Training - Custom Class Metric

```
def on_epoch_end(self,epoch, batch, logs={}):
    X_val,X_val_gcc, y_val = self.validation_data[0],
        self.validation_data[1],self.validation_data[2]
    pred = model.predict([X_val,X_val_gcc])
    pred_thresh = pred > 0.5
    #print pred_thresh
    score_list = metrics.compute_scores(
        pred_thresh, y_val, frames_in_1_sec=frames_1_sec)
    self._f1 = score_list['f1_overall_1sec']
    self._er = score_list['er_overall_1sec']
    #error rate over epoch
    self.er_mean_batch += self._er
    self.er_mean= float(self.er_mean_batch) / (epoch+1)
    if self.er_mean > self.er_mean_prev:
        self._fail_count+=1
        if self._fail_count >= 100:
            self.model.stop_training = True
    else:
        self._fail_count = 0
    self._er_prev = self._er
    self.er_mean_prev = self.er_mean
    self._er_list.append(self._er)
    self._f1_list.append(self._f1)
    self.er_mean_list.append(self.er_mean)
    return
```

Training - Parameters

- Adam optimizer with 0.0001 learning rate
- Suggested dropout rate: 0.2
- Actually used dropout rate: 0.5
- Early stopping patience 100 epochs
- only mbe features batch size: 128
- gcc and mbe features batch size: 64

Metric

The proposed SED method is evaluated using the poly-phonic SED metrics proposed in [1]. Particularly we use:

- segment wise error rate (ER).
- F-score calculated in one-second length segments (43 frames).

Metric - F-score

$$F = \frac{2 \cdot \sum_{k=1}^{K} TP(k)}{2 \cdot \sum_{k=1}^{K} TP(k) + \sum_{k=1}^{K} FP(k) + \sum_{k=1}^{K} FN(k)} \tag{2}$$

where for each one-second segment k:

- TP(k) is the number of true positives.
- FP(k) is the number of false positives.
- FN(k) is the number of false negatives.

Metric - ER

The error rate is measured as:

$$ER = \frac{\sum_{k=1}^{K} S(k) + \sum_{k=1}^{K} D(k) + \sum_{k=1}^{K} I(k)}{\sum_{k=1}^{K} N(k)} \tag{3}$$

where:

- N(k) is the total number of active sound events in the ground truth of segment k.
- S(k) is the number of substitutions.
- D(k) is the number of deletions.
- I(k) is the number of insertions.

Metric - ER

are measured using the following equations for each of the K one second segments:

$$
\begin{aligned}
S(k) &= \min(FN(k), FP(k)) \\
D(k) &= \max(0, FN(k) - FP(k)) \\
I(k) &= \max(0, FP(k) - FN(k))
\end{aligned}
\tag{4}
$$

Results

C3RNN	ER	F1
dropout = **0.2**		
mbe-gcc	70.6	51.9
mbe	70.2	51.2
dropout = **0.5**		
mbe-gcc	72.3	53.2
mbe	74.7	52.4

Table: 4-folds mean metric scores for SED using the C3RNN.

CRNN	ER	F1
dropout = **0.2**		
mbe	70.5	50.3
dropout = **0.5**		
mbe	71.4	51.9

Table: 4-folds mean metric scores for SED using the CRNN mbe only.

Results - Plots

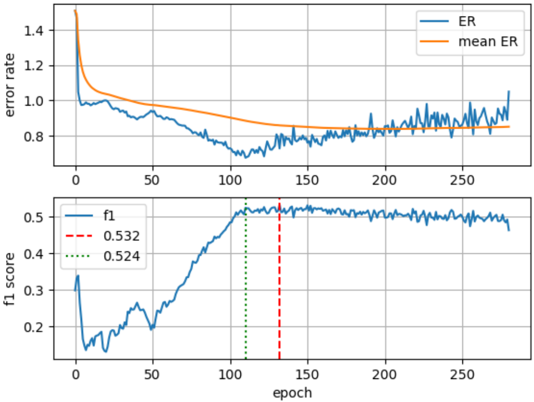


Figure: C3RNN mbe features only with dropout $= 0.2$, best f1 (- - - -), f1 for the best ER (⋯⋯).

Results - Plots

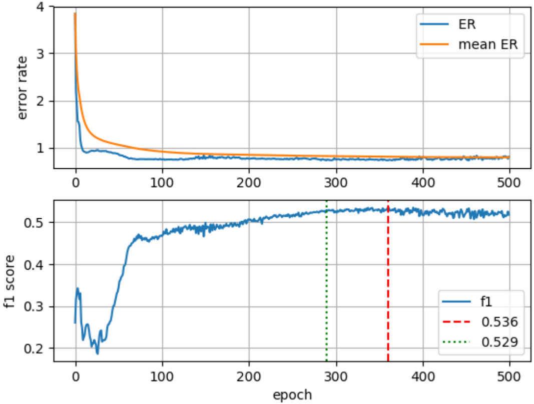


Figure: C3RNN mbe features only with dropout = 0.5, best f1 (- - - -), f1 for the best ER (⋯⋯).

Results - Plots

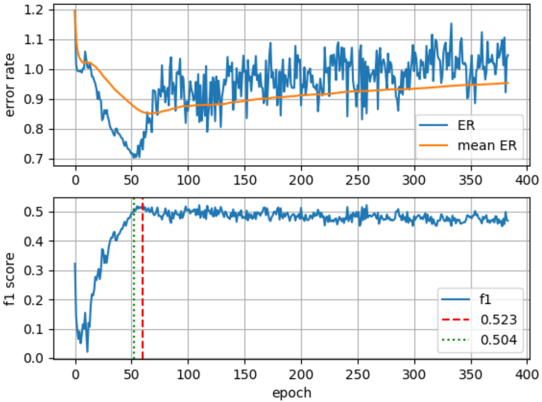


Figure: C3RNN gcc and mbe features with dropout = 0.2, best f1 (- - - -), f1 for the best ER (······).

Results - Plots

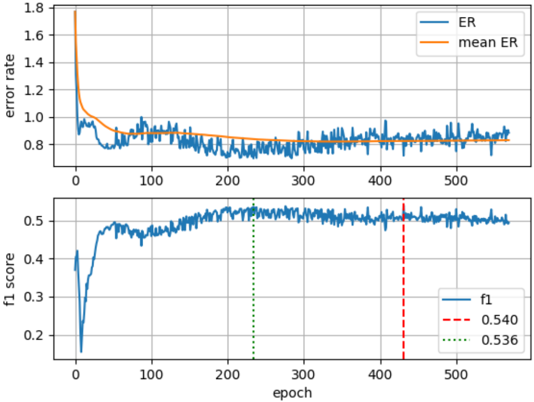


Figure: C3RNN gcc and mbe features with dropout = 0.5, best f1 (- - - -), f1 for the best ER (......).

Thank You

References

[1] Sharath Adavanne, Archontis Politis, and Tuomas Virtanen. "Multichannel Sound Event Detection Using 3D Convolutional Neural Networks for Learning Inter-channel Features". In: *CoRR* abs/1801.09522 (2018). arXiv: 1801.09522. URL: http://arxiv.org/abs/1801.09522.

[2] Sharath Adavanne and Tuomas Virtanen. "A report on sound event detection with different binaural features". In: *CoRR* abs/1710.02997 (2017). arXiv: 1710.02997. URL: http://arxiv.org/abs/1710.02997.

[3] Il-Young Jeong et al. "Audio Event Detection Using Multiple-Input Convolutional Neural Network". In: (Nov. 2017), pp. 51–54.

[4] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. "Metrics for polyphonic sound event detection". English. In: *Applied Sciences* 6.6 (2016). ISSN: 2076-3417. DOI: 10.3390/app6060162.

[5] Annamaria Mesaros et al. "DCASE2017 Challenge Setup: Tasks, Datasets and Baseline System". In: *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*. Nov. 2017, pp. 85–92.