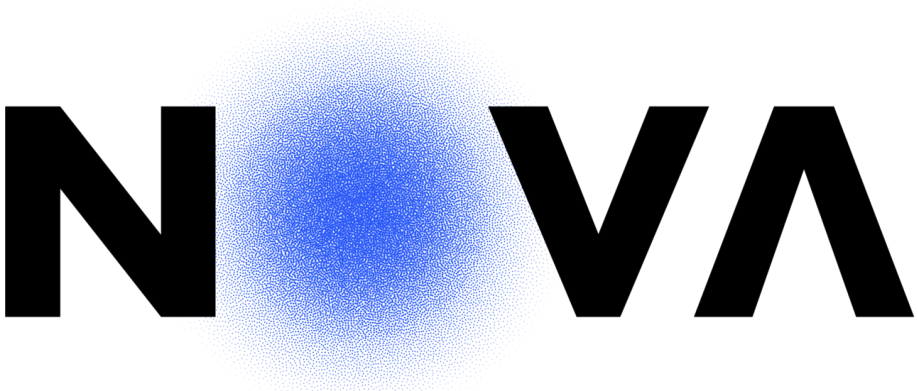

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Learning from Unstructured Data Report

Riccardo Galarducci 66819

AY. 2022-2023

1 Introduction

The aim of the project is to solve a multi class classification problem on audio data using three types of models:

1. Convolutional Network
2. Recurrent Network
3. Transformer Network

Each model is presented in a dedicated section together with its performances on test data as well as the tuning process of architecture and hyperparameters.

2 Data Preparation

The data set is consituted by *wav* files containing single channel sound signal sampled at 8000. In the preprocessing phase the *wav* files are resampled to 16 kHz single-channel audio.
The training data set counts 64000 records and is composed by *trainval* $1, 2, 3, 4$, while validation set (*trainval* 5) counts 1572 records and test set counts 436 records.
There are 10 labels that correspond to the source of the generated sound. The data set is unbalanced wrt some labels, in particular we have fewer examples of labels 1 which corresponds to *car horn* and label 6 *gun shot* as we can see in figure 1.

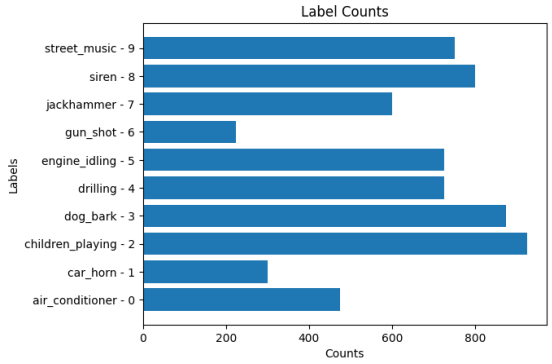

Figure 1: Labels count

2.1 Short Time Fourier Transform.

Each networks is trained both and the waveforms and on the STFT. The STFT divides the signals into overlapping frames of a specified length, applies a window function to each frame, and computes the discrete Fourier transform (DFT) of each windowed frame. We set the *frame_length* parameter, which is the size of the window, to 128 samples and the *frame_steps* parameter, which determines the overlap between consecutive frames, to 64 samples. The shape of the transformed waveform is time steps, features, and has a considerably smaller size wrt the raw waveforms. Figure 2 shows the raw waveform of a children playing while figure 3 shows its spectrogram obtained from the short time fourier transformation.

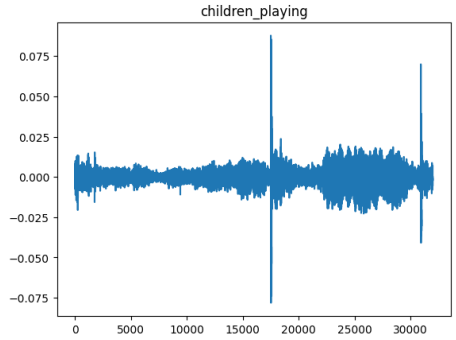

Figure 2: Waveform - Children Playing

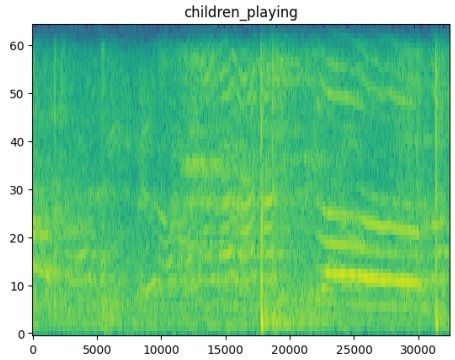

Figure 3: Spectrogram - Children Playing

2.2 Training Settings

The waveforms we are dealing have different length, for this reason we exploit PaddedBatchDataset to pass to the models batches of 256 samples that are padded in order to have same length within the batch.
All the models are trained using **Adam optimizer**, with learning rate equal to 0.001. We performed some trials during the training of some models using a learning rate scheduler by monitoring the validation loss and reducing the learning rate by a factor of 2-10 once learning stagnates for 5 to 10 epochs. The use of the scheduler didn't yields good results and slow down the training too early, leading the model to underfit the data. For this reason we abandoned the idea and proceed with a fixed learning rate across the trainings.
The loss function used is the **Sparse Categorical Crossentropy** which is suitable when there are two or more label classes as in the current scenario.

3 Convolutional Network

3.1 CNN on waveforms

Architecture & Hyperparameters. The CNN on waveforms is composed by a features extractor block constituted by repetitions of convolution layers and pooling layers, and a head with one dense layer and an output layer with 10 units as the number of labels. Between the feature extractor block and the head a GlobalMaxPooling1D works as a bridge, the data cannot be flattened because we have samples with different lengths in the data set. We add a preprocessing layer at the beginning of the model in particular a Normalization layer. Because we are dealing with sequences we used Conv1D layers and MaxPooling1D in the feature extractor blocks. We used four stacks of conv+pool layers to be able to capture increasingly complex pattern in the sequences. Moreover, as the depth of the model increased, we increased the kernel size, in particular the models that yields the best results has Conv1D layers with kernels of size 5, 10, 15 and 20 respectively. Kernel with longer size allow to capture patterns in the sound that occur over longer time periods. Each Conv1D layer have *elu* activation function which yields higher performances on both training and validation set wrt the more popular *relu*, while the weights are initialized with *HeNormal* strategy.The number of filters in each convolutional layer is 64, a higher number of layers (128) decrease the performances on both training and validation set.
For what concern the pooling layers, we tried poll sizes of 2 and 4, with the latter that leads to higher performances. The head is composed by a dense layer, with 128 hidden units and the output layer.

Regularization. As regularization techniques we exploit SpatialDropout1D layer with rate equal to 0.5, after each pooling layer. Additionaly, we plugged a Dropout layer with rate equal to 0.3 after the GlobalMaxPooling1D and the first Dense layer in the head of the model.

Training. The model has been trained for 50 epochs. Validation accuracy remains stable since the tenth epoch as we can see in figure 4.

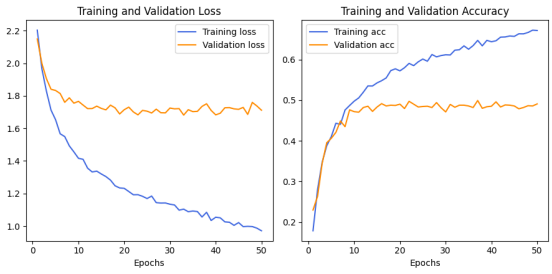

Figure 4: Training& Validation Losses and Accuracy

Performances. The best performances of the model on training and validation set are respectively 0.6808 and 0.4927 accuracy, while once tested on the test set it reaches 0.6159. Figure 5 shows the confusion matrix on the test set.

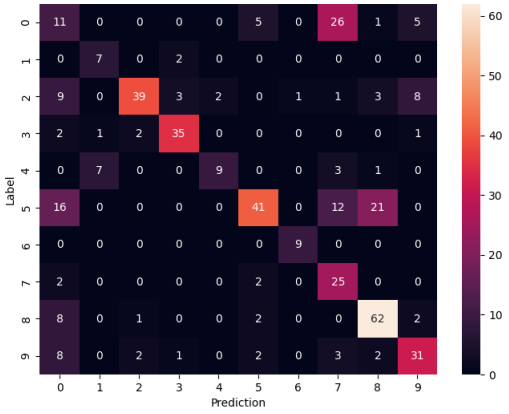

Figure 5: Confusion Matrix Test Set

In listing 1 is shown the model architecture. The model has 194, 506 trainable parameters.

Listing 1: CNN on waveforms

```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, None)]            0

reshape (Reshape)            (None, None, 1)           0

normalization (Normalizatio  (None, None, 1)           3
n)

conv1d (Conv1D)              (None, None, 64)          384

max_pooling1d (MaxPooling1D  (None, None, 64)          0
)

spatial_dropout1d (Spatial1D (None, None, 64)          0
ropout1D)

conv1d_1 (Conv1D)            (None, None, 64)          41024

max_pooling1d_1 (MaxPooling  (None, None, 64)          0
1D)

spatial_dropout1d_1 (Spatia  (None, None, 64)          0
lDropout1D)

conv1d_2 (Conv1D)            (None, None, 64)          61504

max_pooling1d_2 (MaxPooling  (None, None, 64)          0
1D)

spatial_dropout1d_2 (Spatia  (None, None, 64)          0
lDropout1D)

conv1d_3 (Conv1D)            (None, None, 64)          81984

global_max_pooling1d (Globa  (None, 64)                0
lMaxPooling1D)

dropout (Dropout)            (None, 64)                0

dense (Dense)                (None, 128)               8320

dropout_1 (Dropout)          (None, 128)               0

dense_1 (Dense)              (None, 10)                1290

=================================================================
Total params: 194,509
Trainable params: 194,506
Non-trainable params: 3
_________________________________________________________________
```

3.2 CNN on STFT

Architecture & Hyperparameters. The CNN on STFT have basically the same general structure of the

CNN on waveform, with a feature extractor block , composed by three stacks of Conv1D + MaxPooling1D layers, and ANN in the head.
Because of the reduced size of the STFTs we tried smaller kernel size in the Conv1D layers, the best models in terms of performances has kernels with size 3 in all the Conv1D in the three stacks. Instead wrt the CNN on waveforms we have increased the number of filters to 128. Also in this model we have used elu activation function and HeNormal kernel initializer both in the Conv1D layers and in the Dense layers.
The pool size in the MaxPooling1D layers has been set to 2.

Regularization. As regularization techniques we exploit SpatialDropout1D layers with rate equal to 0.4, after each pooling layers. In addition, we plugged a Dropout layer with rate equal to 0.3 after the GlobalMaxPooling1D and the first Dense layer in the head of the model.

Training. The model has been trained for 100 epochs, to be sure to not have an increasing validation accuracy as we can see in figure 8.

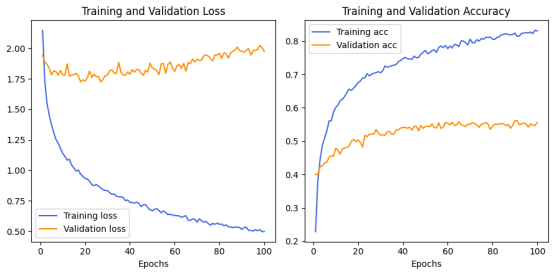

Figure 6: Training& Validation Losses and Accuracy

Performances. The best performances in terms of accuracy of the model on training and validation set are respectively 0.8324 and 0.5621 accuracy, while once tested on the test set it hits 0.7408. Figure 5 shows the confusion matrix on the test set.

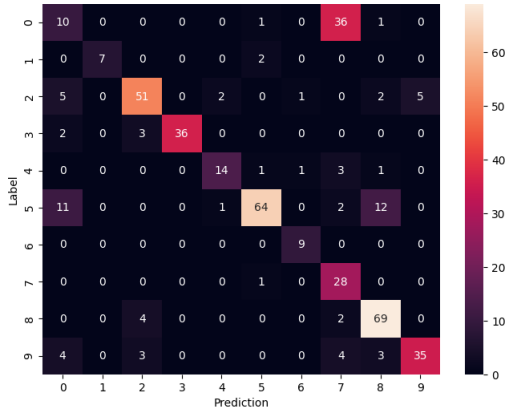

Figure 7: Confusion Matrix Test Set

In listing 1 is shown the model architecture. The model has 194, 506 trainable parameters.

Listing 2: CNN on STFT

```
 Layer (type)                 Output Shape              Param #
=================================================================
 input_5 (InputLayer)         [(None, None, 65)]        0

 normalization_4 (Normalizat  (None, None, 65)         131
 ion)

 conv1d_4 (Conv1D)            (None, None, 128)         25088

 max_pooling1d_4 (MaxPooling  (None, None, 128)         0
 1D)

 spatial_dropout1d_4 (Spatia  (None, None, 128)         0
 lDropout1D)

 conv1d_5 (Conv1D)            (None, None, 128)         49280


 max_pooling1d_5 (MaxPooling  (None, None, 128)         0
 1D)

 spatial_dropout1d_5 (Spatia  (None, None, 128)         0
 lDropout1D)

 conv1d_6 (Conv1D)            (None, None, 128)         49280


 global_max_pooling1d_4 (Glo  (None, 128)               0
 balMaxPooling1D)

 dropout_8 (Dropout)          (None, 128)               0

 dense_7 (Dense)              (None, 128)               16512

 dropout_9 (Dropout)          (None, 128)               0

 dense_8 (Dense)              (None, 10)                1290

=================================================================
Total params: 141,581
Trainable params: 141,450
Non-trainable params: 131
_________________________________________________________________
```

4 Recurrent Network

4.1 RNN on waveforms

Architecture & Hyperparameters. The recurrent network on the waveforms is a CRNN. The model has a features exctractor block composed by stacks of Conv1D + MaxPooling layers. We exploit the same structure used in CNN on waveforms as it seems to be a good feature extractor. The stacks of conv + pool has also the aim of reduce the length of the sequences as the recurrent units needs to have as inputs sequences of small sizes to get good performances. The feature extractor block is connected to a stack of recurrent layers. We tried the model with two and three layers of 64 and 128 LSTM and GRU units. The highest results in terms of performances is reached by the two layers with 128 LSTM units. To pass the outputs to the head of the model composed we tried both a LSTM layers which do not return the complete processed sequence but just the last output in the output sequence and a GlobalMaxPooling1D after the last LSTM layers which return the complete sequence. The latter reached slightly better results in terms of accuracy on validation set but in less training epochs. This have lead us to prefer the latter in the final configuration of the model. The head is composed by a Dense layers with 128 units and elu activation functiona followed by the classical output layer with ten units.

Regularization. The fraction of the units to drop for the linear transformation of the inputs in the recurrent layer has been set to 0.2. Its the rate that yield the best generalization results.

Training. The model has been trained for 100 epochs. Figure XX shown the validation accuracy.

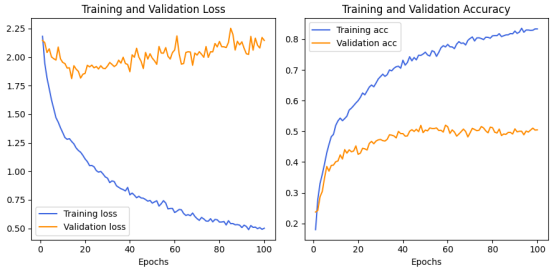

Figure 8: Training& Validation Losses and Accuracy

Performances The best performances of the model on training and validation set are respectively 0.8356 and 0.5156 accuracy, while once tested on the test set it hits 0.6078. Figure 9 shows the confusion matrix on the test set.

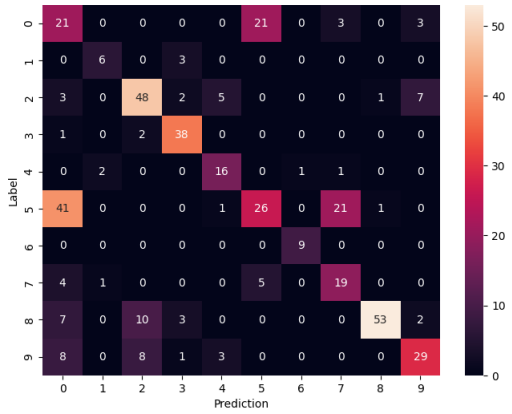

Figure 9: Confusion Matrix Test Set

In listing 4 is shown the model architecture.

Listing 3: RNN on waveforms

```
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, None)]            0

reshape_1 (Reshape)          (None, None, 1)           0

normalization_1 (Normalizat  (None, None, 1)           3
ion)

conv1d_4 (Conv1D)            (None, None, 64)          384

max_pooling1d_4 (MaxPooling  (None, None, 64)          0
1D)

spatial_dropout1d_4 (Spatia  (None, None, 64)          0
lDropout1D)

conv1d_5 (Conv1D)            (None, None, 64)          41024

max_pooling1d_5 (MaxPooling  (None, None, 64)          0
1D)

spatial_dropout1d_5 (Spatia  (None, None, 64)          0
lDropout1D)

conv1d_6 (Conv1D)            (None, None, 64)          61504

max_pooling1d_6 (MaxPooling  (None, None, 64)          0
1D)

spatial_dropout1d_6 (Spatia  (None, None, 64)          0
lDropout1D)

conv1d_7 (Conv1D)            (None, None, 64)          81984

max_pooling1d_7 (MaxPooling  (None, None, 64)          0
1D)

spatial_dropout1d_7 (Spatia  (None, None, 64)          0
lDropout1D)

lstm_3 (LSTM)                (None, None, 128)         98816

lstm_4 (LSTM)                (None, None, 128)         131584

global_max_pooling1d (Globa  (None, 128)               0
lMaxPooling1D)

dense_2 (Dense)              (None, 128)               16512

dropout_1 (Dropout)          (None, 128)               0

dense_3 (Dense)              (None, 10)                1290

=================================================================
Total params: 433,101
Trainable params: 433,098
Non-trainable params: 3
_________________________________________________________________
```

4.2 RNN on STFT

Architecture & Hyperparameters. The recurrent network on the STFT has the same general strcuture of

the RNN on waveforms, the model is constituted by a feature extractor block constituted by two stacks of conv + pool layers and a stacks of recurrent units, with a head composed by a dense layer and an output layer. We have exploited the feature extractor of the CNN with same changes on the kernel size which has been increased to 5 for the first Conv1D layer and to 10 for the second, increasing the kernel size have increased the performances of the model on both training and validation set. The RNN block is composed by two LSTM layers constituted by 128 units followed by a GlobalMaxPooling1D layer which is a bridge from the recurrent block to the head of the model. The head of the model is constituted by a Dense layer with 128 units and elu activation function followed by the output layer.

Regularization. As regularization techniques we exploit SpatialDropout1D layers with rate equal to 0.3, after each pooling layers. Moreover, we used Dropout layer with rate equal to 0.3 after the GlobalMaxPooling1D and the first Dense layer in the head of the model.

Training. The model has been trained for 100 epochs. The validation accuracy seems to pretty stable since 50_{th} epoch as we can see in figure 10.

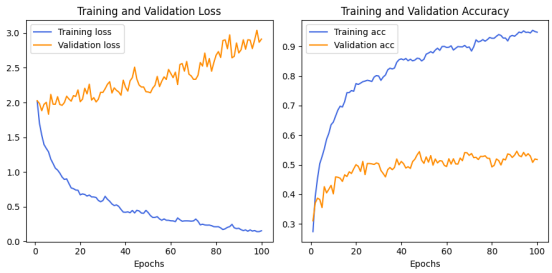

Figure 10: Training& Validation Losses and Accuracy

Performances The best performances of the model on training and validation set are respectively 0.9501 and 0.5455 accuracy, while once tested on the test set it reaches 0.7294 test accuracy. Figure 11 shows the confusion matrix on the test set.

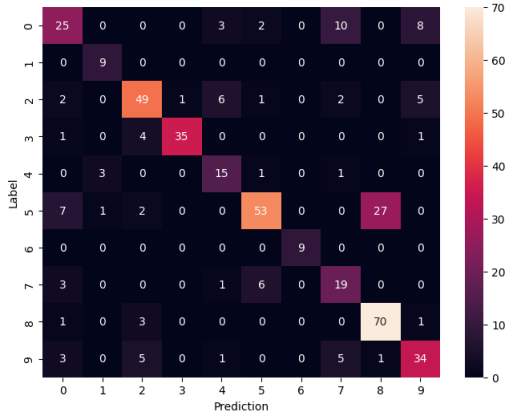

Figure 11: Confusion Matrix Test Set

In listing **??** is shown the model architecture.

Listing 4: RNN on STFT

```
Layer (type)                 Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, None, 65)]        0

 normalization_1 (Normalizat  (None, None, 65)        131
 ion)

 conv1d_2 (Conv1D)           (None, None, 128)         41728

 max_pooling1d_2 (MaxPooling  (None, None, 128)        0
 1D)

 spatial_dropout1d_2 (Spatia  (None, None, 128)        0
 lDropout1D)

 conv1d_3 (Conv1D)           (None, None, 128)         163968

 max_pooling1d_3 (MaxPooling  (None, None, 128)        0
 1D)

 spatial_dropout1d_3 (Spatia  (None, None, 128)        0
 lDropout1D)

 lstm_2 (LSTM)               (None, None, 128)         131584

 lstm_3 (LSTM)               (None, None, 128)         131584

 global_max_pooling1d_1 (Glo  (None, 128)              0
 balMaxPooling1D)

 dropout_2 (Dropout)         (None, 128)               0

 dense_2 (Dense)             (None, 128)               16512

 dropout_3 (Dropout)         (None, 128)               0

 dense_3 (Dense)             (None, 10)                1290

=================================================================
Total params: 486,797
Trainable params: 486,666
Non-trainable params: 131
```

5 Transformer Network

5.1 Transformer on waveforms

Architecture & Hyperparameters. The Transformer model on waveforms is constituted by the same feature extractor block used in CNN on waveform, followed by a position embedding layer applied to the output of the last pooling layer. The output of the last pooling layer and of the position embedding layer is added in an element-wise fashion to create an intermediate representation. This combination allows the model to incorporate both local features extracted by convolution and positional information provided by position embedding. The intermediate representation is passed to a stack of two TransformerDecoder layers with intermediate dimension equal to 128 (hidden size of feedforward network) and 8 MultiHeadAttention. The output is passed to a GlobalMaxPooling1D which is the bridge for the output layer with 10 units.

Regularization. We add SpatialDropout layer with rate equal to 0.3 after the first three pooling layers in the feature extractor block, and a Dropout layer with rate equal to 0.2 after the GlobalMaxPooling layer.

Training. The model has been trained for 100 epochs with a stable validation accuracy after epoch 30 as we can see in figure XX.

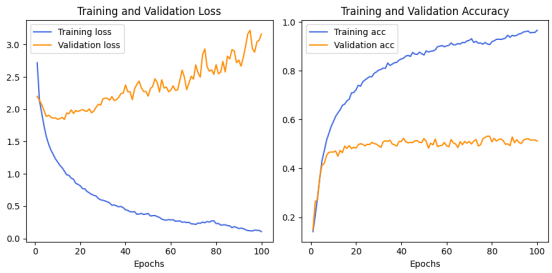

Figure 12: Training& Validation Losses and Accuracy

Performances The best performances of the model on training and validation set are respectively 0.9664 and 0.5321 accuracy, while once tested on the test set it reaches 0.6803 test accuracy. Figure 13 shows the confusion matrix on the test set.

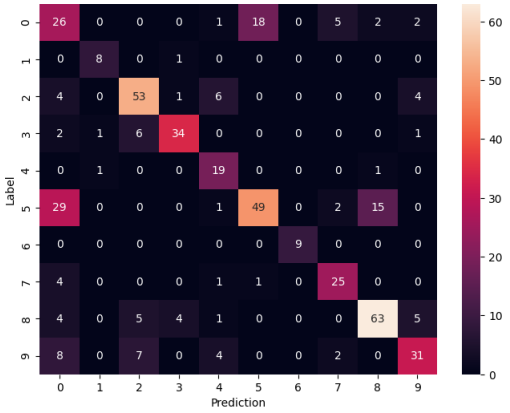

Figure 13: Confusion Matrix Test Set

In listing 6 is shown the model architecture.

Listing 5: Transformer on waveform

```
 Layer (type)                 Output Shape              Param #
=================================================================
 (InputLayer)                [(None, None)]            0

 (Reshape)                   (None, None, 1)           0

 (Normalization)             (None, None, 1)           3

 (Conv1D)                    (None, None, 64)          384

 (MaxPooling1D)              (None, None, 64)          0

 (SpatialDropout1D)          (None, None, 64)          0

 (Conv1D)                    (None, None, 64)          41024

 (MaxPooling1D)              (None, None, 64)          0

 (SpatialDropout1D)          (None, None, 64)          0

 (Conv1D)                    (None, None, 64)          61504

 (MaxPooling1D)              (None, None, 64)          0

 (SpatialDropout1D)          (None, None, 64)          0

 (Conv1D)                    (None, None, 64)          81984

 (MaxPooling1D)              (None, None, 64)          0

 (PositionEmbedding)         (None, None, 64)          8064

 (TFOpLambda)                (None, None, 64)          0

 (TransformerDecoder)        (None, None, 64)          33472

 (TransformerDecoder)        (None, None, 64)          33472

 (GlobalMaxPooling1D)        (None, 64)                0

 dropout (Dropout)           (None, 64)                0

 dense (Dense)               (None, 10)                650

=================================================================
Total params: 260,557
Trainable params: 260,554
Non-trainable params: 3
_________________________________________________________________
```

5.2 Transformer on STFT

Architecture & Hyperparameters. The model architecture of the transformer model on stft is similar to the Transformer on the raw data, the only difference is the convolutional feature extractor block which is adapted to the STFT. We have exploited the feature extractor tuned in CRNN on STFT.

Regularization. We add only a dropout layer after the GLobalMaxPooling layer which connect the TransormerDecoder stacks to the output layer. Adding SpatialDropout to the feature extractor block or to the TransformerDecoder layer decrease the performances of the model on validation set.

Training. The model has been trained at a first instance to 60 epochs. We observed that after reaching a peak of validation accuracy above 0.55 at epoch 30, the performances decrease and stabilize at 0.51% of validation accuracy. For this reason we retrained the model for 30 epochs and it reaches 0.5347 on validation accuracy.

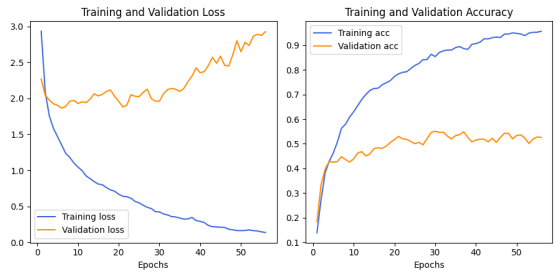

Figure 14: Training& Validation Losses and Accuracy

Performances The best performances of the model on training and validation set are respectively 0.8416 and 0.5347 accuracy, while once tested on the test set it reaches 0.7189 test accuracy. Figure 15 shows the confusion matrix on the test set.

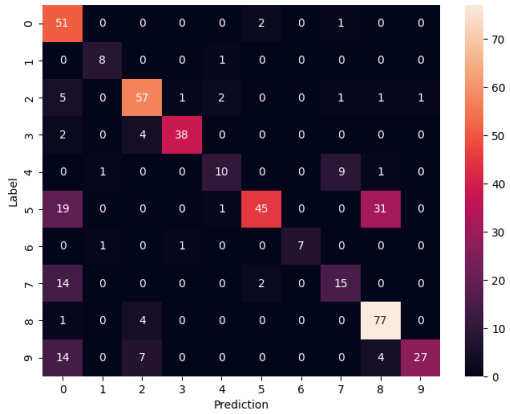

Figure 15: Confusion Matrix Test Set

In listing 15 is shown the model architecture.

Listing 6: Transformer on waveform

```
 Layer (type)                 Output Shape              Param #
=================================================================
 (InputLayer)                [(None, None, 65)]        0

 (Normalization)              (None, None, 65)          131

 (Conv1D)                     (None, None, 64)          20864

 (MaxPooling1D)               (None, None, 64)          0

 (Conv1D)                     (None, None, 64)          41024

 (MaxPooling1D)               (None, None, 64)          0


 (PositionEmbedding)         (None, None, 64)          8064

 (TFOpLambda)                (None, None, 64)           0

 (TransformerDecoder)        (None, None, 64)          25216

 (TransformerDecoder)       (None, None, 64)          25216

 (GlobalMaxPooling1D)              (None, 64)           0

 dropout (Dropout)                 (None, 64)           0

 dense (Dense)                     (None, 10)           650

=================================================================
Total params: 121,165
Trainable params: 121,034
Non-trainable params: 3
_________________________________________________________________
```