Machine Learning – Convolutional Neural Network – Report

Shlomi Ben-Shushan

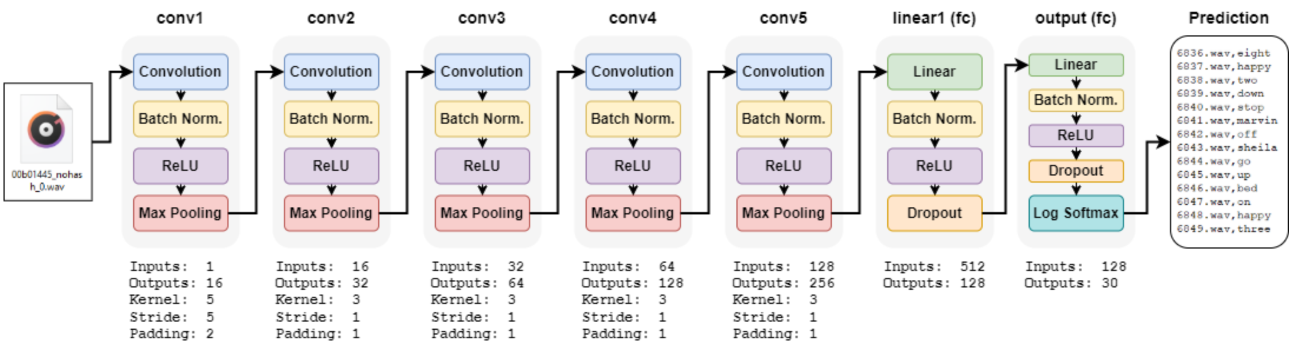
Summary

In this report I will describe the implementation details of a multi-class neural network for classifying audio samples. I will specify the model's description, architecture, hyper-parameters, losses and accuracies per epoch, and I will explain how to use it.

Description

The name of the model is SoundClassifierNN, and it is a multi-class neural network for audio samples classification, implemented in Python using PyTorch. It consists of five convolution layers in sizes 16, 32, 64, 128 and 256, and one linear (fully-connected) layer in sizes 128. After each convolution layer, the functions Batch Normalization, ReLU and Max Pooling are activated, and after the linear layer, the functions Batch Normalization, ReLU and Dropout are activated. This model is trained and optimized by Adam optimizer. The output of the NN comes out through Log-Softmax function.

Architecture



```
6836.wav,eight
6837.wav,happy
6838.wav,two
6839.wav,down
6840.wav,stop
6841.wav,marvin
6842.wav,off
6843.wav,sheila
6844.wav,go
6845.wav,up
6846.wav,bed
6847.wav,on
6848.wav,happy
6849.wav,three
```

```
conv1         conv2         conv3         conv4         conv5          linear1 (fc)   output (fc)
Inputs:  1    Inputs:  16   Inputs:  32   Inputs:  64   Inputs:  128   Inputs:  512   Inputs:  128
Outputs: 16   Outputs: 32   Outputs: 64   Outputs: 128  Outputs: 256   Outputs: 128   Outputs: 30
Kernel:  5    Kernel:  3    Kernel:  3    Kernel:  3    Kernel:  3
Stride:  5    Stride:  1    Stride:  1    Stride:  1    Stride:  1
Padding: 2    Padding: 1    Padding: 1    Padding: 1    Padding: 1
```

Hyper Parameters

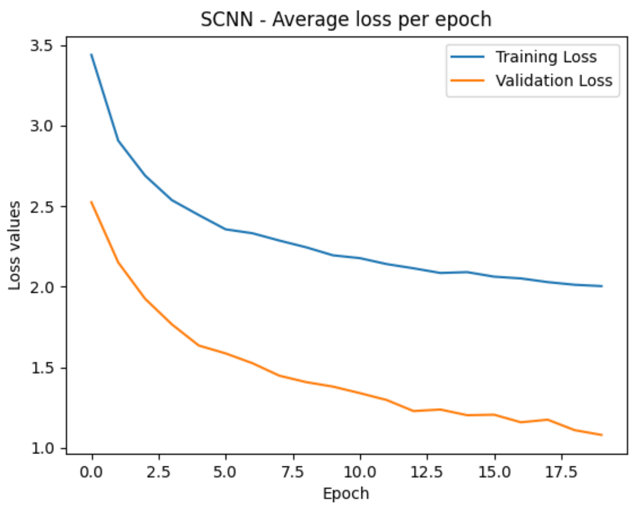
The model uses the following two hyper-parameters:

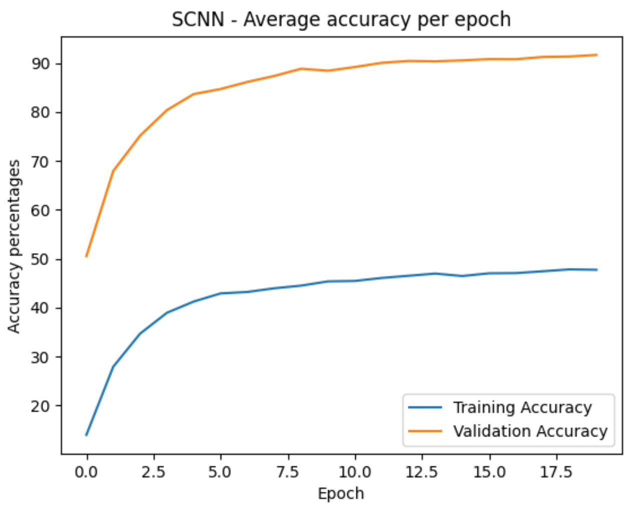
1. Learning Rate: $\eta = 0.0001$. Used by Adam optimizer to update model's parameters.
2. Number of Epochs: $n_{epochs} = 20$. Defines the number of training-validation iterations.

The parameters were found through trial-and-error w.r.t the validation accuracy (the higher the better).

Loss and Accuracy

The following charts shows the model's improvements in terms of losses and accuracies over the epochs.





Running Instructions

- The program file *ex5.py* must resides in the same directory as *gcommands_loader.py* program file.
- The model assumes that the learning dataset is already divided to training and validation.
- The training dataset should be in form of a file-system directory that contains different sub-directories named after the labels in the experiment, and each sub-directory contains samples as *wav* files, correspondent to the name of the directory (which is the label).
- The validation dataset should be in the same form as the form of the training dataset.
- The test dataset should be in the form of a file-system directory that contains another directory (whose name doesn't matter), that contains test samples as *wav* files in names that consist of integers only.
- To run the model, use the command *python ex5.py X Y Z W p*, where:
  - *X* – represents the path of the training-dataset directory.
  - *Y* – represents the path of the validation-dataset directory.
  - *Z* – represents the path of the test-dataset directory.
  - *W* – represents the desired path and name of the output log file (usually *test_y*).
  - *p* – represents an optional input flag that if it is "*-p*" or "*--plot*", then the program will create losses and accuracies per epoch plots after the learning-phase and save them to PNG files at the location of the program file *ex5.py*.

  For example, to run the program without creating plot files, use the command:

  ```
  python ex5.py data/train data/valid data/test test_y
  ```

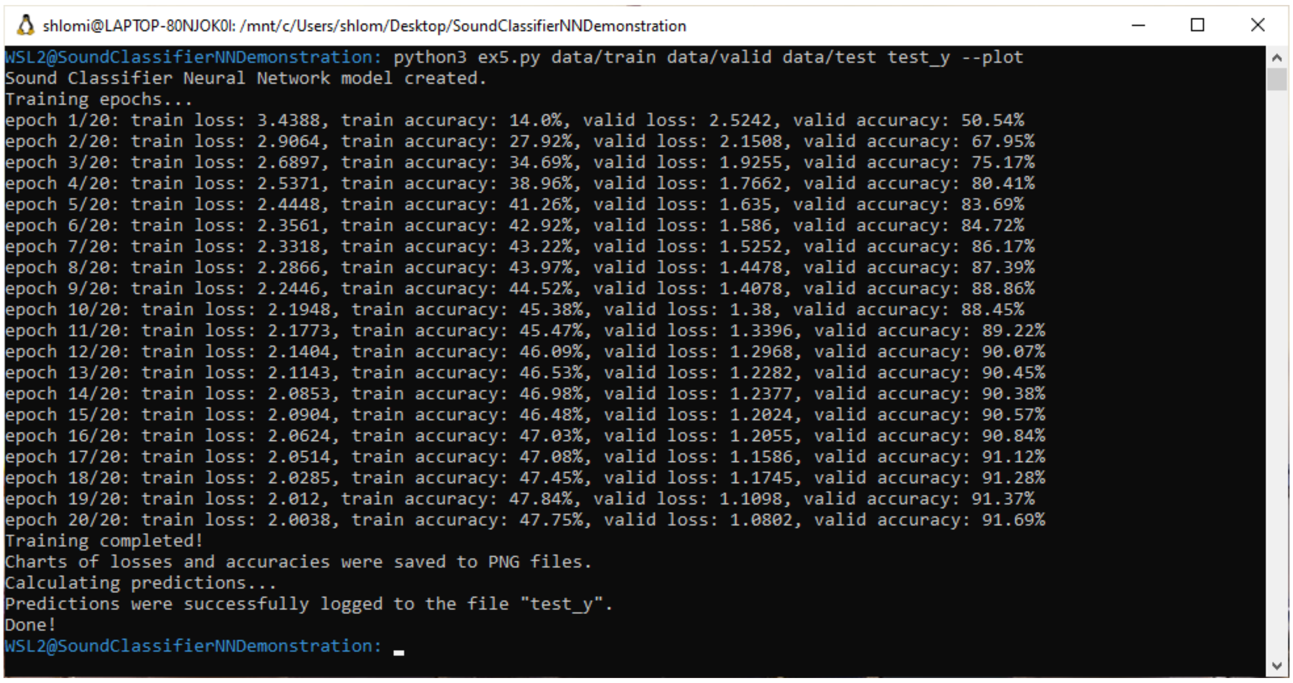
  Alternatively, to make the program create plot files, use one of the commands:

  ```
  python ex5.py data/train data/valid data/test test_y -p
  python ex5.py data/train data/valid data/test test_y --plot
  ```

- Note that the running time of the program may be several hours. On my machine, which uses Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz-1.99 GHz and 16GB of RAM (without GPU), the running time was 3 hours and 39 minutes.

Running Example

The program prints progress information to the console as shown in the following screenshots.



shlomi@LAPTOP-80NJOK0I: /mnt/c/Users/shlom/Desktop/SoundClassifierNNDemonstration

```
WSL2@SoundClassifierNNDemonstration: python3 ex5.py data/train data/valid data/test test_y --plot
Sound Classifier Neural Network model created.
Training epochs...
epoch 1/20: train loss: 3.4388, train accuracy: 14.0%, valid loss: 2.5242, valid accuracy: 50.54%
epoch 2/20: train loss: 2.9064, train accuracy: 27.92%, valid loss: 2.1508, valid accuracy: 67.95%
epoch 3/20: train loss: 2.6897, train accuracy: 34.69%, valid loss: 1.9255, valid accuracy: 75.17%
epoch 4/20: train loss: 2.5371, train accuracy: 38.96%, valid loss: 1.7662, valid accuracy: 80.41%
epoch 5/20: train loss: 2.4448, train accuracy: 41.26%, valid loss: 1.635, valid accuracy: 83.69%
epoch 6/20: train loss: 2.3561, train accuracy: 42.92%, valid loss: 1.586, valid accuracy: 84.72%
epoch 7/20: train loss: 2.3318, train accuracy: 43.22%, valid loss: 1.5252, valid accuracy: 86.17%
epoch 8/20: train loss: 2.2866, train accuracy: 43.97%, valid loss: 1.4478, valid accuracy: 87.39%
epoch 9/20: train loss: 2.2446, train accuracy: 44.52%, valid loss: 1.4078, valid accuracy: 88.86%
epoch 10/20: train loss: 2.1948, train accuracy: 45.38%, valid loss: 1.38, valid accuracy: 88.45%
epoch 11/20: train loss: 2.1773, train accuracy: 45.47%, valid loss: 1.3396, valid accuracy: 89.22%
epoch 12/20: train loss: 2.1404, train accuracy: 46.09%, valid loss: 1.2968, valid accuracy: 90.07%
epoch 13/20: train loss: 2.1143, train accuracy: 46.53%, valid loss: 1.2282, valid accuracy: 90.45%
epoch 14/20: train loss: 2.0853, train accuracy: 46.98%, valid loss: 1.2377, valid accuracy: 90.38%
epoch 15/20: train loss: 2.0904, train accuracy: 46.48%, valid loss: 1.2024, valid accuracy: 90.57%
epoch 16/20: train loss: 2.0624, train accuracy: 47.03%, valid loss: 1.2055, valid accuracy: 90.84%
epoch 17/20: train loss: 2.0514, train accuracy: 47.08%, valid loss: 1.1586, valid accuracy: 91.12%
epoch 18/20: train loss: 2.0285, train accuracy: 47.45%, valid loss: 1.1745, valid accuracy: 91.28%
epoch 19/20: train loss: 2.012, train accuracy: 47.84%, valid loss: 1.1098, valid accuracy: 91.37%
epoch 20/20: train loss: 2.0038, train accuracy: 47.75%, valid loss: 1.0802, valid accuracy: 91.69%
Training completed!
Charts of losses and accuracies were saved to PNG files.
Calculating predictions...
Predictions were successfully logged to the file "test_y".
Done!
WSL2@SoundClassifierNNDemonstration: 
```