COP290 Assignment 1: Task 3 Report

Shubham Sarda (2018TT10958) Ankit Kumar (2020CS10323)

In this subtask, we aim to stitch together an audio processing library for identifying keywords in an audio sample using functions created in previous subtasks. We've created the following source files:

1) libaudio.cpp 2) main.cpp

We've created a Makefile which does the following on using **$make** command:

Note: We've assumed that environment variable **$MKL_BLAS_PATH** contains the path to directory containing lib and include folders of OpenBLAS. Further **$LD_LIBRARY_PATH** should contain the path to libaudio.so (library) file.

1) Creates a library libaudio.so from libaudio.cpp
2) Creates an executable "yourcode.out" using "main.cpp" and links it to libaudio.so

Using the executable, we can run the following command to find the keyword in an audio sample:

```
$./yourcode.out audiosamplefile outputfile
```

1) audiosamplefile: Points to path of file containing features extracted from the audio file in the form of a [1 x 250] vector.
2) outputfile: Points to the path of file (created if doesn't exist), where top 3 possible keywords are outputted along with their softmax probabilities. For instance:
   "audiosamplefile yes unknown left 0.999690 0.000288 2.21e-5"

Note: The keywords detected are {"silence", "unknown", "yes", "no", "up", "down", "left", "right", "on", "off", "stop", "go"}

In the library, we've implemented the following function:

```
pred_t* libaudioAPI(const char* audiofeatures, pred_t* pred){…}
```

This function takes the name of input feature file and pred_t placeholder array. We've created a struct by the name of pred_t whose fields contain the keyword and softmax probability values for a particular prediction. The function returns a pred_t array containing predictions in the descending order of their softmax probabilities i.e. pred[0] gives the keyword with highest softmax probability.

We've implemented a simple neural network with the following architecture:

FC1 [250x144] -> RELU -> FC2 [144x144] -> RELU -> FC3 [144X144] -> RELU -> FC4 [144x12] -> softmax

Pretrained weights provided in "dnn_weights.h" have been used.

Implementation Used

In subtask 2, our best implementations were ones which used MKL and OpenBLAS library functions. Since there was only a slight time efficiency difference between MKL and OpenBLAS implementations, we've tested both implementations in this subtask to select the most efficient. We tested by finding running both the implementations 50 times and taking the average and finally decided to go with OpenBLAS.

Implementation	MKL	OpenBLAS
Average time to identify keywords in a single audio sample in microseconds	2621	2241

Error handling

Error checking has also been implemented as done in previous subtasks. A check is applied to verify if the number of arguments inputted are same as that required to execute the task (argc==3). It also verified that input file given in command line exists or not. In case of any error, program is terminated nicely.