Advanced Topics in Deep Learning:
Speech Signals Exercise 1

Due: 30.11.2022 23:59

Yosi Shrem and Joseph Keshet

November 15, 2022

Guidelines

1. Submit your files using Moodle system.
2. You are allowed to submit in **pairs**. if you choose to do so, **both students have to submit**.
3. In order to submit your solution please submit the following files:
   (a) `ex_1.py` - Python 3.7+ code file with your implementation for part 1.
   (b) `output.txt` - your model's predictions on the given test set (see instructions below).
   (c) `ex_1_report.pdf` - A pdf file of your answers for part 2.

   Follow the instructions and submit all files needed for you code to run.

Good Luck!

1 Part 1

1.1 DTW

In this exercise you will implement your first word recognizer using the Dynamic Time Warping (DTW) algorithm. Your algorithm will recognize the digits 1 - 5.
For that, You will implement the DTW algorithm as described below, where the distance metric (d) should be the Euclidian distance.

$$\begin{aligned} DTW[0,0] &= d(0,0) \\ DTW[i,j] &= d(i,j) + \min \begin{pmatrix} DTW[i-1,j-1] \\ DTW[i,j-1] \\ DTW[i-1,j] \end{pmatrix} \end{aligned} \tag{1}$$

You must implement DTW by yourselves - using a built-in DTW implementation is prohibited!

1.2 Dataset

You are provided with five labeled examples for each class ['one', 'two', 'three', 'four', 'five'] to use as your training set. For the test set you are provided with 253 unlabeled examples. Each file is exactly 1 second long.

1.3 Code

In practice, DTW can be applied to varying length sequences. However, to better understand the advantages of the DTW algorithm, you will compare its performance to a standard Euclidian distance (recall each file is exactly 1 second long).
To sum up, you need to run a 1 nearest neighbor classifier using both Euclidian distance and DTW distance. You need to compute both distance metrics (Euclidian and DTW) for each file in the test set with all training examples. Then, classify each test file as the label of the file with the minimal distance.
You should generate a file named: 'output.txt', with the predictions for each test file using both euclidian distance and DTW distance. The output file should be constructed as follows:

<filename> - <prediction using euclidian distance> - <prediction using DTW distance>

For example,

```
sample1.wav - 4 - 4
sample2.wav - 1 - 1
sample3.wav - 3 - 3
...
Note: DTW and Euclidean distances can yield different predicitons.
```

1.4 Features

In class, we learned how to transform a time domain waveform to the frequency domain using the Fourier Transform. However, in many applications we would like to use more compressed representation. One representation like that is the Mel Frequency Cepstrum Coefficients (MFCCs). In order to extract these features and load the wave files, you will use a python package called 'librosa', using the following lines of code:

```
import librosa
y, sr = librosa.load(f_path, sr=None)
mfcc = librosa.feature.mfcc(y=y, sr=sr)
```

The dimensions of the MFCC object should be (20, 32), meaning 20 MFCC features over 32 time steps. A more detailed explanation of the MFCC can be found here: [paper], and an intuitive explanation can be found here: [blog].

1.5 Sanity-check

You are provided with a python file called `sanity_check.py`. Before submitting, please run it in the same directory as your `output.txt` file to make sure the format is correct.

2 Part 2

Nowadays, neural nets are used for this task.
An example is shown here - (https://github.com/adiyoss/GCommandsPytorch). You can observe the performance and code - There is no need to run it.
Answer the following questions (**up to 1 page**):

1. GCommands is a dataset composed of 30 different words, where each word is pronounced by 1000 speakers. Compare CNN trained on this dataset to the DTW algorithm in terms of training time, inference time, and memory.
2. Given a dataset A consists of 50 samples and dataset B consists of 1,000,000 samples, which approach, CNN or DTW, would you choose for each dataset? explain **shortly**.
3. Assume that the input consists of speech signals of 5 concatenated digits (e.g., zip codes). Suggest an adaptation for the CNN model to support the prediction of multiple digits at once - the entire zip code.

Good Luck!