Speech Signal Processing
— Exercise 6 —
Speech Recognition

Timo Gerkmann, Robert Rehr, Martin Krawczyk-Becker

In the next 3 exercise sessions, you will program a simple speech recognizer, which will be used to distinguish single words like “yes”, “no”, “go” and so on. Log-Mel-Spectrograms will serve as features for characterizing the speech signals and the back-end is given by the dynamic time warp (DTW) algorithm.

1 Audio material

You are provided with audio material that you require throughout the exercises. The vocabulary of the speech recognizer consists of five words which are:

yes, *no*, *go*, *stop*, *next*.

All words are spoken by different speakers. For each speaker, the words were spoken in two different ways: once normally and once slowly. The following name scheme is used for all recorded audio files: `word_speed_speakerinitials.wav`, e.g., `yes_normal_rr.wav`, `no_normal_rr.wav`, `yes_slow_rr.wav`, `no_slow_rr.wav` and so on.

2 Feature extraction

This part of the exercise deals with the feature extraction. In this step, the time domain signal is transformed to another domain which is more suitable for speech recognition. Often, spectro-temporal representations similar to the short-time Fourier transform (STFT) are chosen. In this exercise, we will employ the Log-Mel-Spectrograms which are closely related to the amplitude spectrogram. The main difference is the application of a Mel filterbank which transforms the frequency bins between 64 Hz and 4 kHz to 23 Mel bands. The filters are relatively narrow at low frequencies and become broader at high frequencies.

2.1 Log-Mel-Spectrogram

Write a Matlab function, which computes the Log-Mel-Spectrograms from the input signal. The function header might look like this:

```
mLogMelSpectrogram = ComputeLogMelSpectrogram(vSignal);
```

Here, `mLogMelSpectrogram` is a matrix which contains the Log-Mel-Spectrograms and `vSignal` is a vector whose elements are the time domain samples.

In your function, the signal has to be transformed to the STFT domain first. This can be done with your functions `myWindowing` and `mySTFT` that you developed in your first and your second exercise session. Use the following parameters for your STFT.

- *frame length:* 32 ms
- *frame shift:* 10 ms
- *window function:* periodic Hann window

After you obtained your spectrogram, compute the squared amplitude of the DFT coefficients which may be stored in a matrix named `mSTFTAmplitudeSquared`.

Now, you can apply the Mel-filterbank. It is stored in the file `MelFilterbank.mat`, which is provided in the zip-file you downloaded for this exercise. You load the filterbank parameters to your Matlab environment using the `load` command, i.e., `load('MelFilterbank.mat')`. After that, two new variables, `mMelFilterbank` and `vMelFrequencies`, should appear in your workspace. The filterbank can be applied via a matrix multiplication, e.g., `mMelSpectrogram = mMelFilterbank * mSTFTAmplitudeSquared`. The vector `vMelFrequencies` contains the center frequencies of the Mel filters. For this to work, your spectra have to be stored in the columns of `mSTFTAmplitudeSquared`. Finally, compute the logarithm of your Mel transformed spectrogram for obtaining the Log-Mel-Spectrogram. The frames of the Log-Mel-Spectra in your matrix `mLogMelSpectrogram` will also be referred to as feature vectors throughout this exercise.

Questions

1.1 What may have been the inspiration for the parameters that you should employ for computing the Log-Mel-Spectrogram (64 - 4000 Hz; 32 ms; 10 ms; 23 bands)? Think about the properties of speech production and the human perception.

1.2 Plot the Log-Mel-Spectrograms as 2D plot using `imagesc` for the normally spoken word "stop" of speaker `df`. In order to apply the frequency axis correctly perform the following steps in Matlab. These should be applied for every plot of a Log-Mel-Spectrogram.

```
% Draw the Log-Mel-Spectrogram, where vTimeFrame contains the time instants for
% each frame. The vector 1:23 is the y axis which links each Mel band to an
% integer number.
imagesc(vTimeFrame, 1:23, mLogMelSpectrogram);

% Invert y-axis.
axis xy;

% Set the correct frequencies by exploiting the previously employed integer
% numbering for the Mel bands.
set(gca, 'YTickLabel', round(vMelFrequencies(get(gca, 'YTick'))));
```

a) Which typical speech characteristics can you recognize in the Log-Mel-Spectrograms?

b) Which are not so well represented?

1.3 Plot the Log-Mel-Spectrograms for all normally spoken words of the speaker `df`.

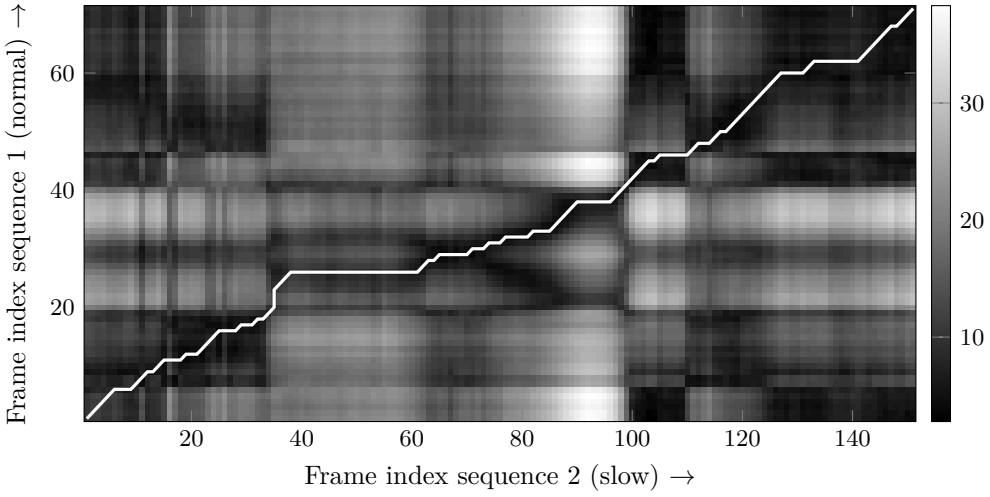
a) Can you distinguish the words by looking at the Log-Mel-Spectrograms?

b) Are there utterances which cannot be easily distinguished?

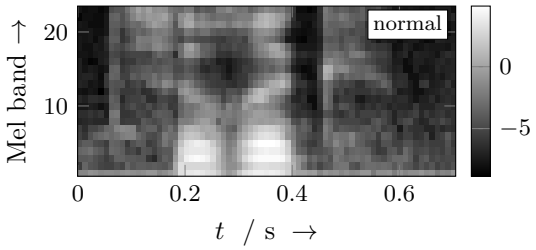
Include some meaningful plots and examples in your report.

3 Dynamic Time Warp

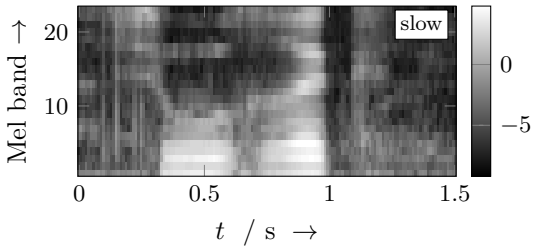
In this part of the exercise, you will implement the DTW algorithm which will be used to recognize an utterance based on a vocabulary which has been previously trained. In speech recognition, often the problem occurs that the speaking tempo of two different recordings, which may contain the same word, is not the same. Consequently, also the lengths of the Log-Mel-Spectrograms differ and therefore, they cannot be easily compared. The DTW tries to determine the similarity between two utterances though they were spoken in a different tempo. For this, it tries to find the best possible time alignment of both utterances. In order to compensate for the differences in both utterances, the DTW algorithm stretches parts in both signals where the speaking tempo is higher compared to the other signal. For example, the word "stop" may be spoken in two different ways: One time with a long "s" sound in the beginning, where the rest of the word is spoken normally, and a second time with a long "o" sound in the ending, where the first part of the word is spoken normally. The DTW would now possibly extend the "s" sound of the second utterance and then the "o" sound of the first utterance to match both signals. Ideally, the speaking tempo in both utterances are the same after the alignment. Technically, the stretching of the faster parts is accomplished by repeating frames in the Log-Mel-Spectrograms.



(a) Distance matrix.



(b) Log-Mel-Spectrograms of the normally spoken utterance (Sequence 1).



(c) Log-Mel-Spectrograms of the slowly spoken utterance (Sequence 2).

Figure 1: Distance matrix between a normally and a slowly spoken word.

Write a function that implements the DTW. It should determine the DTW distance between two sequences. The following sections explain the term DTW distance and how this value is computed. The function may look like this:

```
[DTWCost, mDistanceMatrix, mPath] = DynamicTimeWarp(mLogMelSpectrogramA,
    mLogMelSpectrogramB)
```

The matrices `mLogMelSpectrogramA` and `mLogMelSpectrogramB` contain the two Log-Mel-Spectrograms that should be compared. Further, `DTWCost` is the distance according to the DTW, `mDistanceMatrix` is the distance matrix computed as in Section 3.1 and `mPath` is the optimal path as described in Section 3.2.

3.1 Distance matrix

In order to find the best possible mapping between two utterances, a measure of the similarity is required. Here, we will use the Euclidean distance between the feature vectors. Assume that the Log-Mel-Spectrograms are represented by the vectors $\mathbf{x}^{(1)}[\ell]$, for the first sequence, and $\mathbf{x}^{(2)}[\ell]$, for the second sequence. Here, ℓ indexes the frames while N and M denote the number of frames for the first and the second feature sequence, respectively. The Euclidean distance between the ith frame of the first sequence and jth frame of the second sequence is then defined as

$$d_{i,j} = \sqrt{\sum_k \left(x_k^{(1)}[i] - x_k^{(2)}[j] \right)^2}, \tag{1}$$

where $x_k[\ell]$ is the kth element of the feature vector $\mathbf{x}[\ell]$. All Euclidean distances which can be computed between the first and second sequence are stored in a matrix $\mathbf{D} = (d_{i,j}) \in \mathbb{R}^{N \times M}$ which is called the distance matrix.

In your function `DynamicTimeWarp` compute the distance matrix for the two input spectrograms.

3.2 Search for the optimal path

The repetition of Log-Mel-Spectra in one of the two sequences can be understood as mapping between the feature vectors of the two sequences. For example, for two sequences with $N = M = 4$ the mapping shown in Table 1 could occur. Here, the first frame of the first sequence is repeated four times and after that, the last frame of the second sequence is repeated four times.

step	1	2	3	4	5	6	7
index sequence 1	1	1	1	1	2	3	4
index sequence 2	1	2	3	4	4	4	4

Table 1: Example for a mapping between two feature sequences.

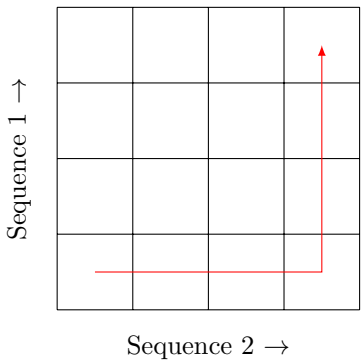


Figure 2: Visualization of the path for the mapping shown in Table 1.

Such a mapping can be visualized by a path which is shown in Figure 2. The path always starts at the position $(1, 1)$ and ends at (N, M). This means, that the DTW always maps the beginnings and the endings of both signals onto each other. Between these two points, different mappings can occur. A horizontal movement of the path means that the feature vectors from the first sequence are repeated and vertical movement means that the vectors from the second sequence are repeated. A diagonal movement means that no repetition occurs.

Such a path is also shown in Figure 1 as a white line. Using this path, the overall distance, or better said, the similarity between the two utterances can be determined. For this, all elements in the distance matrix which are touched by the path are accumulated. Now, in order to find the best possible time alignment, the task is to find a path for which the accumulated distance is minimal.

The optimal path – meaning the path from $(1, 1)$ to (N, M) for which the accumulated distance is minimal – can be obtained using dynamic programming. This approach tests all possible paths in an efficient way. For this, two new matrices are required, e.g., `mAccDistance` and `mPrevious`. `mAccDistance` will store the minimal accumulated distance that is required to reach each position in the distance matrix including (N, M). `mPrevious` stores the direction of the optimal path for each element in the matrix. Using these directions it is possible to reconstruct the optimal path. The inner part of the matrices, i.e., all elements excluding the first row and the first column, can be computed using the following Matlab code which you may use in your own implementations. As the

previous accumulated distances need to be known before the accumulation can be done for a new field, the matrix needs to be built up starting from the element (1, 1).

```
% compute accumulated distance for horizontal, diagonal and vertical step
Horizontal = mAccDistance(n, m - 1) + mDistance(n, m);
Diagonal = mAccDistance(n - 1, m - 1) + 2 * mDistance(n, m);
Vertical = mAccDistance(n - 1, m) + mDistance(n, m);

% compute the minimum cost and remember the direction of the optimal path.
[mAccDistance(n, m), mPrevious(n, m)] = min([Horizontal, Diagonal, Vertical]);
```

The result in `mAccDistance(N, M)`, where `N` and `M` are the sequence lengths N and M, corresponds to the accumulated distance obtained for the optimal path. In order to make the result independent of the length of the feature sequences, a normalization with the path length is required, i.e.,

```
DTWCost = mAccDistance(N, M) / (N + M);
```

While with the DTW distance we now know the cost of the optimal path, the course of this path is not known yet and has to be reconstructed from the matrix `matPrevious`. The second output argument of `min` returns the index of the smallest element in the vector `[Horizontal, Diagonal, Vertical]`. Consequently, a value of 1 would mean that the optimal path origins from the horizontal position for example. As final step in your function, compute the back trace of the optimal path from position (N, M).

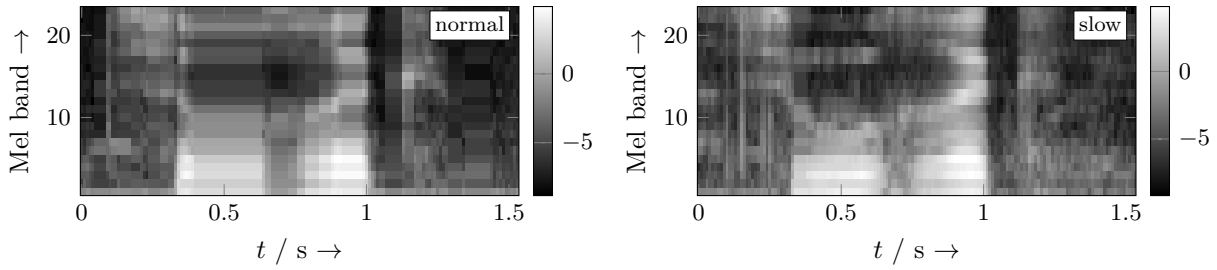


Figure 3: Warped sequences.

Questions

3.1 The Matlab code given above can only be used to determine the elements of `mAccDistance` and `mPrevious` for `n > 1` and `m > 1`. This excludes the first column and the first row, i.e., elements for which `n = 1` or `m = 1` holds.

- What is different for these elements?
- How can the values of the first row and the first column of `mAccDistance` and `mPrevious` be determined?

3.2 Pick the word `stop` of the speaker `df` and plot the distance matrix between the normally and slowly spoken version of it.

- Is it possible to see the optimal path from the matrix?

3.3 With the aid of the optimal path, plot the warped Log-Mel-Spectrogram for the normally and slowly spoken word `stop` of the speaker `df` as shown in Figure 3.

a) Are there any differences which the DTW cannot compensate for?

b) What would be the consequences?

4 Automatic speech recognition

With the feature extraction from Section 2 and the DTW algorithm from Section 3 your automatic speech recognizer is complete. Now, the recognizer needs to be trained. Afterwards, you will perform a recognition experiment, where you will try to identify the utterances recorded by the other groups.

4.1 Training and analysis of the training data

In this exercise, a rather simple method is chosen for training. The DTW recognizer is trained by selecting a set of reference samples for the words that should be recognized. These form implicitly the pool of words that your recognizer is able to distinguish. A new input utterance is compared to all the reference samples using the DTW algorithm and detected as the word for which the total costs are minimal.

In the following, you will assess how reliable the training data is. Compare the normally spoken words of the speaker `df` to themselves using the DTW. This means you should compare all possible word pairs, e.g., *yes* and *yes*, *yes* and *no*, *yes* and *go* and so on from the same speaker and the same speaking speed.

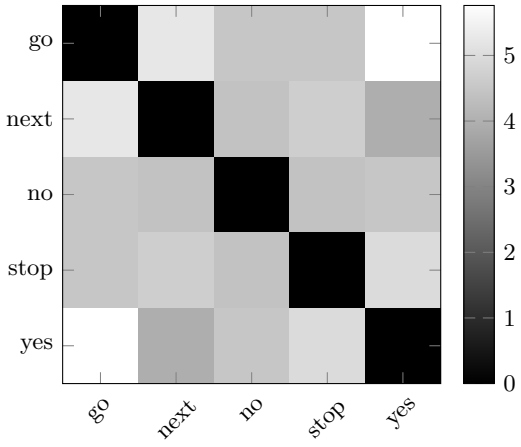


Figure 4: DTW distance of different training words.

Questions

4.1 The training only consists of selecting a set of example utterances which are compared to new test files.

a) What are the disadvantages of such a simple training?

b) What can the classifier learn and what cannot be learned?

4.2 Plot the distances between your utterances as in Figure 4 or display them in a table.

a) Which word pairs are most similar according to the DTW? Which are most different?

b) Do the results agree with your observations that you obtained from the Log-Mel-Spectrograms in Section 2?

4.2 Recognition experiments

In this part, you will conduct some recognition experiments. In order to make it easier for you to perform these experiments, you should develop a small evaluation framework. You can use the following Matlab codes in your implementation.

```
% Set the paths to folders where the audio files are stored.
% Here, it is assumed that all files are stored in the directory 'AudioData', which
% resides in the working directory.
AudioPath = 'AudioData';
```

```
% The dir command can be used to list the files in a directory.
% fullfile is used to generate the path which is passed to dir.
% The result is stored in an array of structs.
stAllFiles = dir(fullfile(AudioPath, '*.wav'));

% This would select all files by the speaker rr.
stRRFiles = dir(fullfile(AudioPath, '*_rr.wav'));

% This would select all files with the word yes.
stYesFiles = dir(fullfile(AudioPath, 'yes_*.wav'));

% This would return the name of the first file found by Matlab.
stAllFiles(1).name

% This would read the first wave file.
[vSignal, dFs] = wavread(fullfile(AudioPath, stAllFiles(1).name));

% You can use the file name to identify the speaker, the spoken word and talking
% speed. Here, end-4 is used to remove '.wav' from the end of the file.
cSplits = regexp(stAllFiles(1).name(1:end-4), '_', 'split');

Word = cSplits{1};      % e.g. yes
Speed = cSplits{2};     % e.g. normal
Speaker = cSplits{3};   % e.g. rr

% For comparing strings you can use strcmpi. The 'i' stands for case insensitive
% meaning that capital and non-capital letters are not distinguished, e.g.
% the strings 'yEs' and 'yes' would be equal.
if strcmpi(Word, 'yes')
        % the word was yes
else
        % the word was not yes
end
```

Write a function which takes a set of reference words `stTrainFiles`, an array of structs, and a single struct `stTestFile` of a new input signal as input arguments and returns the recognized word. The vocabulary given by `stTrainFiles` may be returned by the `dir` command. The function header may look like this:

```
RecognizedWord = RecognizeDTW(AudioPath, stTrainFiles, stTestFile)
```

In this function, you should perform the following steps. You can use the Matlab code, given in the upper example.

- Load the wave file given by `stTestFile` and extract the features.
- For each file in the training set:
  - Load the wave file.
  - Extract the Log-Mel-Spectrogram for the training file.
  - Use the DTW algorithm to compute the DTW distance between the testing and the training sequence.
- Find the utterance in the training set for which the smallest distance to the utterance in `stTestFile` was determined.
- Return the recognized word.

Questions

5.1 Train the DTW recognizer on the speech data of the speaker `df` spoken in normal speed. Try to recognize the slow utterances of the same speaker.

a) How well can the words be identified?

b) Which mistakes do you observe? Can you explain them when you look at the warped spectra and the optimal path?

5.2 Now, try to recognize the words from different speakers. Select the files of `df` including normally and slowly spoken utterances as training set. Determine the word error rate, i.e., the proportion of words which have been erroneously identified. Make sure that none of your training data is used for testing the recognizer.

Identify the files for which the classification failed. Look at the warped Log-Mel-Spectrograms.

a) Can you observe mismatches between the training data and the test data (,e.g., different genders of the speakers)?

b) In the cases an error occurs, can you explain why the DTW fails?

5.3 Repeat this experiment with training data from other speakers. For this, select at least two further training and test sets different from the previous ones.

a) Do you obtain similar results?

b) What can you conclude from this experiment?

c) What influences may be responsible for your results in your recognition experiments?

In your report, describe the recognition the experiments you conducted, e. g. the selected training and test set, and show the error rates that you obtained. Include some plots of the optimal path or warped Log-Mel-Spectrograms to support your explanations, e. g. in cases where the DTW failed.