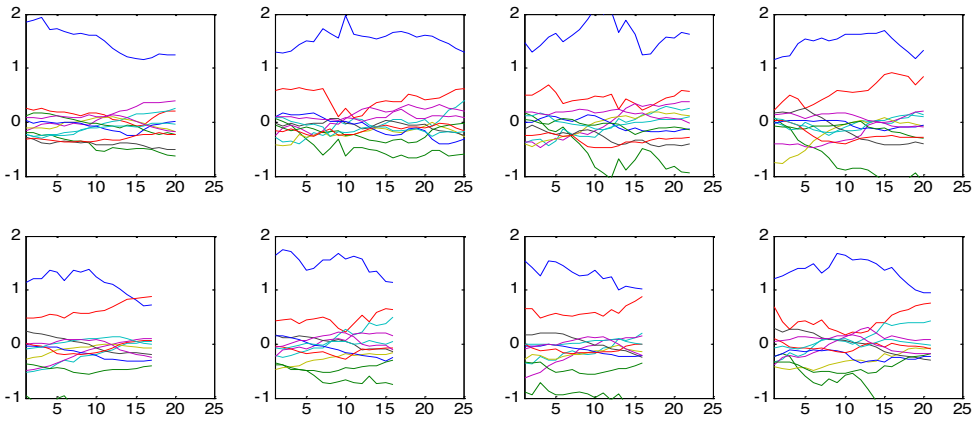
Semester Project Suggestion "Japanese Vowel speaker classification"

Overview. In this project, you will train a 9-class classifier on the basis of a real-life dataset with training examples $(\mathbf{x}_i, y_i)$, where $y_i \in \{1,..., 9\}$. The input patterns are short 12-channel spectral recordings of vocal utterances of a Japanese vowel, recorded from 9 male speakers, and the task is to identify the speaker. This is a classical benchmark dataset, not difficult by today's advanced deep learning standards, but it is not easy to achieve good results. It has been used often some 20 years ago, thus there is an instructive research trail in the literature. I find this a very good challenge for a student semester project. The challenge lies in the circumstance that the input patterns $\mathbf{x}_i$ are *timeseries* and that the temporal organization matters; that the timeseries have different lengths; and that the examples from the different classes are, by human visual inspection, rather similar.

Data. The data are short preprocessed audiorecordings taken from 9 Japanese male speakers who uttered the vowel /ae/. The dataset is obtainable from the UCI machine learning dataset repository at https://archive.ics.uci.edu/ml/datasets/Japanese+Vowels, where you also find a more detailed documentation. The dataset is also included in this materials package for the semester project (files ae.test and ae.train). The dataset comes with a predefined split into a training and a testing dataset. The objective is to identify the speaker, so this is a 9-class classification task.

There are 270 training recordings, 30 from each of the 9 individual apes. There are altogether 370 test recordings, with varying numbers for the 9 classes.

Each individual recording is a short discrete-time time series of 10-30 timesteps length (varying durations of the recordings!), with 12 real-valued channels obtained from a preprocessing filter which transforms the raw microphone signal into *cepstrum coefficients*, which roughly correspond to signal energies in different frequency bands. The figure below shows plots of 4 exemplary recordings from two individuals.



The raw data are contained in the ASCII files ae.train and ae.test. For your convenience I wrote a Matlab script aeDataImport.m which imports these into Matlab structures; the comments in that script should explain the structure of the raw data files. Python users will easily migrate that Matlab code.

Task. Using the training data (and **only** the training data!), design and train a classification algorithm which takes as input a 12-dim time series of the format found in the training data, and returns a "1" or a "2" or a ... "9" class decision. When your classifier is fully optimized (using only the training data! ... a loud call for cross-validation), check its performance on the test data and report the percentage of misclassifications.

Comments. Beware of overfitting! It's a small dataset by today's standards and overfitting lurks behind the corner. There are many design issues where you can hone your ML wits, for instance

- definition / automated extraction of the features in the first place,
- dealing cleverly with the different lengths of the recordings,
- optimizing the number of features (more = more complicated, more discriminative, and more dangerous w.r.t. overfitting)
- regularization.

All in all, it's not trivial at all to come up with a good classifier – the dataset is not easy (small dataset does not mean easy!). Most published papers report a misclassification rate on the test set in the range of about 3-5%, though better is doable: In fact, there is one study where a zero test error was obtained – with a very substantial effort, a number of tricks dedicated to precisely only this dataset, and a lot of parameter optimization. This is not untypical for machine learning: the more specific effort and specific insight one invests on a specific task, the better the result. One does not get super results from applying out-of-the-box tools.