# Assignment 6 - Speech recognition

1. Assignment description / Problem statement

Build a machine learning model that can recognize and classify recordings of single word (“yes”, “no”, “stop”, “go”).

1. Analysis & Preprocessing

Firstly, the word recordings and their corresponding classification IDs were loaded into numpy arrays. Then we inspected the data structure, 4-dimensional numpy array, to analyze how the recordings are stored and represented. Each of 15737 recordings is represented as 62x65 matrix consisting of 62 frames with 65 frequency values each. Because of that we can perceive sound recordings as images. Therefore, to analyze and better understand the recordings we were working with we plotted image-like visualizations, spectrograms, of some of the word recordings, depicting the frequencies over time frames.

1. Used algorithms
2. Convolutional Neural Network (CNN)

As required in the assignment we tuned and trained a CNN-model to classify these recordings. After trying multiple variations of the number of layers and all their parameters, mostly the *activation\_function* and the *kernel\_size*, we defined a Tensorflow sequential model consisting of following 8 layers; 2 pairs of convolution-pooling layers followed by the third convolution, flatten to gain one-dimensional vector that was processed into 64- and eventually into 4-dimensonal vector with each dimension holding a probability of belonging to the corresponding class.

1. Long short-term memory Recurrent Neural Network (LSTM RNN)

As a second model we decided to train a long short-term memory RNN. Using 2 layers only; LSTM that is capable of learning and remembering over long sequences and is followed by the Dense layer with *softmax* activation to conclude the network with 4-dimensonal probability vector as in the first model.

\*After trying multiple approaches for the training/fitting of the model, we ended up using *Adam* as our model optimizer for both CNN and RNN models.

1. Performance

We managed to achieve high accuracies with both models, while the latter, LSTM RNN, consisted of much fewer layers (2) and had a lower tendency for overfitting than the CNN.

1. CNN accuracy: 0.95
2. RNN accuracy: 0.94
3. Reflection / Learning outcomes

LSTM networks are well-suited for tasks involving sequential data, like speech recognition, because they have "memory" in the form of hidden states that can carry information from one step in the sequence to another. This allows them to capture temporal dependencies in the data, which is crucial in speech recognition where the meaning of a sound can depend on the previous sounds.

On the other hand, CNNs are excellent for spatial data and are typically used for image processing tasks. They can capture spatial hierarchies in the data by applying filters to different parts of the input, but they do not capture temporal dependencies as LSTM networks do.

CNNs can be used for speech recognition tasks and achieve good results as we have demonstrated, but they lack the aspect of temporal dependencies which can reduce their accuracies with increasing length of the recordings. In contrast, LSTM networks can handle both the temporal dependencies and the feature extraction from the raw data, making them a popular choice for speech recognition tasks.