# Phoneme recognition in TIMIT with BLSTM-CTC

<http://www.i6.in.tum.de/Main/Publications/Fernandez2008a.pdf>

Connectionist Temporal Classification (CTC) allows developing neural network classifiers using a sequence of labels as the desired output target [5]. Labels correspond to events occurring in the input data sequence, such as phones in a speech data stream.

The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (TIMIT) contains recordings of prompted English speech accompanied by manually segmented phonetic transcripts [2].

TIMIT contains a total of 6300 sentences, 10 sentences spoken by each of 630 speakers from

8 major dialect regions of the United States.

The 61 categories were folded onto 39 categories as described by Lee and Hon [10]. This is shown in table 1.

Tabulka 1: Folding the 61 categories in TIMIT onto 39 categories (from [10]). The phones in

the right column are folded onto their corresponding category in the left column (the phone

’q’ is discarded). All other TIMIT phones are left intact.

|  |  |
| --- | --- |
| aa | aa, ao |
| ah | ah, ax, ax-h |
| er | er, axr |
| hh | hh, hv |
| ih | ih, ix |
| l | l, el |
| m | m, em |
| n | n, en, nx |
| ng | ng, eng |
| sh | sh, zh |
| sil | pcl, tcl, kcl, bcl, dcl, gcl, h#, pau, epi |
| uw | uw, ux |
| — | q |

## Performance:

<https://en.wikipedia.org/wiki/Levenshtein_distance>

Performance was measured as the normalised edit distance (label error rate; LER) between the target label sequence and the output label sequence given by the system.

# Phonemic Similarity Metrics to Compare Pronounciation Methods

<https://homes.cs.washington.edu/~bhixon/papers/phonemic_similarity_metrics_Interspeech_2011.pdf>

**Phoneme error rate** (PER) is the Levenshtein distance between a predicted pronunciation and the reference pronunciation.

In the following two pairs, however, the pair on the right has an un-reasonable pronunciation for “soda”:

S OW D AH S OW D AH

S OW D AA S OW D L

Nonetheless, WER, PER, and Levenshtein distance are the same for these two pairs (100 %, 25 %, and 1, respectively). The WPSM metrics described here are sensitive enough to overcome these limitations.

**Needleman-Wunsch** is a dynamic programming algorithm that finds the maximum similarity score of two strings. Needleman-Wunsch iteratively aligns increasingly long string prefixes. For each prefix pair it chooses the maximum score that results when either the last entry in one prefix is substituted for the last entry in the other, or the last character in one string is aligned with a gap. Applied to pronunciation, the Needleman -Wunsch algorithm requires quantitative phoneme similarity scores, for which various derivation methods have been proposed. Perceptual listening tests have also been used to create a matrix of empirical confusion scores between English phonemes [18], from which substitution costs may be derived [17].

# Mel Frequency Cepstral Coefficient (MFCC) tutorial

<http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/>

The MFCC feature vector describes only the power spectral envelope of a single frame. Deltas and Delta-Deltas carry the information about speech dynamics (trajectories of MFCC coefficients over time).

Deltas ... 1st order differential ... velocity of the MFCC coefficients change

Delta-Deltas ... 2nd order differential ... acceleration of the MFCC coefficients change

# Udacity – NLP Course

**intentions** (what is supposed to happen) -> **utterances** (parts of speech which trigger specific intentions)

# Outline:

1. Input layer – MFCC & Deltas (mostly done)
   1. Features are 12 Cepstrums, 12 Delta Cepstrums, 12 Delta-Delta Cepstrums
2. BLSTM(+CTC) for phoneme/grapheme (**ARPABET**) classification (Acoustic model)
   1. BLSTM to capture the backward and forward context
   2. CTC to remove the need for pre-generated frame labels
   3. large enough corpus for training BLSTM
      1. TIMIT - english (not free)
      2. **ORTOFON v1 – Czech, with phonetic transcript (ok for academic purposes)**
   4. best-path decoding vs prefix search decoding
3. (Language model for phonemes to words)
   1. Maybe a dictionary which maps phonemes to graphemes?
   2. **WFST-based Decoding** (https://arxiv.org/abs/1507.08240)
4. Filter for keywords (keyword spotting)
5. Controlling interface with keywords

# Links:

## Phonemes vs graphemes in ASR:

<http://publications.idiap.ch/downloads/reports/2004/rr04-48.pdf>

## ORTOFON v1: balanced corpus of informal spoken Czech with multi-tier transcription (transcriptions & audio)

<https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2579>