DT2118 Lab3B: Deep Learning for ASR

Mateusz Buda buda@kth.se
Masoumeh Poormehdi Ghaemmaghami mpg@kth.se

May 11, 2016

1 Prepare the database

Number of male and female speakers in the training and test set respectively:

- Training set:
  - Number of woman: 57
  - Number of man: 55
- Test set:
  - Number of woman: 57
  - Number of man: 56

Number of training and test utterances:

- Number of training utterances: 8623
- Number of test utterances: 8700

Number of phonemes (including silence): 22
Number of nodes and arcs in the recognition network:

- nodes: 16
- arcs: 36

2 Train and test G-HMMs

2.1 Feature extraction configuration

```
# Feature extraction configuration
TARGETKIND = MFCC_0_D_A
TARGETRATE = 100000.0
SAVECOMPRESSED = F
SAVEWITHCRC = F
WINDOWSIZE = 200000.0
USEHAMMING = T
USEPOWER = T
PREEMCOEF = 0.97
```

```
NUMCHANS = 40
LOFREQ = 133.33
HIFREQ = 6835
CEPLIFTER = 21
NUMCEPS = 12
ENORMALISE = F
```

`MFCC_0_D_A` means that delta and acceleration coefficients are to be computed and appended to the static MFCC coefficients computed and stored during the coding process [1].

The filterbank have 40 channels and 12 MFCC coefficients should be output.

2.2 HMM definition prototype

It is a continuous density HMM with 5 states in total, 3 of which are emitting.

Definitions indicate that the observation vectors have 39 components (`<VecSize>` 39). It means each ellipsed vector is of length 39. The number, 39, is computed from the length of the parametrised static vector (MFCC 0 $= 13$) plus the delta coefficients ($+13$) plus the acceleration coefficients ($+13$).

Models stored in `hmm1-3` contain the improvement by Baum-Welch re-estimation and models in `hmm4` have the result of adding a short pause and in `hmm5`, middle state of silence (sil) with pause (sp) is tied. For `hmm6-7` again Baum-Welch for more improvements is used. So the difference between models stored in `hmm1-3` and `hmm5-7` is related to the short pause which is used for improvement.

2.3 Test results

```
SENT: %Correct=82.69 [H=7194, S=1506, N=8700]
WORD: %Corr=97.22, Acc=93.69 [H=27788, D=149, S=646, I=1009, N=28583]
```

H is the number of correct labels, D is the number of deletions, S is the number of substitutions, I is the number of insertions and N is the total number of labels in the defining transcription files.

The percentage number of labels correctly recognised is given by:

$$\%Correct = \frac{H}{N} \times 100\%$$

And the accuracy is computed by:

$$\%Accuracy = \frac{H - I}{N} \times 100\%$$

```
Word accuracy: 93.69
Correct words: 97.22
```

3 Train and test GMM-HMMs with increasing number of Gaussians

Here use the MFCC_0_D_A features and train GMM-HMM models with 2, 4, 8 and 16 Gaussian components per state.

Models with 2 Gaussian components per state:

```
SENT: %Correct=89.78 [H=7811, S=889, N=8700]
WORD: %Corr=98.45, Acc=96.37 [H=28141, D=117, S=325, I=596, N=28583]
```

Models with 4 Gaussian components per state:

```
SENT: %Correct=93.18 [H=8107, S=593, N=8700]
WORD: %Corr=99.02, Acc=97.68 [H=28304, D=79, S=200, I=384, N=28583]
```

Models with 8 Gaussian components per state:

```
SENT: %Correct=94.93 [H=8259, S=441, N=8700]
WORD: %Corr=99.39, Acc=98.31 [H=28410, D=57, S=116, I=311, N=28583]
```

Models with 16 Gaussian components per state:

```
SENT: %Correct=96.01 [H=8353, S=347, N=8700]
WORD: %Corr=99.59, Acc=98.66 [H=28465, D=46, S=72, I=264, N=28583]
```

Accuracy and correct words increase with increasing number of Gaussian components. It is because by increasing the number of Gaussian components, each word could be modeled with a Gaussian.

4 Data Preparation

4.1 Training and Validation Sets

To split the training files into training set and validation set we just filtered all men and women speakers that have id starting with 'a' and added them to the validation set. The rest of speakers was added to the training set. We achieved this with the following commands:

```
awk ’/\/man/’ train.lst > train_man.lst
awk ’/\/woman/’ train.lst > train_woman.lst

awk ’/\/a/’ train_man.lst > train_va.lst
awk ’/\/a/’ train_woman.lst >> train_va.lst

awk ’!/\/a/’ train_man.lst > train_tr.lst
awk ’!/\/a/’ train_woman.lst >> train_tr.lst
```

With this strategy we have split the original training set so that validation accounts for 7% and training consists of the remaining 93%. It seems a bit small validation set but it should be enough (we hope so).

4.2 Feature Normalisation

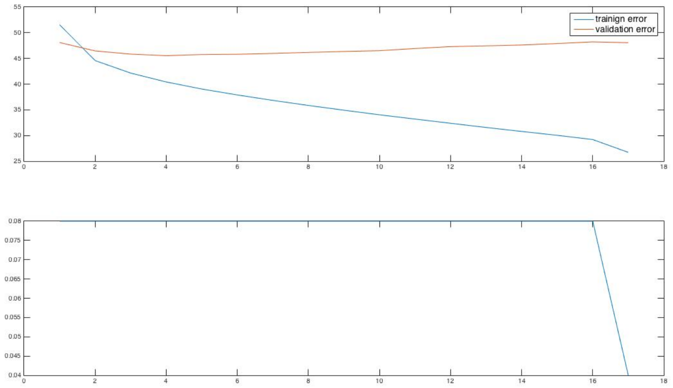
Features, in each file separately, where normalised so that each feature coefficient has zero mean and unit variance. This is not entirely correct because in theory we don't have access to unseen data and we scale and transform them not using the distribution parameters of training data. If we assume that all sets come from the same distribution we should compute mean and variance of available training data and then use them to also transform validation and test sets.

Per-utterance normalization can help when we deal with utterances long enough to have some uniformity to their spectral balance, but the background conditions are rather variable. On the other hand, for short utterances we are more likely to encounter phonetic imbalance in individual utterances that introduce features deformation [2].

5 Phoneme Recognition with Deep Neural Networks

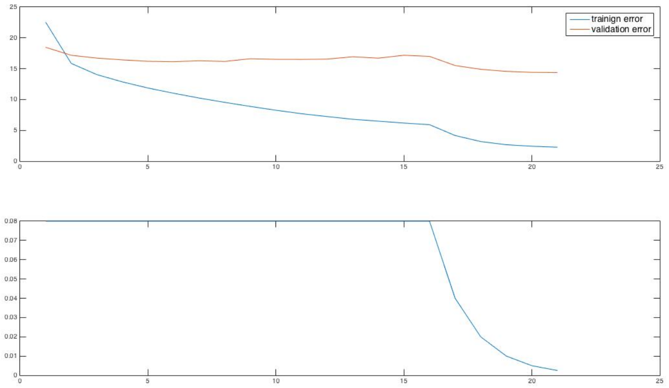
Below we present the training trace of each network. In the upper figure we plotted training and validation error whereas in the lower figure we plotted learning rate. We also included training log output for epochs witch the smallest training and validation error.

1. Deep Neural Network based on normalised `FBANK` features with 4 hidden layers of 1024 units each and rectified linear unit activation functions.



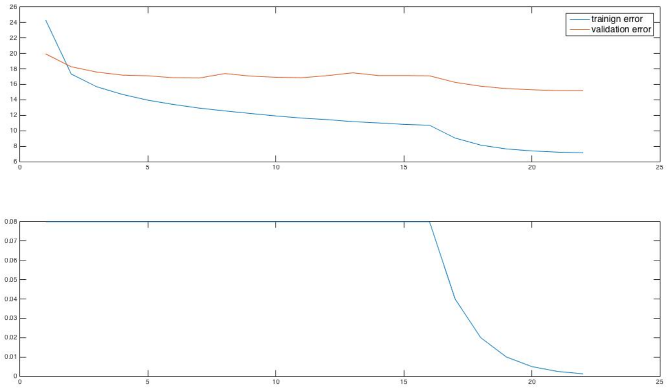
```
epoch 4, training error 40.415726 (%)
epoch 4, lrate 0.080000, validation error 45.531985 (%)
epoch 17, training error 26.720559 (%)
epoch 17, lrate 0.040000, validation error 48.061378 (%)
```

2. Same as before but with a context input window of 5 frames in the past and in the future.



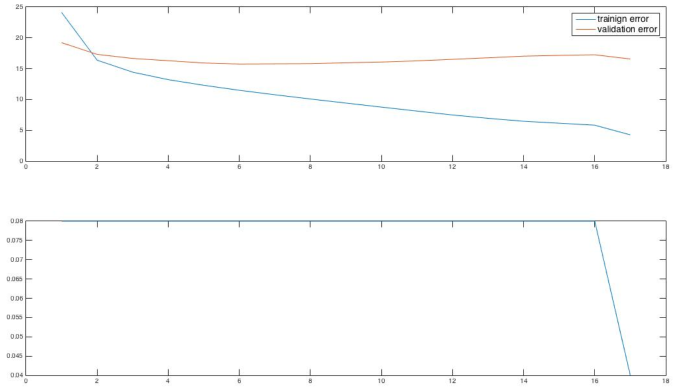
```
epoch 21, training error 2.273660 (%)
epoch 21, lrate 0.002500, validation error 14.377707 (%)
```

3. Same as at the previous point, but with only 256 units per hidden layer.



```
epoch 22, training error 7.143793 (%)
epoch 22, lrate 0.001250, validation error 15.167659 (%)
```

4. Same as at the first point, but based on normalised `MFCC_0_D_A` features instead of normalised `FBANK`.



```
epoch 6, training error 11.499860 (%)
epoch 6, lrate 0.080000, validation error 15.757464 (%)
epoch 17, training error 4.269380 (%)
epoch 17, lrate 0.040000, validation error 16.576423 (%)
```

The best performance on both training and validation set has the second Deep Neural Network based on normalised `FBANK` features with 4 hidden layers of 1024 units each and rectified linear unit activation functions with a context input window of 5 frames in the past and in the future. And this one we chose for further evaluation.

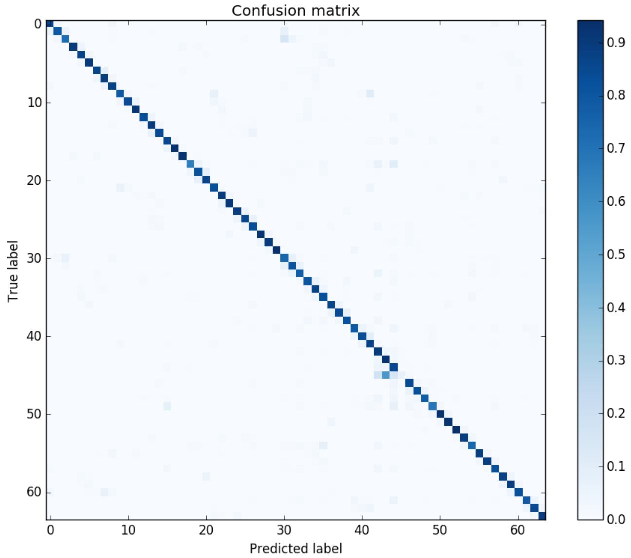


Figure 1: Frame level confusion matrix

Frame level error rate on the test set for this network was 14.488603% that is close to validation error (14.377707%). In the figure 1 we can see that there are mistakes between distant states but many of them lay on upper and lower second diagonal of the confusion matrix.

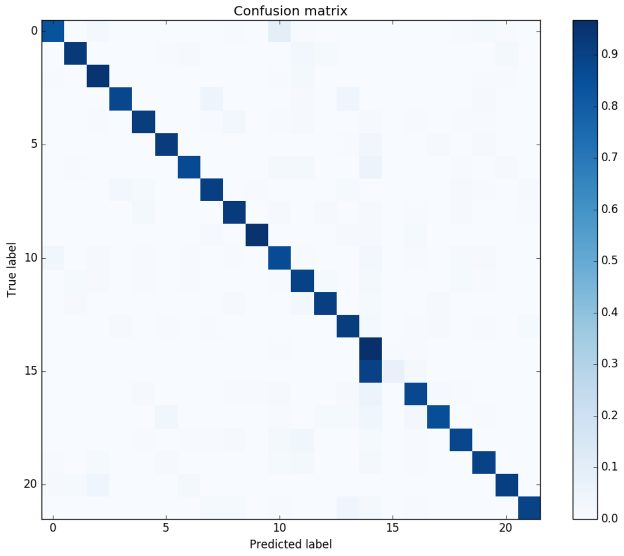


Figure 2: Phoneme level confusion matrix

In the figure 2 we can see that a lot of mistakes on a state level were made within the same phoneme. That improved the error rate to 8.615900%.

References

[1] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland. *The HTK Book, version 3.4*. Cambridge University Engineering Department, Cambridge, UK, 2006.

[2] Dan Ellis. ICSI Speech FAQ: 5.5 What kinds of normalization are there? How do you calculate them?, 2000. Online; accessed 10 May 2016.