# Phoneme Recognition in TIMIT using LSTM networks

**Ludvig Ericson**
ludv@kth.se

**Paulina Hensman**
phensman@kth.se

## Abstract

In this project, we use LSTM networks in different configurations to recognize phonemes in speech. We experiment on different network structures by using one or two layers, and use either LSTM units or the closely related GRU units. We find that a two-layer LSTM network has the best performance, and we also investigate that performance on a per-phoneme basis.

## 1 Introduction

### 1.1 Automatic Speech Recogition

Automatic Speech Recognition (ASR) is a popular research area with many applications. In addition to classic applications like automatic phone help lines there have been recent developments in automatic transcriptions of speech, which could be a great help to the hard of hearing as well as an aid to automatic translations of speech.

One of the most basic components in speech is the *phoneme*. As an example, the word "wash" consists of the phonemes "w," "aa," and "sh." There are about 44 defined phonemes in the English language, but all are not used, and more importantly, all combinations of phonemes are not used. In fact, knowing the preceding phoneme greatly narrows down the possibilities for the following phoneme [1].

This project will compare the performance of different configurations of neural networks for a phoneme recognition task.

### 1.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) consist of several identical units linked together in a chain structure, as shown in fig. 1. This structure makes RNNs suitable for modelling sequences, such as text and speech, where one unit is used for each word or time step. However, RNNs tend to have problems carrying on information through longer sequences due to vanishing or exploding gradients, and therefore certain adaptations are preferred.

### 1.3 Long Short Term Memory

Long Short Term Memory (LSTM) [3] networks are an adaption of RNN. Each unit takes in both a vector representing an input frame, and a state vector. Each unit updates the state vector and passes it on to the next unit in the chain. It also gives output for each step. This structure lets the network retain information from previous timesteps in the state vector.

The important part of the LSTM is the updating of the state vector. This is done in several steps, shown in some detail in fig. 2.

The updating is done in several layers, often called gates. First is the forget gate, determining how much of the state vector should be removed or forgotten. Next is the input gate, determining how

much information to add to the state vector. Finally, the output layer reads from the updated state vector and determines what to output. This careful regulation of the state vector allows the LSTM to save important features indefinitely while forgetting less important ones fast.

### 1.4 Gated Recurrent Unit

The Gated Recurrent Unit (GRU) [4] is a simplification of LSTM, combining the input gate and the forget gate into a single update gate. This introduces a dependency between forgetting and input not present in the LSTM. The state vector is also more exposed to the unit, allowing more information throughput. GRU have been shown to perform empirically better than LSTM in a number of instances [5], but neither has been conclusively shown to be better than the other.

## 2 Method

### 2.1 Data Processing

The speech data was divided into frames of 10 000 $\mu$s each. We calculated the Mel filterbank features for each frame using `HList` from the HTK language processing package [6]. We took every third such filterbank frame and labelled it with the most relevant phoneme for the time frame, as determined by the phoneme transcribed closest in time. Lastly, we prepended the 19 immediately preceding filterbank frames, and padded with zeroes where necessary.

### 2.2 Networks

For the networks, we used the Keras neural networks library [7] with a Theano backend [8]. The structure was a simple sequence of one or two LSTM or GRU layers with 128 units each followed by a dense layer and a softmax activation layer. The networks were trained for 20 epochs, with mini-batches of 4096 samples. We used a categorical cross-entropy loss function, and an RMS-prop optimizer. The network structures are shown in fig. 3

## 3 Experiments

### 3.1 Dataset

We used the TIMIT dataset [9], a corpus of read speech. It has 630 speakers, each reading 10 phonetically rich sentences, for example "The cranberry bog gets very pretty in autumn." The breakdown of the dataset is shown in table 1. After processing as previously described in section 2.1, we obtained 473 177 training samples, and 172 776 validation samples.

### 3.2 Parameters

We made a selection of parameters to experiment with, as follows.

- LSTM Layers (1, 2)

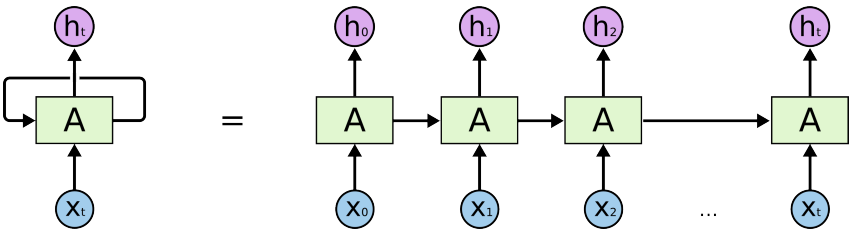


Figure 1: A graphical representation of an RNN, the left picture showing it as a loop going through a single network, and the right showing it as a sequence of identical networks. Image from [2] (modified)

Table 1: Table of breakdown between genders in test and training data sets.

| | **Test** | **Training** | Σ |
|---|---|---|---|
| **Men** | 1 120 | 3 260 | 4 380 |
| **Women** | 560 | 1 360 | 1 920 |
| Σ | 1 680 | 4 620 | 6 300 |

- GRU or LSTM

Generally, more layers tend to mean better generalization, which leads to better results. We will try networks with 1 or 2 layers, with either LSTM or GRU layers.

We made different combinations of the above parameters, ending up with the networks shown in table 2.

Table 2: Table of network parameters.

| **Network** | **Type** | **Layers** |
|---|---|---|
| Network 1 | LSTM | 1 |
| Network 2 | GRU | 1 |
| Network 3 | LSTM | 2 |
| Network 4 | GRU | 2 |

## 4 Results

The accuracy for each net is shown in table 3, and loss function plots in fig. 4.

Table 3: Table of accuracy for each network, where accuracy is defined as the rate of correct predictions in the maximum likelihood sense.

| **Network** | **Accuracy** |
|---|---|
| Network 1 | 46.66% |
| Network 2 | 46.93% |
| Network 3 | 57.26% |
| Network 4 | 46.33% |

A detailed analysis of the best-performing network, the two-layer LSTM network, is presented as a confusion matrix in fig. 5, which is summarized in table 4.

## 5 Discussion & Future work

Network 3, with two LSTM layers, strongly outperformed the other networks. We did expect the two-layer networks to outperform the single-layers ones, but we did not expect this large difference between the LSTM and the GRU networks. There are many things we did not try due to time constraints that may further explain the results.

We ran the networks with large batches and few epochs. It is possible that we would get different results if we used smaller batches and more epochs. There are also many parameters that we never tried to tune, for example the number of units in the LSTM and GRU layers. It is possible that certain parameter settings worked better with LSTM than with GRU, meaning that different settings could make another network perform better. This would be interesting to investigate.

Since two-layer LSTM performed better than single-layer LSTM, it is possible that three layers would be even better. In future work, it would be interesting to try more layers.

For now, we conclude that a two-layer LSTM can outperform single-layer LSTM networks and a two-layer GRU network.

Table 4: Table of accuracy with the three most frequent classifications for each phoneme using the two-layer LSTM network.

| **Phoneme** | **Samples** | **Classifications of phoneme** | | | | | |
|---|---|---|---|---|---|---|---|
| h# | 21 378 | h# | 87.3% | pau | 37.6% | hh | 16.8% |
| sh | 3 139 | sh | 69.5% | ch | 8.1% | epi | 6.9% |
| iy | 8 004 | iy | 61.3% | ey | 10.3% | ih | 8.0% |
| hv | 832 | hv | 49.7% | zh | 4.7% | hh | 3.8% |
| ae | 7 019 | ae | 38.6% | eh | 5.9% | ey | 3.8% |
| dcl | 2 759 | dcl | 45.3% | tcl | 7.1% | gcl | 7.1% |
| d | 992 | d | 43.6% | dh | 6.0% | t | 5.1% |
| y | 1 453 | y | 68.0% | iy | 4.5% | ih | 2.3% |
| er | 3 239 | er | 36.1% | axr | 18.1% | r | 6.4% |
| aa | 4 703 | aa | 45.4% | aw | 15.1% | ay | 14.3% |
| r | 5 242 | r | 59.8% | er | 14.0% | axr | 8.5% |
| kcl | 3 933 | kcl | 65.7% | gcl | 7.6% | tcl | 7.2% |
| k | 2 746 | k | 77.0% | t | 7.0% | hh | 6.8% |
| s | 10 058 | s | 71.9% | z | 11.5% | ch | 8.2% |
| uw | 630 | uw | 65.2% | uh | 7.7% | ux | 7.3% |
| dx | 897 | dx | 34.9% | nx | 2.4% | dcl | 2.3% |
| ih | 4 491 | ih | 35.2% | uh | 15.4% | eh | 11.1% |
| ng | 906 | ng | 56.1% | n | 5.0% | gcl | 1.6% |
| gcl | 1 391 | gcl | 65.0% | g | 8.9% | dcl | 7.6% |
| g | 776 | g | 70.5% | d | 6.0% | k | 4.5% |
| w | 2 851 | w | 65.7% | ao | 4.2% | l | 3.5% |
| epi | 788 | epi | 51.8% | zh | 3.1% | en | 1.7% |
| q | 2 733 | q | 49.2% | oy | 3.8% | tcl | 3.5% |
| ao | 4 746 | ao | 46.2% | aa | 15.6% | aw | 8.3% |
| l | 4 814 | l | 53.0% | el | 18.8% | aw | 14.7% |
| axr | 3 832 | axr | 49.1% | er | 24.3% | r | 8.0% |
| ow | 3 339 | ow | 60.9% | ah | 14.4% | nx | 11.9% |
| n | 4 495 | n | 51.8% | nx | 31.0% | ng | 19.8% |
| m | 3 098 | m | 67.8% | n | 14.8% | en | 9.8% |
| tcl | 4 197 | tcl | 50.6% | dcl | 12.2% | kcl | 9.0% |
| t | 2 466 | t | 56.4% | d | 6.7% | ch | 6.2% |
| ix | 5 113 | ix | 36.5% | ih | 16.3% | uh | 7.7% |
| eh | 4 288 | eh | 27.8% | ae | 10.8% | ey | 7.5% |
| oy | 1 477 | oy | 66.4% | ao | 6.3% | el | 4.8% |
| ay | 4 192 | ay | 47.5% | aa | 10.4% | ae | 6.1% |
| dh | 1 365 | dh | 49.5% | th | 8.0% | d | 6.4% |
| hh | 798 | hh | 45.2% | hv | 10.7% | zh | 1.6% |
| z | 3 618 | z | 71.8% | zh | 14.1% | s | 13.9% |
| pcl | 2 248 | pcl | 68.7% | bcl | 8.6% | tcl | 8.0% |
| ax | 2 207 | ax | 33.0% | uh | 30.8% | ah | 13.5% |
| th | 769 | th | 44.8% | f | 4.8% | dh | 2.7% |
| bcl | 1 722 | bcl | 73.9% | b | 11.5% | pcl | 11.1% |
| b | 506 | b | 65.2% | d | 5.6% | p | 5.0% |
| ux | 2 024 | ux | 66.4% | uw | 10.1% | ih | 5.8% |
| f | 3 133 | f | 68.5% | th | 13.4% | hh | 5.1% |
| el | 1 063 | el | 56.4% | uh | 7.7% | ax | 6.7% |
| v | 1 414 | v | 59.1% | f | 6.6% | hv | 6.2% |
| aw | 1 245 | aw | 27.9% | ae | 4.4% | aa | 1.7% |
| p | 1 409 | p | 71.6% | dh | 5.0% | t | 4.3% |
| ah | 2 655 | ah | 26.0% | eh | 7.1% | ay | 6.5% |
| ey | 3 575 | ey | 44.8% | eh | 7.4% | ae | 4.5% |
| en | 646 | em | 50.0% | en | 41.8% | n | 3.7% |
| ch | 741 | ch | 57.5% | jh | 5.2% | t | 4.0% |
| uh | 563 | uh | 23.1% | ah | 4.4% | ax | 2.1% |
| pau | 2 509 | pau | 35.5% | ax-h | 33.3% | h# | 6.6% |
| jh | 730 | jh | 63.4% | ch | 10.3% | zh | 9.4% |
| nx | 341 | nx | 47.6% | dx | 2.9% | n | 2.2% |
| ax-h | 135 | ax-h | 66.7% | t | 1.0% | th | 0.8% |
| zh | 215 | zh | 45.3% | sh | 3.2% | jh | 1.4% |
| em | 115 | em | 50.0% | en | 1.7% | uw | 1.4% |
| eng | 13 | en | 0.7% | ux | 0.1% | n | 0.1% |

**Acknowledgments**

The computations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at the Center for High Performance Computing (PDC), Royal Institute of Technology.

## References

[1] Xuedong Huang et al. *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice Hall PTR, 2001.

[2] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 05/24/2016).

[3] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[4] Kyunghyun Cho et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078* (2014).

[5] Junyoung Chung et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling". In: *arXiv preprint arXiv:1412.3555* (2014).

[6] S.J. Young and Sj Young. "The HTK Hidden Markov Model Toolkit: Design and Philosophy". In: *Entropic Cambridge Research Laboratory, Ltd* 2 (1994), pp. 2–44.

[7] François Chollet. *keras*. https://github.com/fchollet/keras. 2015.

[8] Theano Development Team. "Theano: A Python framework for fast computation of mathematical expressions". In: *arXiv e-prints* abs/1605.02688 (May 2016). URL: http://arxiv.org/abs/1605.02688.

[9] JS Garofolo et al. "TIMIT Acoustic-Phonetic Continuous Speech Corpus, Linguistic Data Consortium, Philadelphia". In: (1993). DOI: LDC93S1.

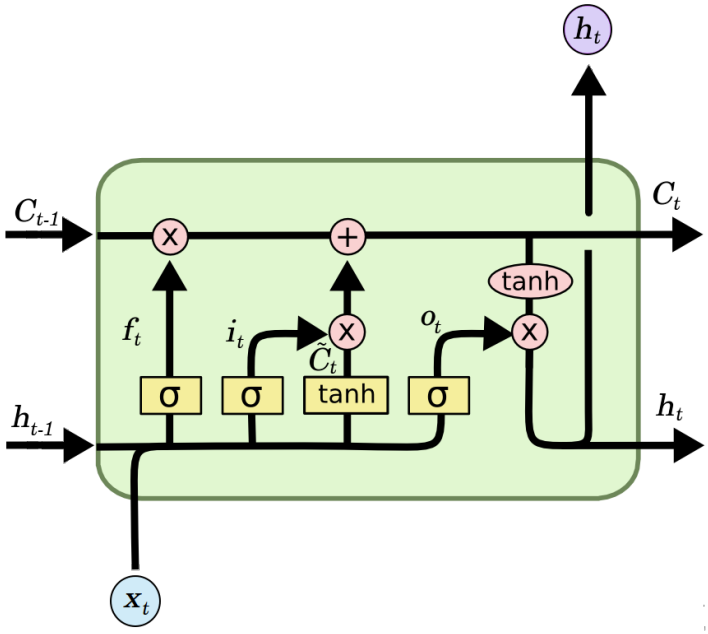


Figure 2: A block in an LSTM network. The input activation at time $t$ is denoted $x_t$, the output at that time $h_t$, the cell state $C_t$, the forget gate $f_t$, input gate $i_t$, and output gate $o_t$. Image from [2] (modified to add detail).

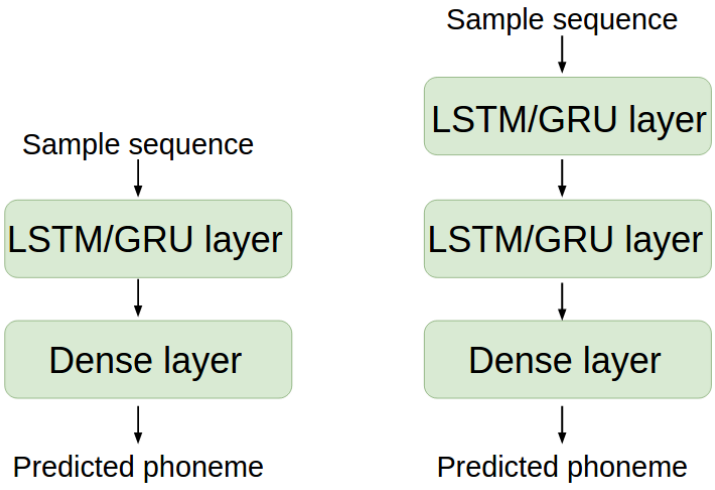


Figure 3: Figures of our networks. As the figure shows, they are rather simple. The LSTM/GRU layers each have 128 units.

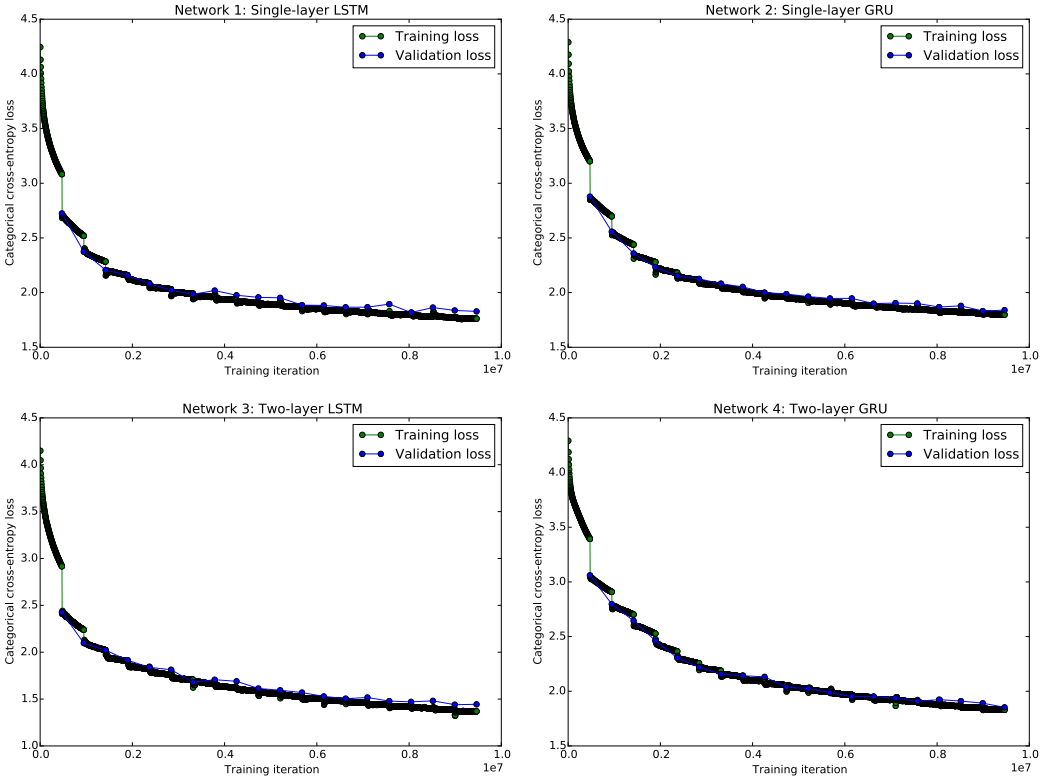


Figure 4: Plots of loss function on both training and validation sets for the four networks. The training loss is calculated for each iteration, whereas validation loss is only calculated at the end of an epoch. The validation data thus serves to demarcate epochs.

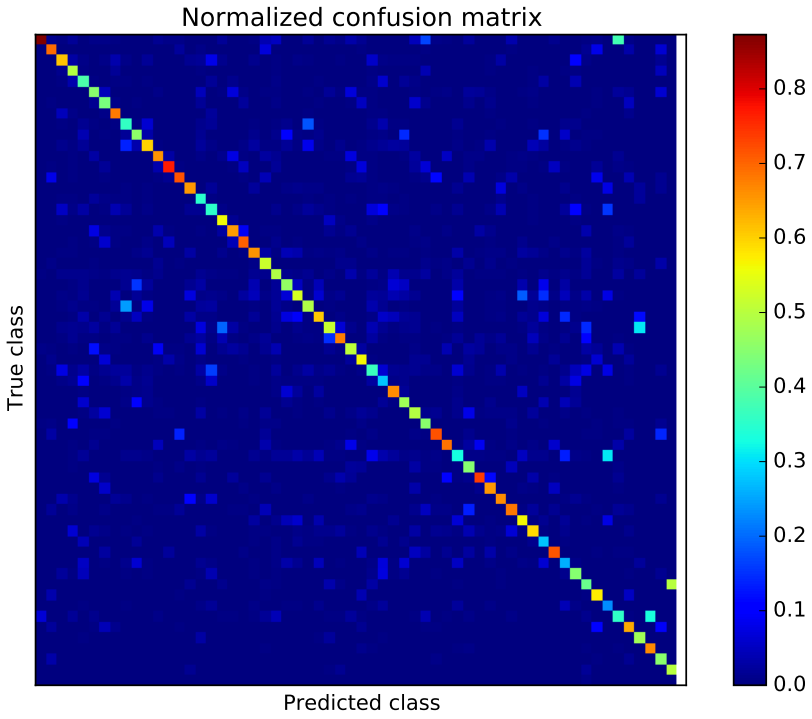


Figure 5: Normalized confusion matrix for the two-layer LSTM network. The diagonal is quite clearly highlighted, and this is expected as this represents the correct classifications.