University of Groningen

Introduction to Neural Networks

Predicting Surname Origin

Authors:
Felix Zailskas *S3918270*

October 25, 2021

rijksuniversiteit groningen

Contents

1 Introduction

In this report I will design and test a recurrent neural network to predict the origin of given surnames. The surnames can have one of these classes: American, Belgian, Dutch, Indian, Italian or Zimbabwean.
Multiple design structures will be compared and tested for different hyper parameters. Results of the different methods used will be discussed.

2 Method

2.1 Data Set

The used data set consists of 4962 surnames and their corresponding origins. 20% ($N = ...$) of the data were used as a test set while the remaining 80% ($N = ...$) of the data were used to train the different prediction models.

2.2 Preprocessing

To prepare the given data fro the used neural network structures a series of preprocessing steps need to be taken.
Firstly, the surnames are tokenized and padded to 20 characters so that they are represented by a vector of numbers. This encoding is then fed into an embedding layer of the network architecture.
Secondly, the output classes are encoded using a one hot encoding to allow the network to predict probabilities for each of the output classes.

2.3 Model Structure

Each tested model is comprised of the following layers: First the encoded input is fed into an embedding layer to create a vector representation of the current surname. These vectors are then passed to a RNN structure that evaluates one character per time-step. The RNN structures investigated in this report are a SimpleRNN, a LSTM and a bi-directional LSTM.

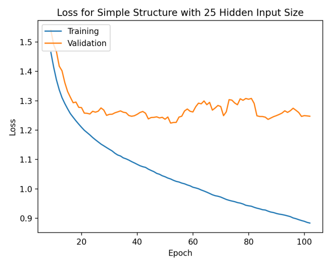
2.4 Training

For each different model type the maximum training epochs are set to 1000. This value has been shown to perform satisfactory in previous experiments in the lab exercises of this course. A validation split was used in each training epoch to avoid over-fitting during the training process. 20% of the training data is not used to train in every given epoch, but rather to test the newly set weights and biases. After 50 epochs with no improvement in the validation loss the training process is finished. Different hidden input sizes were tested to investigate a proper amount of hidden inputs. For each network structure hidden input sizes of 25, 50, 100 and 500 were tested. Each of these trained weights and biases is saved to allow the user to use the trained networks without having to train the network again. Training processes ranged from ca. 2min to ca. 40min.

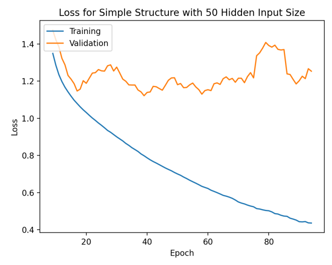
3 Results

In this section I will present the accuracy and loss curves for the different models during training. Furthermore, I will present the accuracy for each model on the training and on the test set.

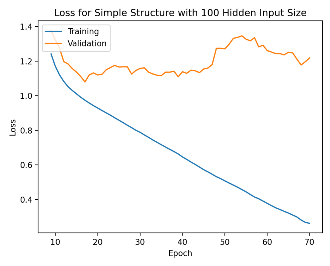
3.1 Simple RNN



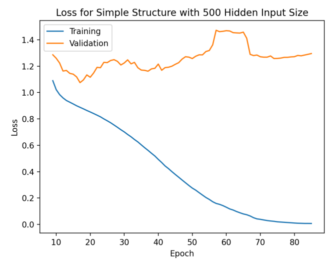
(a) Loss for the SimpleRNN with hidden dimension 25.



(b) Loss for the SimpleRNN with hidden dimension 50.



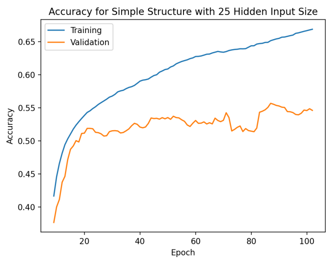
(c) Loss for the SimpleRNN with hidden dimension 100.



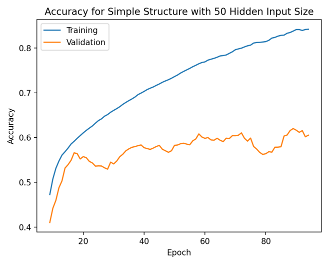
(d) Loss for the SimpleRNN with hidden dimension 500.

Figure 1: Loss for the SimpleRNN with different hidden dimensions for the training and validation set during training.

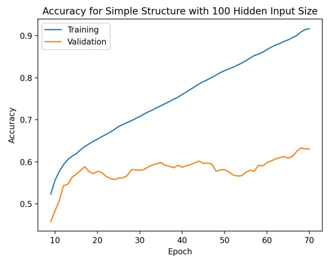
In Figure 1 it can be seen that the loss on the training set is steadily decreasing over all training epochs and for all hidden dimensions. The validation loss however stops decreasing after roughly 20 epochs for all hidden dimensions.



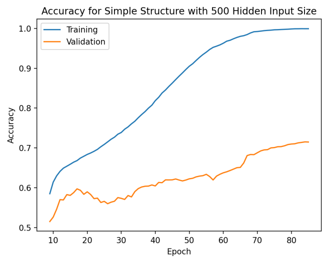
(a) Accuracy for the SimpleRNN with hidden dimension 25.



(b) Accuracy for the SimpleRNN with hidden dimension 50.



(c) Accuracy for the SimpleRNN with hidden dimension 100.

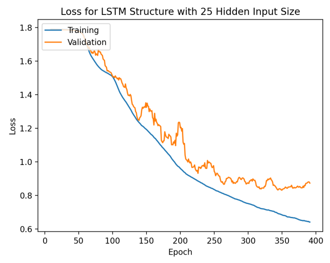


(d) Accuracy for the SimpleRNN with hidden dimension 500.

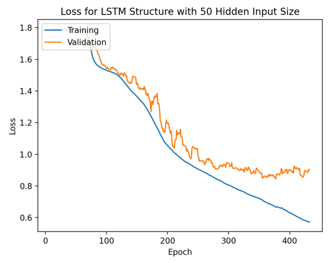
Figure 2: Accuracy for the SimpleRNN with different hidden dimensions for the training and validation set during training.

In Figure 2 it can be seen that the accuracy on the training set is steadily increasing over all training epochs and for all hidden dimensions. The validation accuracy however stops increaseing after roughly 20 epochs for all hidden dimensions, except for the hidden dimension 500, where a less prominent but steady increase can be seen over all epochs.

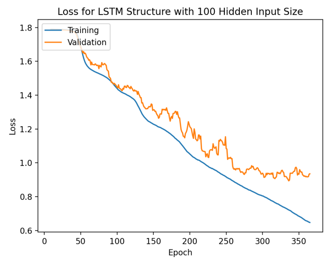
3.2 LSTM



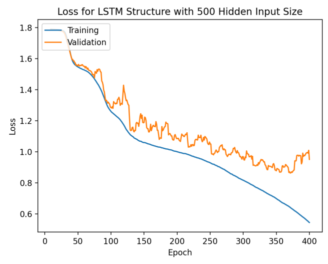
(a) Loss for the LSTM with hidden dimension 25.



(b) Loss for the LSTM with hidden dimension 50.



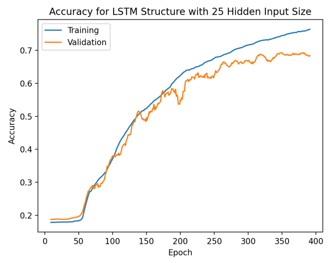
(c) Loss for the LSTM with hidden dimension 100.



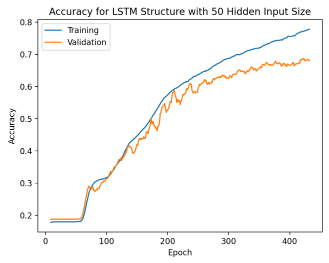
(d) Loss for the LSTM with hidden dimension 500.

Figure 3: Loss for the LSTM with different hidden dimensions for the training and validation set during training.

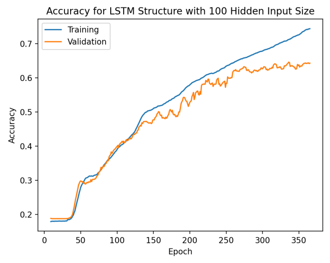
In Figure 3 it can be seen that the loss on the training set is steadily decreasing over all training epochs and for all hidden dimensions. The validation loss also decreases constantly for hidden dimensions 25, 50, and 100 for roughly the first 300 epochs. For hidden dimension 500 the validation loss decreases with a lower rate from roughly epoch 150. Both the validation and training loss curves are very close to each other for all hidden dimensions.



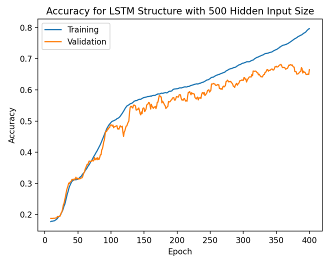
(a) Accuracy for the LSTM with hidden dimension 25.



(b) Accuracy for the LSTM with hidden dimension 50.



(c) Accuracy for the LSTM with hidden dimension 100.

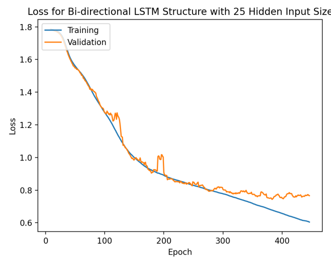


(d) Accuracy for the LSTM with hidden dimension 500.

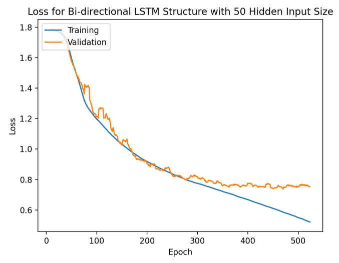
Figure 4: Accuracy for the LSTM with different hidden dimensions for the training and validation set during training.

In Figure 4 it can be seen that the accuracy on the training set is steadily increasing over all training epochs and for all hidden dimensions. The validation accuracy also increases constantly for hidden dimensions 25, 50, and 100 for roughly the first 300 epochs. For hidden dimension 500 the validation accuracy increases with a lower rate from roughly epoch 150. Both the validation and training accuracy curves are very close to each other for all hidden dimensions.

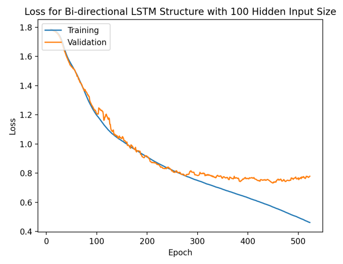
3.3 Bi-Directional LSTM



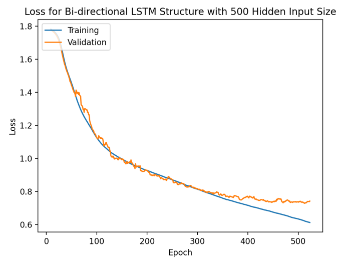
(a) Loss for the Bi-directional LSTM with hidden dimension 25.



(b) Loss for the Bi-directional LSTM with hidden dimension 50.



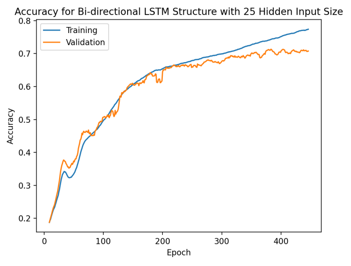
(c) Loss for the Bi-directional LSTM with hidden dimension 100.



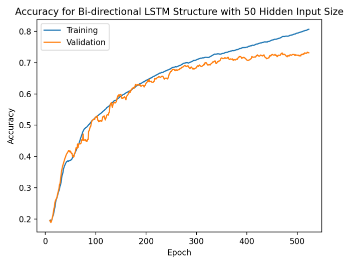
(d) Loss for the Bi-directional LSTM with hidden dimension 500.

Figure 5: Loss for the Bi-directional LSTM with different hidden dimensions for the training and validation set during training.

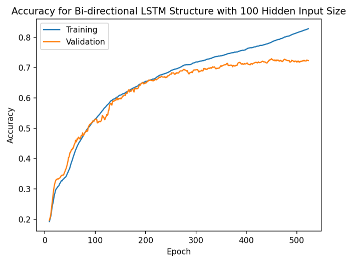
In Figure 5 it can be seen that the loss on the training set is steadily decreasing over all training epochs and for all hidden dimensions. The validation loss also decreases constantly for all hidden dimensions for roughly the first 300 epochs. Both the validation and training loss curves are very close to each other for all hidden dimensions.



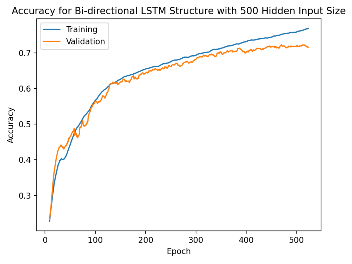
(a) Accuracy for the Bi-directional LSTM with hidden dimension 25.



(b) Accuracy for the Bi-directional LSTM with hidden dimension 50.



(c) Accuracy for the Bi-directional LSTM with hidden dimension 100.



(d) Accuracy for the Bi-directional LSTM with hidden dimension 500.

Figure 6: Accuracy for the Bi-directional LSTM with different hidden dimensions for the training and validation set during training.

In Figure 6 it can be seen that the accuracy on the training set is steadily increasing over all training epochs and for all hidden dimensions. The validation accuracy also increases constantly for all hidden dimensions for roughly the first 300 epochs. Both the validation and training accuracy curves are very close to each other for all hidden dimensions.

3.4 Accuracy

Hidden Dimension / RNN Structure	25	50	100	500
Simple	60.95 / 49.65	82.21 / 61.73	81.36 / 58.01	94.05 / 68.98
LSTM	73.85 / 64.05	78.23 / 63.04	69.09 / 61.53	77.00 / 63.24
Bi-LSTM	75.28 / 68.98	77.85 / 69.39	80.95 / 72.91	76.11 / 71.60

Table 1: Accuracy of the different RNN structures for different hidden dimensions for the Training set and the Test set.

As can be seen in Table 1 for the simple RNN structure the best results are achieved for a hidden dimension of 500. Surprisingly, there is a decrease in test

accuracy from hidden dimension 50 to 100. However, the test accuracy of the network is quite similar for both hidden dimensions 50 and 100. There are big increases in test accuracy from hidden dimension 25 to 50/100 and from 50/100 to 500 of roughly 10% each.
For the LSTM network the increase in hidden dimensions did not seem to have a big impact on testing accuracy. The best results are achieved for hidden dimension 25, but the difference in testing accuracy for the different hidden dimensions is at most 2.5%.
For the bi-directional LSTM network the increase in hidden dimensions did not seem to have a big impact on testing accuracy, but bigger than for the LSTM network. The best results are achieved for hidden dimension 100.

3.5 Conclusion

From the loss and accuracy curves for the simple RNN structure in Figure 2 and Figure 1 it can be seen that the training and validation curves are quite far from each other. This might indicate that the network structure cannot pick up on the complexity of the given problem. This is also reflected in the accuracy values in Table 1. For the LSTM and bi-directional LSTM networks the training and validation curves in Figure 4, Figure 3 and Figure 6, Figure 5 are very close. This indicates that these network structures are a better fit for the problem. In Table 1 we can also see that generally the LSTM and bi-directional LSTM perform better than the simple RNN. The best results have been achieved for the bi-directional LSTM with hidden dimension 100. Hence, this network will be submitted for the blind test.