Signal peptide prediction

Timothy Bergström

January 15, 2018

Contents

Abstract

A neural network classifier was created to predict signal peptides in proteins. The model was trained with about 1 million samples of protein sequences, which yielded a classification accuracy of 94.2% on the test set. The model was tested on predicting signal peptides in the Humans and Mouse proteome, which yielded 93.0% and 93.6% accuracy respectively in signal peptide prediction. The model has shown that it is capable of learning general structures of signal peptides and minor improvements could be made to increase the prediction accuracy even more.

1 Introduction

Signal peptides (SPs) are short regions in peptides that are on average between 16 to 30 amino acids long, but can be as short as 8 and as long as 65 amino acids [1]. SPs function is to translocate proteins to different parts of the cell, such as insertion into membranes, moving proteins to organelles or to help the cell excrete the proteins [2]. When analyzing an organisms proteome, the SPs can give an indication where in the cell the proteins are translocated to, which could be of an interest when researching an organism. Not all proteins contain SPs, which is why signal peptide prediction is useful for determining the layout of the cell and for annotating organisms proteomes.

Due to the large amount of peptide data available, manually annotating each peptide would require many man-hours, while creating and using a classifier for SPs would be more efficient.

2 Methods and models

2.1 The structure of an Signal Peptide

SPs are located near the N-terminal of a protein and have three regions; the n-region, the h-region and the c-region.

The n-region is the first region, which usually contains positively charged amino acids, making the region very polar and hydrophilic. Then comes the h-region, which contains mostly hydrophobic amino acids that forms an α-helix and lastly the c-region, which is usually a sequence of 3 to 8 non-charged polar amino acids. After the c-region, there is usually a cleavage site that have a tendency to be surrounded by small, uncharged amino acids.

The problem with signal peptide prediction is that polar regions in trans-membrane proteins can mistakenly be predicted as the h-region of a SP, which makes predicting SPs in trans-membrane proteins difficult.

2.2 Design choices

The neural network was written in python with Keras 2.0 and Tensorflow 1.4 as the backend. Keras can be used to quickly prototype and develop neural networks while the Tensorflow backend can utilize a gpu to run the calculations.

2.3 Data sets

A small sample of region-annotated proteins were provided by the course in `fasta` format, containing mostly eukaryotic proteins. The samples also contained information if the proteins were transmembrane or not. The data set contained both positive (contains a SP) and negative samples (does not contain a SP). Some samples was extracted from NCBI ftp server, by extracting proteins annotated with signal peptide. Few positive samples and a large set of negative samples were obtained, mostly bacterial proteins. Many of the positive samples were scraped from Signal Peptide database. A majority of the samples were bacterial proteins. Most samples were extracted from Swissprot protein database. Mostly eukaryotic proteins, both negative and positive.

Proteomes from Homo sapiens and Mus musculus were obtained from Swissprot, a database of high-quality manually annotated proteins [3].

The training data sets were shuffled and divided in a train set, validation set and test set.

Database	Positive	Negative	Contains
Swissprot	554 196	573 089	Misc eukaryotic proteins (not including Human or Mice)
SP database	13 098	0	Misc eukaryotic proteins (including Human and Mice)
Proline db	1984	0	Misc eukaryotic proteins (including Human and Mice)
Course data set	1320	1334	Misc proteins, from non_TM and TM proteins

Table 1: Training data sets

Data set	Positive	Negative	From
Human proteome	18 807	144 573	Swissprot + Ensembl
Mouse proteome	10 765	72 675	Swissprot + Ensembl
TM	45	247	Course data set
non_TM	1147	1087	Course data set
TM + non_TM	1320	1334	Course data set

Table 2: Query data sets

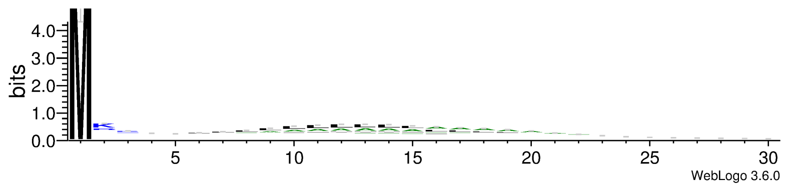

Figure 1: Sequence logo of from all positive samples [4]

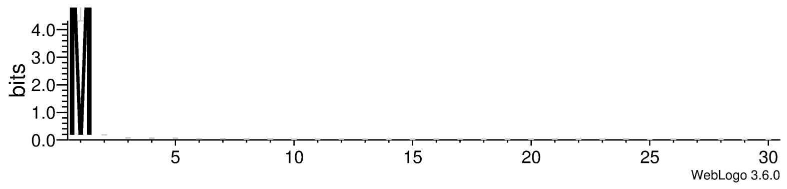

Figure 2: Extracted data from all negative samples [4]

2.4 Data preprocessing

The protein sequences (samples) were contained in `.fasta` files. The first amino acid was removed, due to high bias towards some amino acids, mainly Methionine as seen in figure 1 and 2. The samples were then cleared from all amino acids that were not one of the 20 standard amino acids. Unknown amino acids were labeled as X. The following 30 amino acids from each sample were used for the training data. Sequences that were too short were padded with X. Each amino acids were then one-hot encoded and merged into one 2d array for each sample, creating a 3d array for all the samples. Visualization of one sample can be seen in appendix 5.1, figure 21. Negative samples were labeled as 0 and positive samples were labeled as 1. The data was lastly undersampled, making the ratio between negative and positive samples 50/50 to prevent overfittning.

2.5 Neural Network structure

The neural network was based on bidirectional LSTMs, due to LSTMs effectiveness with sequences and pattern recognition [5]. The bidirectionality of the network was added due to its success in other areas of machine learning with sequences and time-series predictions [6] [7]. The input was one 2d array and the output was one float, ranging from 0 to 1. A regularization layer was added between the LSTM layers to prevent overfitting.

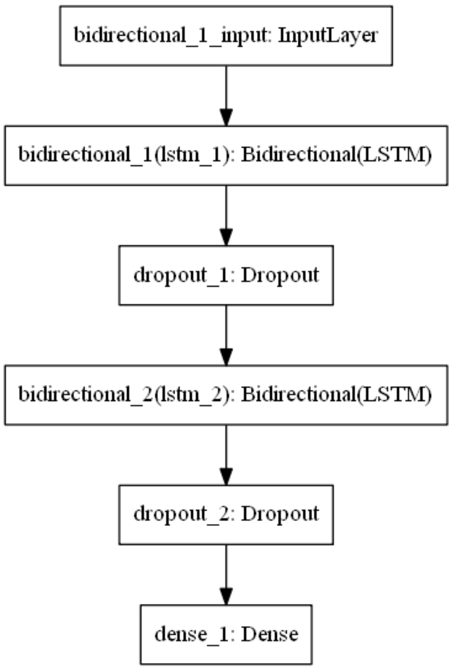
bidirectional_1_input: InputLayer
bidirectional_1(lstm_1): Bidirectional(LSTM)
dropout_1: Dropout
bidirectional_2(lstm_2): Bidirectional(LSTM)
dropout_2: Dropout
dense_1: Dense

Figure 3: Prediction model

2.6 Related works

There are multiple softwares and websites that predicts signal peptides, such as SignalP, SignalBlast and Predisi to name a few [8] [9] [10]. Neural Networks (NN) are the most used methods while some sites use Hidden Markov Models (HMM) instead. Similar methods can used to predict other biological problems, such as motifs and trans-membrane regions in proteins.

3 Evaluating model

3.1 Model performance

The model performed very well and showed no signs of overfitting during training, which can be seen in figure 5. A model that has overfitted can be seen in figure 22 in appendix 5.2.

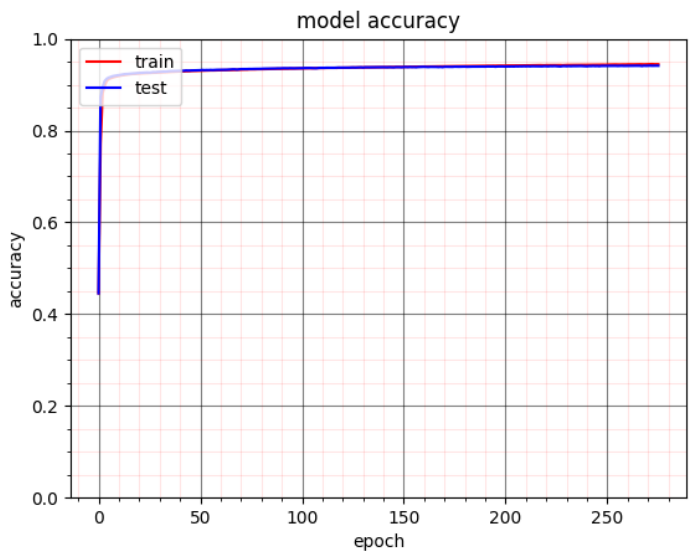

Figure 4: Model accuracy during training

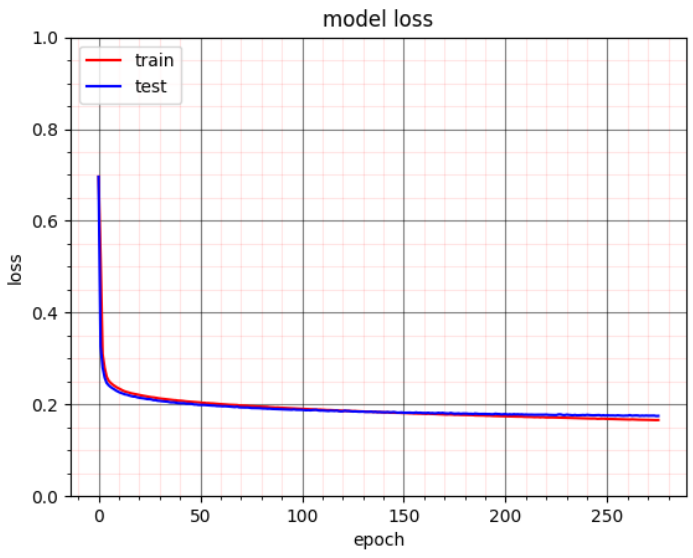

Figure 5: Model loss during training

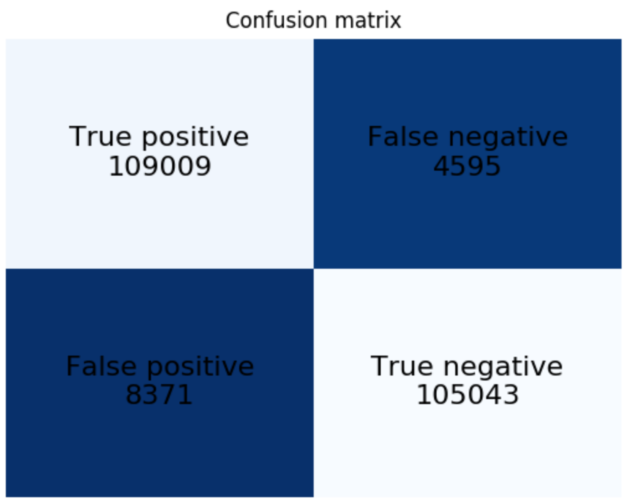

Figure 6: Confusion matrix on test set after training

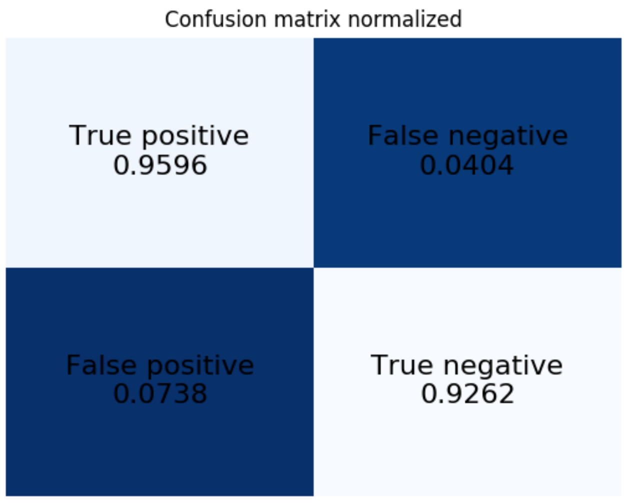

Figure 7: Normalized confusion matrix on test set after training

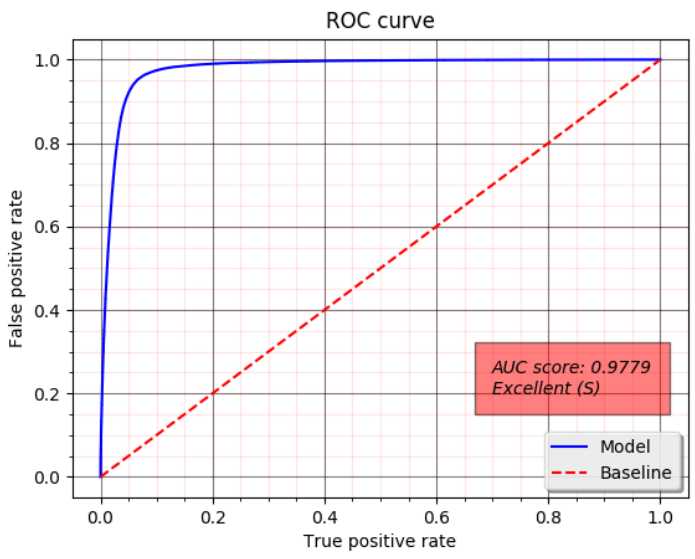

Figure 8: ROC curve on test set after training

Statistical value	value
Accuracy	0.9429
Precision	0.9287
Sensitivity	0.9596
Specificity	0.9262

Table 3: Statistical values: Training

3.2 Signal peptide detection in transmembrane proteins

3.2.1 Non TM proteins

The model performed better with non TM proteins than with the training data set. All statistical values are high, showing no signs of overfitting to the data.

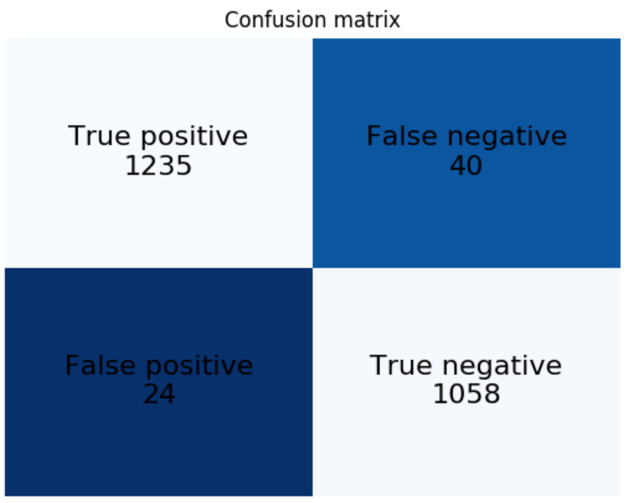

Figure 9: Confusion matrix on non TM proteins

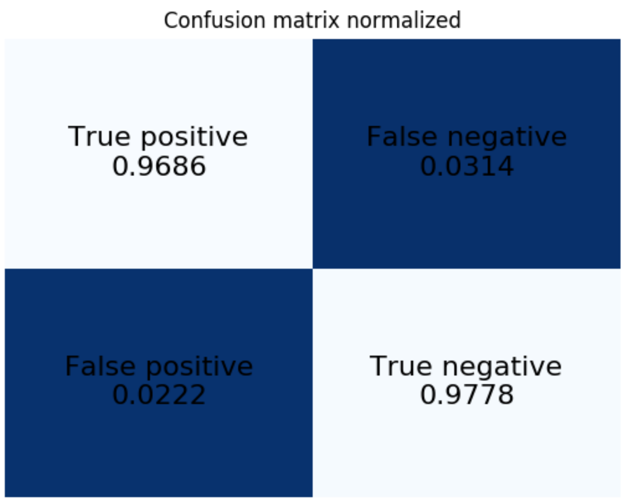

Figure 10: Normalized confusion matrix on non TM proteins

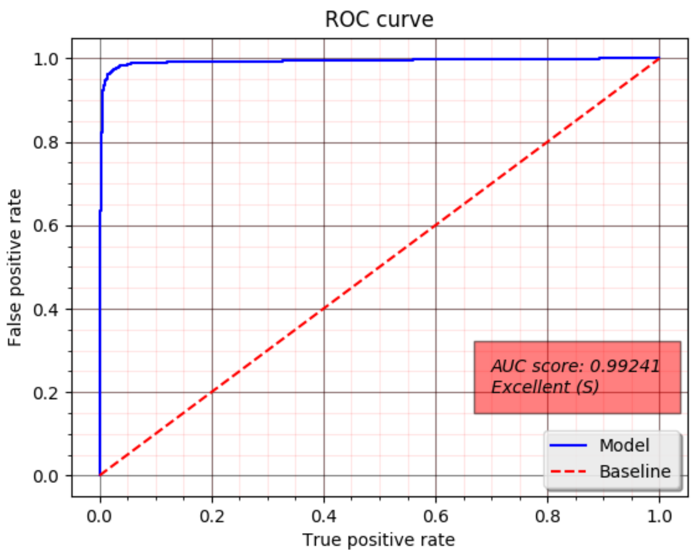

Figure 11: ROC curve on non TM proteins

Statistical value	value
Accuracy	0.9728
Precision	0.9809
Sensitivity	0.9686
Specificity	0.9778

Table 4: Statistical values: non TM proteins

3.2.2 TM proteins

The precision is rather low compared to non TM, but it still has an acceptable prediction peformance.

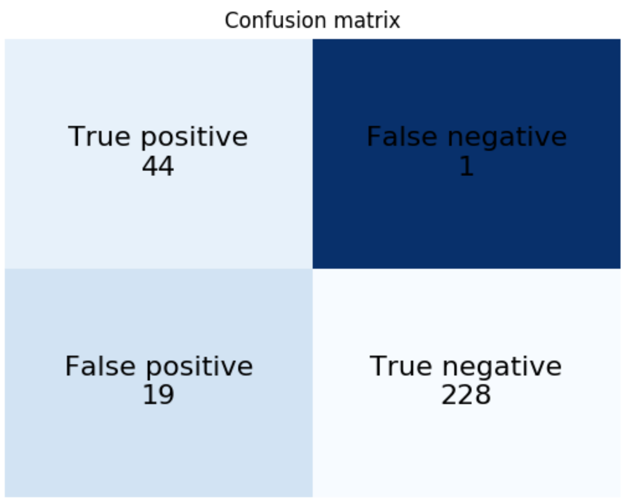

Figure 12: Confusion matrix on TM proteins

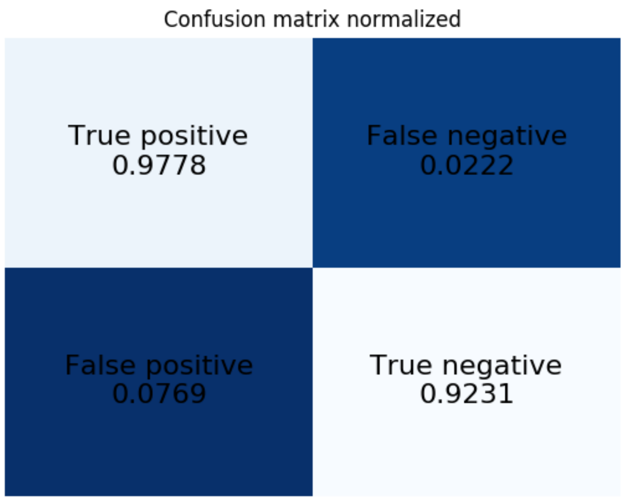

Figure 13: Normalized confusion matrix on TM proteins

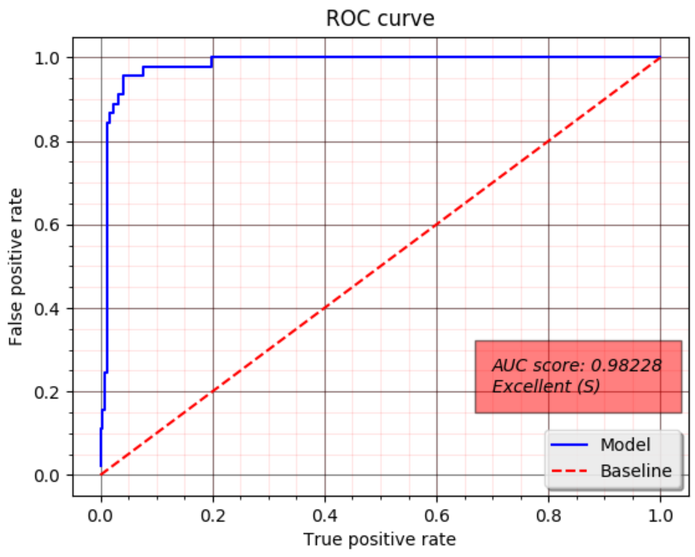

Figure 14: ROC curve on TM proteins

Statistical value	value
Accuracy	0.9315
Precision	0.6984
Sensitivity	0.9778
Specificity	0.9231

Table 5: Statistical values: TM proteins

3.3 Signal peptide detection in proteomes

3.3.1 Human proteome

As the TM proteins, the precision is rather low compared to the training data set and the non TM data set.

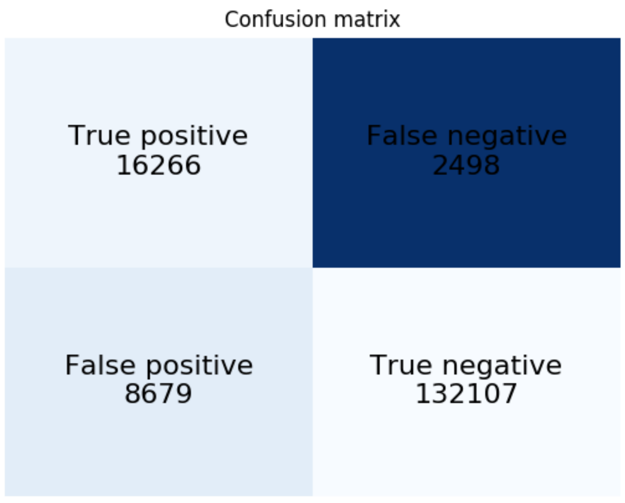

Figure 15: Confusion matrix on human proteome

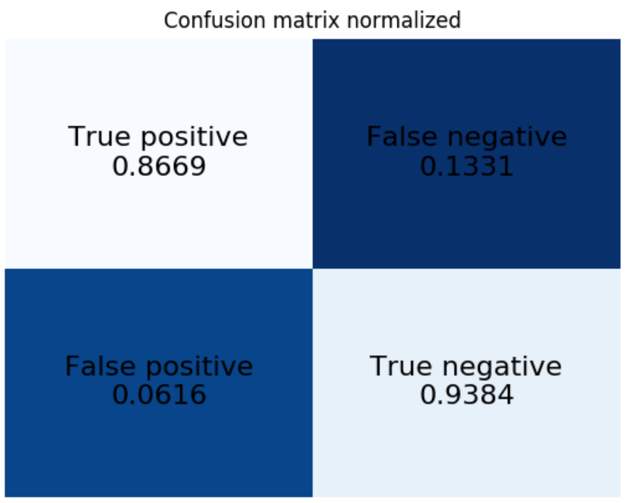

Figure 16: Normalized confusion matrix on human proteome

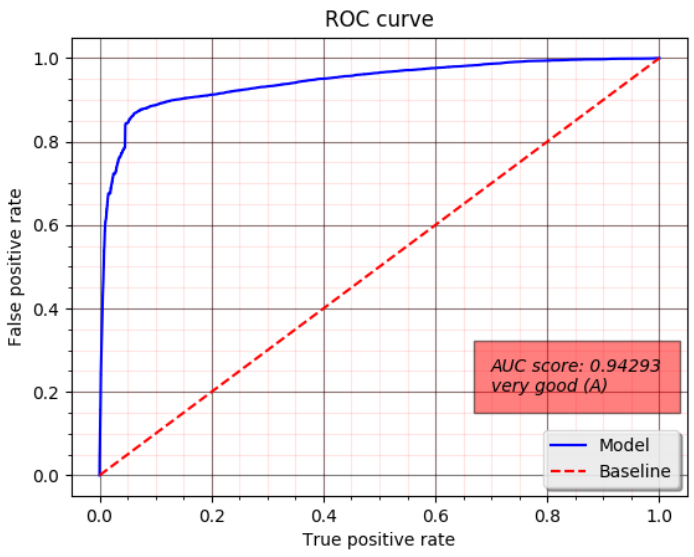

Figure 17: ROC curve on human proteome

Statistical value	value
Accuracy	0.9299
Precision	0.6521
Sensitivity	0.8669
Specificity	0.9384

Table 6: Statistical values: Human

3.3.2 Mouse proteome

Precision is higher than the Human proteome, but still rather low.

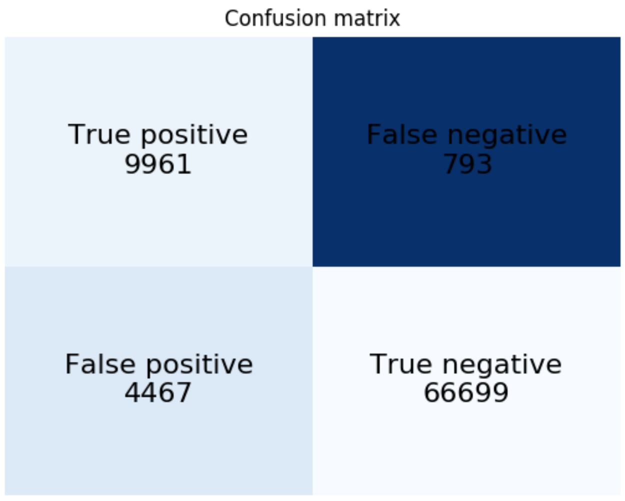

Figure 18: Confusion matrix on mouse proteome

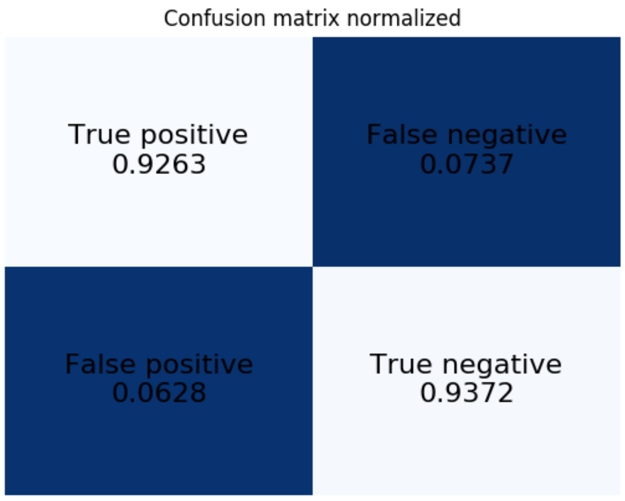

Figure 19: Normalized confusion matrix on mouse proteome

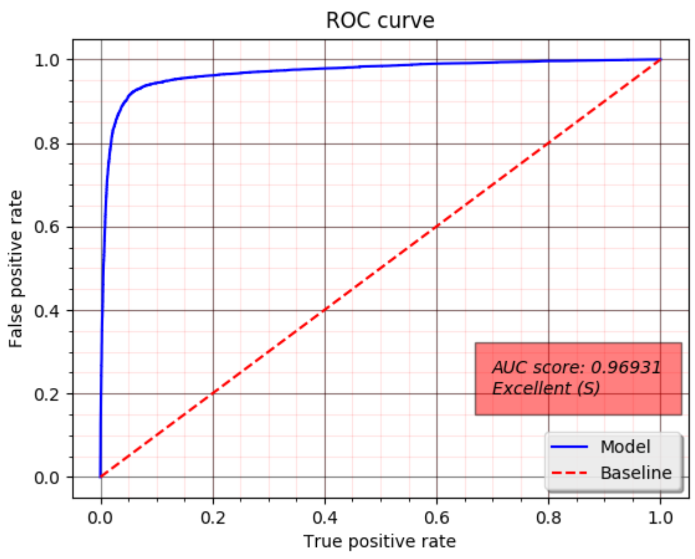

Figure 20: ROC curve on mouse proteome

Statistical value	value
Accuracy	0.9358
Precision	0.6904
Sensitivity	0.9263
Specificity	0.9372

Table 7: Statistical values: Mouse

4 Conclusion

Why are the precisions low for TM proteins and the proteomes compared to the training data set and non TM proteins?

One reason could be that there are very few positive samples in the data sets compared to the amount of negative samples. Precision is calculated with the equation:

$$precision = \frac{TP}{FP + TP} \tag{1}$$

The equation shows that a high amount of false positives (negative data predicted as positive) will decrease the precision, while in table 2 it can be seen the amount of positives are much lower than the amount of negatives for the data sets, about 5.5 times more negative samples than positive for TM proteins.

Another reason for the discrepancy could be that the training data mostly contained non TM proteins and that it is bad at predicting TM proteins. It might also be that a majority of the proteomes are TM proteins.

How does the model compare to other models on the web?

Comparing the model to PrediSi and SignalP, the created model is superior in accuracy for positive eukarytic proteins, while the created model has an inferior accuracy for negative samples as shown in table 8. However, you have to take into account that PrediSi and SignalP also tries to predict cleavage sites and other regions in proteins, while the created model simply predict if the proteins contain signal peptides or not.

Website	Positive accuracy	Control accuracy
PrediSi	72.66	98.30
SignalP (NN)	82.11	99.21
SignalP (HMM)	78.73	97.74

Table 8: Other models and accuracies for eukarytoic proteins [11]

In conclusion; The neural network is a good prediction model that has shown potential. Improvements can be made, such as higher quality data for training, a larger architecture and longer training times. The model could potentially be used for other applications, such as predicting exons in DNA or other areas, such as text or speech recognition.

All the source code can be found on github:
`https://github.com/TimothyBergstrom/DD2404-project`

References

[1] Henrik Nielsen, *SignalP V1.1 World Wide Web Prediction Server* http://www.cbs.dtu.dk/services/SignalP-1.1/sp_lengths.html

[2] Wikipedia, *Signal peptides wikipedia* https://en.wikipedia.org/wiki/Signal_peptide

[3] The UniProt Consortium UniProt: the universal protein knowledgebase Nucleic Acids Res. 45: D158-D169 (2017), *Swissprot database* http://www.uniprot.org/uniprot/?query=reviewed:yes

[4] Crooks GE, Hon G, Chandonia JM, Brenner SE WebLogo: A sequence logo generator, Genome Research, 14:1188-1190, (2004) Schneider TD, Stephens RM. 1990.*Sequence Logos: A New Way to Display Consensus Sequences* Nucleic Acids Res. 18:6097-6100

[5] Karpathy, *The Unreasonable Effectiveness of Recurrent Neural Networks* May 21, 2015 http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[6] Ankita Sharma, Yokila Arora, *Sequential LSTM-based Encoder for NLI* https://web.stanford.edu/class/cs224n/reports/2760320.pdf

[7] Zhou Yu et al, *USING BIDIRECTIONAL LSTM RECURRENT NEURAL NETWORKS TO LEARN HIGH-LEVEL ABSTRACTIONS OF SEQUENTIAL FEATURES FOR AUTOMATED SCORING OF NON-NATIVE SPONTANEOUS SPEECH* http://www.cs.cmu.edu/afs/cs/user/zhouyu/www/ASRU.pdf

[8] Henrik Nielsen, *Predicting Secretory Proteins with SignalP*, In Kihara, D (ed): Protein Function Prediction (Methods in Molecular Biology vol. 1611), pp. 59-73, Springer 2017, doi: 10.1007/978-1-4939-7015-5_6, PMID: 28451972, http://www.cbs.dtu.dk/services/SignalP/

[9] Karl Frank; Manfred J. Sippl, *High Performance Signal Peptide Prediction Based on Sequence Alignment Techniques*, Bioinformatics, 24, pp. 2172-2176 (2008), http://sigpep.services.came.sbg.ac.at/signalblast.html

[10] Karsten Hiller, *PrediSi*, Institute for Microbiology Technical University of Braunschweig, http://www.predisi.de/

[11] Karsten Hiller Andreas Grote, Maurice Scheer, Richard Münch, Dieter Jahn, *PrediSi: prediction of signal peptides and their cleavage positions* Nucleic Acids Research, Volume 32, Issue suppl_2, 1 July 2004, Pages W375–W379 Published: 01 July 2004

https://academic.oup.com/nar/article/32/suppl_2/W375/1040487

5 Appendix

5.1 Model details

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_1 (Bidirection (None, 30, 128)           44032
_________________________________________________________________
dropout_1 (Dropout)          (None, 30, 128)           0
_________________________________________________________________
bidirectional_2 (Bidirection (None, 128)               98816
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 142,977
Trainable params: 142,977
Non-trainable params: 0
_________________________________________________________________
Input shape: (None, 30, 21)
Output shape: (None, 1)
```

Train set: 362 844 positive samples and 363 615 negative samples
Validation set: 91 098 positive samples and 90 517 negative samples
Test set: 113 604 positive samples and 113 414 negative samples

12 hours trained with on a GTX 970. Batch size of 4096, dropout of 0.5 for each layer and using ADAM optimizer with default learning rate 0.0001.

The input data was a batch of (30, 21) arrays, corresponding to 30 amino acid long sequences with 21 possible amino acids (20 standard amino acids + unknown acid X).

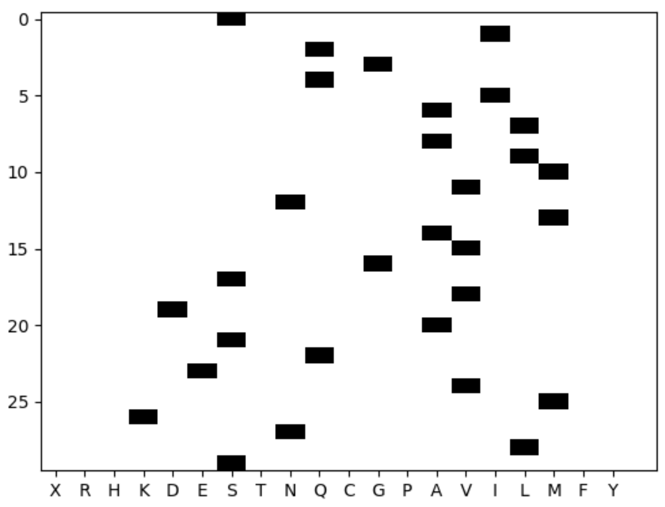

Figure 21: Visualization of sample after one-hot encoding

5.2 Pitfalls when training a model

Examples of overfitting and other problems that you will encounter when training a neural network.

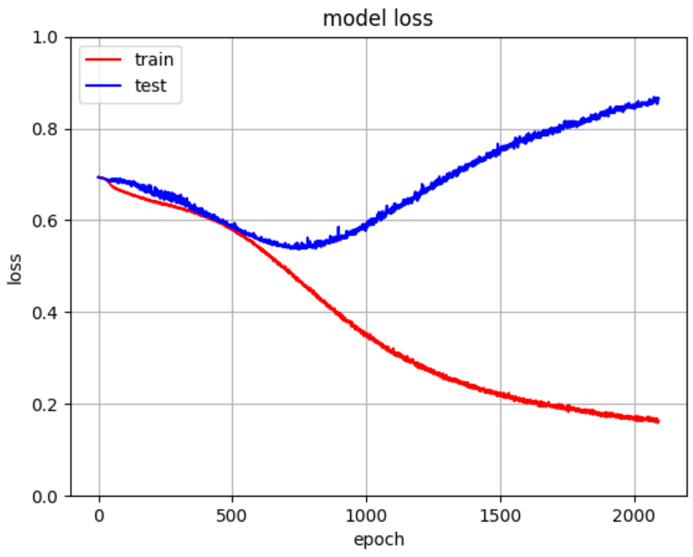

Figure 22: Overfitted

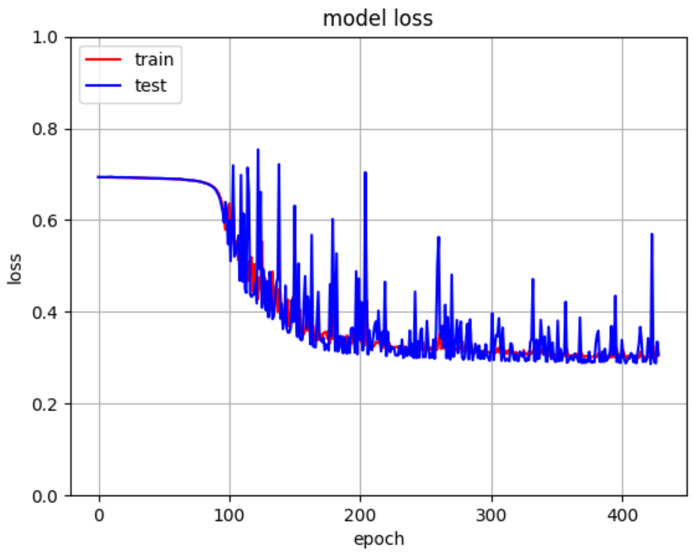

Figure 23: Too high learning rate

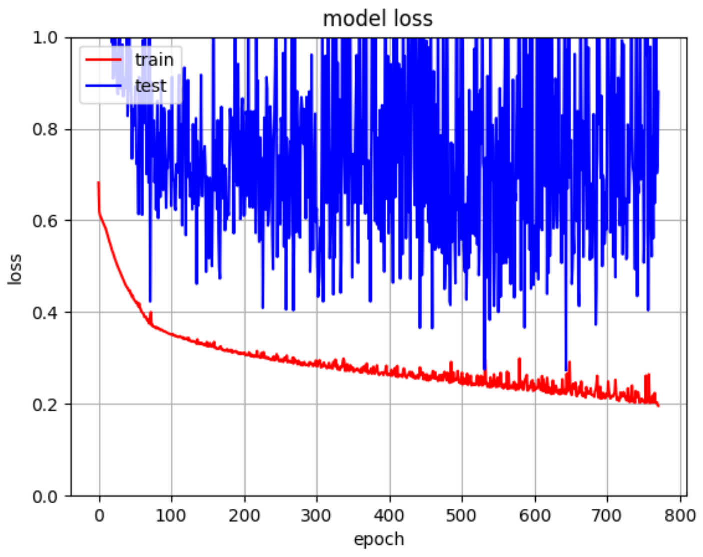

Figure 24: Data not shuffled