Sentiment Analysis with Long Short-Term Memory networks

Research Paper Business Analytics
Vrije Universiteit Amsterdam

Author: Fenna Miedema
Supervisor: Prof. dr. Sandjai Bhulai

August 1, 2018

Abstract

Sentiment classification is a task where text documents are classified according to their sentiment. Recurrent Neural Networks and Long Short-Term networks are both models that are often used for sentiment analysis. The goal of this research is to find out why these models work well for sentiment analysis and how these models work.

Recurrent Neural Networks and Long Short-Term networks introduce a memory into the model. Having a memory in a network is useful because, when dealing with sequenced data such as text, the meaning of a word depends on the context of the previous text. A shortcoming of the Recurrent Neural network is that it is only capable of dealing with short-term dependencies. Long Short-Term networks address this problem by introducing a long-term memory into the network.

A Long Short-Term model is build to classify the sentiment of a movie review dataset. The model correctly classified 86.74% of the reviews in the validation set. The model is very sensitive to overfitting but gives quite a good result even without parameter tuning.

Contents

1 Introduction

The area of computer science and artificial intelligence concerned with the interaction between computers and natural language is called Natural Language Processing (NLP). The natural language contains large diverse vocabularies, words with several different meanings and speakers with different accents. Human interaction and mutual understanding are therefore complicated. For the most part, humans can easily deal with these kinds of problems and skillful use of language is a major part of what makes us human. For this reason, designing a machine that is capable of understanding, speaking and responding properly to natural language has intrigued engineers and scientists for centuries.

In [1] the goal of NLP is described as: "to accomplish human-like processing of natural language". Part of an NLP system could be:

- Paraphrase an input text,
- Translate the text into another language,
- Answer questions about the contents of the text,
- Draw inferences from the text.

There are two distinct focuses made in NLP, language processing and language generation. Language processing refers to analysis of language for the purpose of producing a meaningful interpretation, language generation refers to the task of generating natural language. Another distinction made in NLP is between analysis of spoken language and written language. The emphasis in this paper will be on the processing of language in the written form, especially sentiment analysis.

Sentiment analysis is a subfield in NLP where the goal is to determine the attitude of a speaker or writer. Document-level sentiment classification is a functional task in sentiment analysis, and is crucial to understand user-generated content in social networks or product reviews. The task calls for identifying the overall sentiment polarity of a document.

Models that are often used for sentiment analysis are Recurrent Neural networks and Long Short-Term networks. The goal of this research is to find out why these models work well for sentiment analysis and how these models work.

First, a literature study is performed, the results of this study can be found in Chapter 2. Next, a detailed description of both Recurrent Neural and Long Short-Term Memory networks is given in Chapter 3. Chapter 4 describes a sentiment analysis dataset and in Chapter 5 a Long Short-Term Memory network is built to classify the sentiment of this dataset. In this chapter also the results of the model will be discussed. Finally, in Chapter 6 the conclusions are given.

2 Related Literature

An important part of information-gathering is to find out what other people think. [2] states that due to growing availability of opinion resources such as online review sites and personal blogs, new opportunities and challenges arise when trying to seek out and understand opinions of others. Lots of people do online research on a product, hotel or restaurant, before buying. The interest that individual users show in online opinions about products or services, and the potential influence of such opinions, is something that sellers of these items are paying more and more attention to. This is one of the reasons why sentiment analysis is an important task.

In [3] it is described that when analyzing a text, the meaning of a word in isolation cannot be given, but only the context of a sentence. This concept is captured in the 'principle of compositionality', also known as 'Frege's principle'. The principle of compositionality states that the meaning of a longer expression (e.g., a word in a sentence) depends on the meaning of its predecessors. In [4] it is stated that Recurrent Neural Networks (RNN) are capable of dealing with short-term dependencies in a sequence of data. But RNNs have trouble when dealing with long-term dependencies. These long-term dependencies have a great influence on the meaning and overall polarity of a document. Long Short-Term memory networks (LSTM) address this long-term dependency problem by introducing a memory into the network. In Chapter 3 a description of both RNN and LSTM is given. Next, two articles are discussed in which sentiment classification is performed with both RNN and LSTM. The performance of these models is compared to other more standard models.

Twitter is a rich resource for opinions of various kinds of events and products. [5] describes that detecting the sentiment of these micro blogs is a challenging task that has attracted increased research interest in recent years. The paper states that the traditional RNNs are not powerful enough to deal with complex sentiment expressions, therefore an LSTM network is implemented for classifying the sentiment of tweets. An experiment is conducted on the Twitter Sentiment corpus, a dataset containing 800,000 positive labeled tweets and 800,000 negative labeled tweets. The results show that the LSTM network outperforms all other classifiers including RNN, Support Vector Machine and Naive Bayes.

Another sentiment classification experiment on four large datasets is presented in [6]. The first three datasets are restaurant reviews from Yelp from the years 2013, 2014 and 2015. Each of these restaurant reviews is classified with a 1-5 star rating. The second dataset is the IMDB dataset, which contains movie reviews that are classified as being positive or negative. In the experiment, an LSTM model is compared to other models, like a Support Vector Machine. Just like in the previous article, the LSTM model yields the best performance on all four datasets.

The literature study shows that the LSTM network is a very powerful classifier for sentiment analysis because it makes use of a memory in the network. Having a memory in the network is useful because when dealing with sequenced data such as a text, the meaning of a word depends on the context of the previous text.

3 Models

In the previous section it is explained why RNN and LSTM work well for sentiment analysis. In this section a description of these two models is given. Both models are an extension of the standard neural network. The LSTM network is built upon the RNN, so the RNN will be discussed first.

3.1 Recurrent Neural Networks

RNNs are neural networks with loops in them, allowing for information to persist. Figure 1 shows how this recurrent connection is introduced into an RNN, allowing the network to have a memory. An RNN can be interpreted as multiple copies of the same network, where in each time step information is passed through. Figure 1 also shows this enrollment of an RNN, the gray box indicates a layer with a sigmoid activation function. The sigmoid activation function is given in Equation (1). In the figure, the input consists of a sequence of length T. At each time step t, x_t is entered to the model and an outcome h_t is computed, this output is given back to the model as an input for the next time step. The formula corresponding to the RNN model is given in Equation (2). In this equation, W and U are weights to regulate the input and b is the bias.

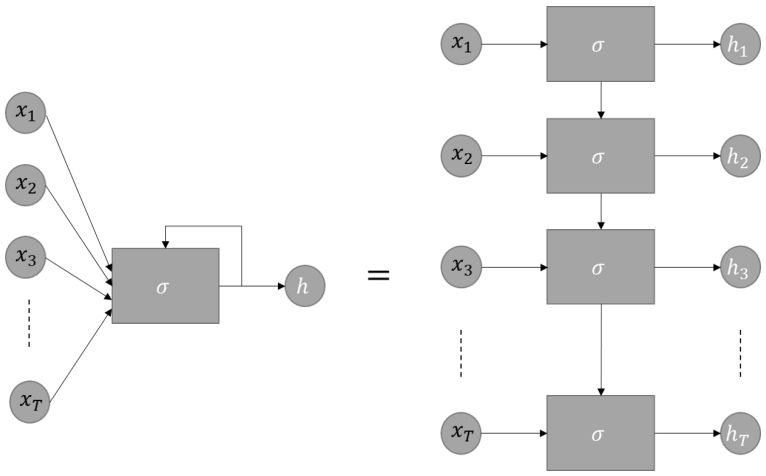

Figure 1: Recurrent neural network enrolled

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

$$h_t = \sigma(W \times x_t + U \times h_{t-1} + b) \tag{2}$$

One shortcoming of the RNN is that it is only able to use the previous context, as described in [7]. But in natural language, text often has a long-term dependency. For example when a model is trying to predict the next word in the following text:

"I live in the Netherlands. .... I speak fluent (...).".

The next word in this text should obviously be "Dutch". Recent information suggests that the next word is probably the name of a language, but if we want to know which language, we have to look back earlier in the text. Long Short-Term Memory networks address this issue of dealing with long-term dependencies.

3.2 Long Short-Term Memory

Instead of only having a short-term memory, LSTM networks also have a long-term memory and thus are capable of handling long-term dependencies. In [8] it is described how LSTMs are used for processing sequenced data by using gate vectors at each position to control the passing of information along the sequence. At each time step t there is a set of vectors, including an input gate i_t, a forget gate f_t, an output gate o_t and a memory cell C_t. All these together are used to compute the output of the hidden layer h_t as follows:

$$f_t = \sigma(W_f \times x_t + U_f \times h_{t-1} + b_f), \tag{3}$$

$$i_t = \sigma(W_i \times x_t + U_i \times h_{t-1} + b_i), \tag{4}$$

$$\tilde{C}_t = tanh(W_C \times x_t + U_C \times h_{t-1} + b_C), \tag{5}$$

$$C_t = i_t \times \tilde{C}_t + f_t \times C_{t-1}, \tag{6}$$

$$o_t = \sigma(W_o \times x_t + U_o \times h_{t-1} + b_o), \tag{7}$$

$$h_t = o_t \times tanh(C_t). \tag{8}$$

In this model, σ is the sigmoid activation function, $tanh$ the hyperbolic tangent activation function, x_t the input at time t, W_i, W_C, W_f, W_o, U_i, U_C, U_f, U_o are weight matrices to regulate the input and b_i, b_C, b_f, b_o are bias vectors.

In [9] a step by step description of the LSTM network is given, instead of having just one single network layer like the RNN, the LSTM has four interacting network layers. The structure of an LSTM network is described in Figure 2. In this figure, the gray boxes represent a neural network layer and the blue circles are pointwise operations like vector addition. The horizontal line running through the right of the diagram is the cell state. This cell state resembles a conveyor belt, it runs down the entire chain with only minor linear interactions. The cell state can also be interpreted as the memory of the network and has the ability to remove or add information by structures called gates. Figure 3

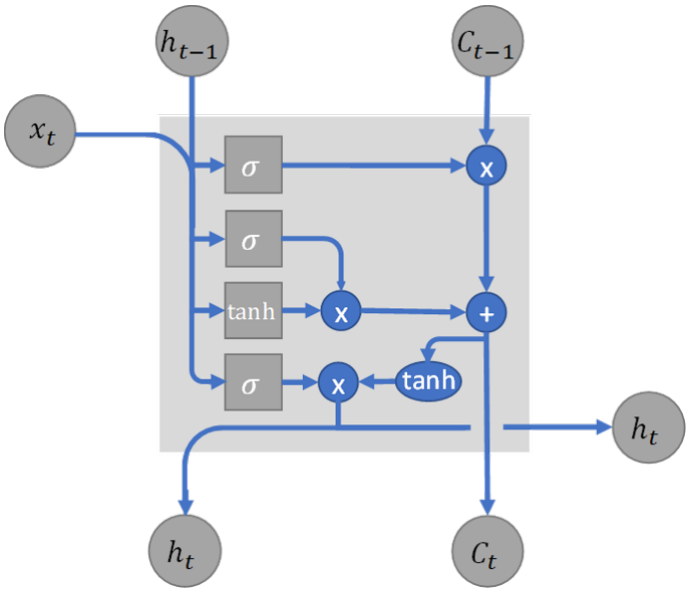

Figure 2: LSTM Network

shows the four steps of the LSTM network, step A, B, C and B. Each step of the LSTM network will be discussed.

- **Step A**
  First, the model needs to determine what to throw away from the cell state, this part of the network is shown in Figure 3(A) and Equation (3). This is referred to as the forget gate values f_t. The input in this step is the output of the previous step h_{t-1} and the input x_t. A sigmoid activation function is used to give output values between 0 and 1, where 0 corresponds to "let nothing through" and 1 to "remembering everything".

- **Step B**
  The next step is to determine what information is going to be added to cell state, shown in Figure 3(B) and Equations (4) and (5). In this step again the inputs are h_{t-1} and x_t. The input layer gate i_t first applies a sigmoid layer over the input to determine which parts of the cell state will be updated. Then a *tanh* layer is used to create new candidate values $\tilde{C}_t$. In the next step, these two will be combined to update the cell state C_t.

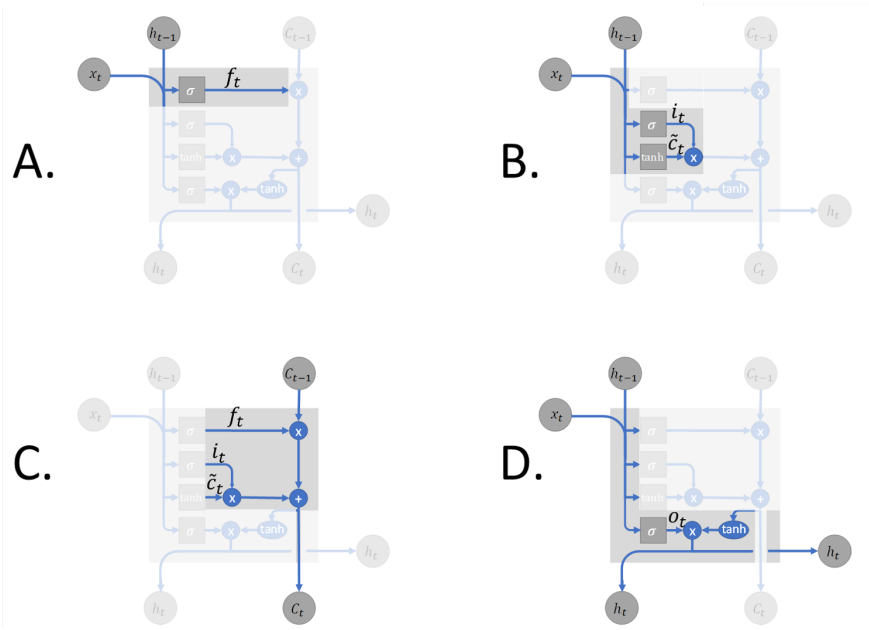

Figure 3: LSTM Network

- **Step C**
  Now the old cell state is multiplied by f_t, to forget the things that are not needed anymore and the new information is added to the cell state memory. This part of the network is shown in Figure 3(C) and Equation (6).

- **Step D**
  In the final step, it is determined what the output h_t is, shown in Figure 3(D) and Equations (7) and (8). This output is based on the cell state but in a filtered version. First, a sigmoid layer is applied to the previous output h_{t-1} and input x_t, to determine the output gate values o_t. This is a value between 0 and 1 indicating which parts of the cell state are going to be output. Then the cell state C_t is transformed by a *tanh* function to get values between -1 and 1. These transformed cell state values are then multiplied by the output gate values o_t to end up with the output h_t. This output will be printed and pushed through to the next step of the network.

4 Dataset

To illustrate how an LSTM network can be used for sentiment classification, the IMDB dataset [10] is used. The IMDB dataset is a set of 50,000 reviews from the Internet Movie Database. Each of these movie reviews is classified as either being "positive" or "negative". The data is already divided into a train and validation set, each with 25,000 reviews and 50% positive and 50% negative reviews. From all the movie reviews only the top 10,000 most frequently occurring words are used. This is to make sure that rare words are discarded and the training data contains vectors of a manageable size. The original ratings on the IMDB dataset are 1-10 star ratings, these are linearly mapped to [0,1] to use document labels when training the model. A "1" indicates a positive review and "0" a negative review.

The dataset has already partly been processed, the reviews have been transformed into sequences of integers, where each word stands for a specific word in a dictionary. For example, the first 10 words of the first movie review in the train set are:

"1 14 22 16 43 530 973 1622 1385 65 ...".

index	**word**
1	[start sequence]
2	[unknown]
3	the
4	and
5	a
6	of

Table 1: Word index matrix IMDB movie review dataset

The movie reviews can be transformed back into text using the word index matrix corresponding to the IMDB dataset. In the matrix, all the words and corresponding indexes are stored. Table 1 shows a part of the word index matrix. A "1" indicates the start of a sequence and "2" indicates an unknown word. The table is ordered according to the frequency of a word, the word that occurred the most frequent in the reviews is "the". The text corresponding to the first movie review is shown below, the sentiment of this review is classified as "positive". The text is not completely correct anymore, no correct sentences are formed, but the main message of the review can still be interpreted. As the transformed text shows, there are quite some unknown words in the review. As rare words already have been excluded from the text, this large number can be

explained by words that are misspelled.

"[start sequence] this film was just brilliant casting location scenery story direction everyone's really suited the part they played and you could just imagine being there robert [unknown] is an amazing actor and now the same being director [unknown] father came from the same scottish island as myself so i loved the fact there was a real connection with this film the witty remarks throughout the film were great it was just brilliant so much that i bought the film as soon as it was released for [unknown] and would recommend it to everyone to watch and the fly fishing was amazing really cried at the end it was so sad and you know what they say if you cry at a film it must have been good and this definitely was also [unknown] to the two little boy's that played the [unknown] of norman and paul they were just brilliant children are often left out of the [unknown] list i think because the stars that play them all grown up are such a big profile for the whole film but these children are amazing and should be praised for what they have done don't you think the whole story was so lovely because it was true and was someone's life after all that was shared with us all"

Not all reviews have the same length, the shortest review contains only 11 words and the longest review contains 2,494 words. When building the model all the reviews need to be of the same length, a length of 500 words is used and zeros are added to the review when it is shorter than 500 words. The train and validation set now each consist of 25,000 reviews with every review a length of 500 words.

The input for an LSTM network is 3-dimensional, the samples, time steps and features. There will be 500 time steps in the model, where each word of the review is being fed to model at each time step. At this point, there is only one feature, the word. A popular method for feature learning in NLP is word embedding. This technique encodes words as real-valued vectors in a high dimensional space, where words with a similar meaning are close in the vector space. This way the model learns itself what the best feature representation is. To include this word embedding into the model, an embedding layer needs to be added to the model. In this layer, each word is transformed into a real-valued vector. In the next chapter it is described how the embedding layer is added to the LSTM network and the results of the network are discussed.

5 Results

The model built to classify the sentiment of the movie reviews, consists of 4 layers. These layers are:

1. Input layer (length 1),
2. Embedding layer (output of length 32),
3. LSTM layer (100 neurons),
4. output layer (1 neuron).

An overview of the model is shown in Figure 4. The input to the model is a text of 500 integers where each integer represents a word. So there are 500 time steps, at each time step one word is given to model. The word is entered into the embedding layer with one neuron, in this layer the word is transformed into a real-valued vector of length 32. This way 32 features are created. Next an LSTM layer with 100 neurons is added to the network, each of the features is multiplied by a weight for each LSTM cell, where each LSTM cell contains four gates. Next to the 32 features, the output of the previous time step is also used as an input for the LSTM cells. The final layer is the output layer with one neuron. Here the weighted sum of the 100 outputs of the LSTM layer is taken and a sigmoid activation is used to make 0 or 1 predictions.

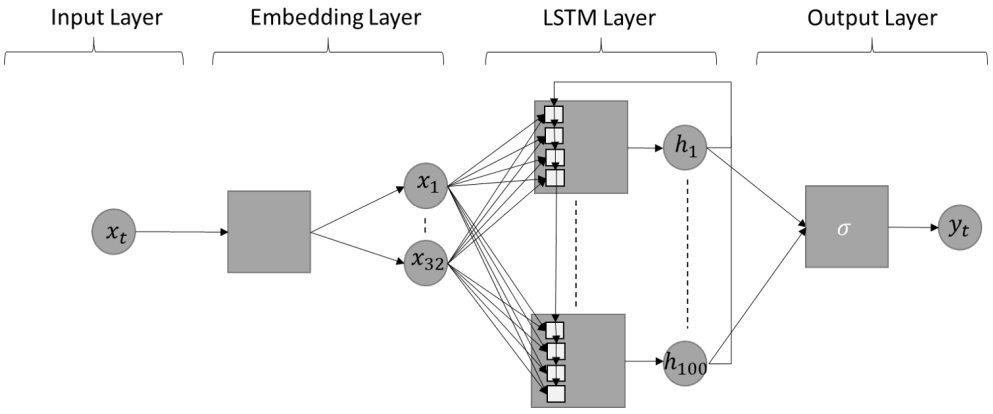

Figure 4: LSTM Network for IMDB sentiment classification

There is a very large number of parameters the model trains:

- the embedding parameters (10,000*32 parameters),
- the LSTM parameters:
  - W_i, W_f, W_C, W_o for each of the 100 LSTM neurons (32*4*100 parameters),
  - U_i, U_f, U_C, U_o for each of the 100 LSTM neurons (4*100*100 parameters),
  - b_i, b_f, b_C, b_o bias vectors for each of the 100 LSTM neurons (4*100 parameters).
- the output layer parameters, 100 weights and 1 bias (101 parameters).

In total there are 373,301 parameters for the model to be trained. This is a binary classification problem so the model is trained with the binary log loss function and 15 epochs are used. The binary log loss function is given in Equation (9), where N is the total number of samples, y_i the actual label and $\hat{y}_i$ the predicted label. The training time for the model is around 60 minutes.

$$-\frac{1}{N}\sum_{i=1}^{N}\Big[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\Big] \tag{9}$$

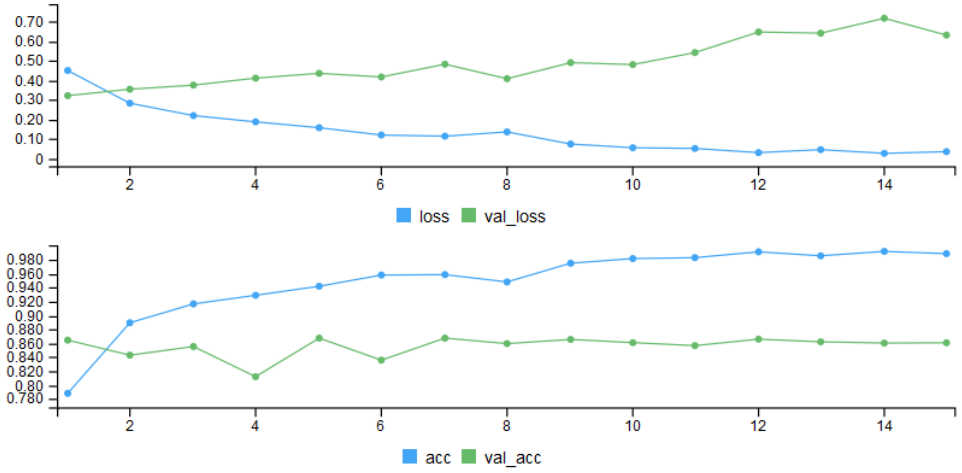

Figure 5: LSTM Network results on train and validation set

Figure 5 shows the loss and accuracy for each epoch on the train and validation set. The plot at the top shows the loss, the plot at the bottom the accuracy. Although the model is not trained on the accuracy, it keeps track of this metric in each epoch. The blue lines show the results on the train set, the green lines on the validation set. The loss on the train set decreases after each epoch, but the validation loss goes up. The accuracy on the validation set peaks in the fifth epoch, this is due to the fact that the model is overfitting the training data. The final model is stopped training after the fifth epoch and the loss and accuracy of the final model on the validation set are 0.4366 and 0.8675, respectively.

To further evaluate the performance of the model, the predictions are reviewed. Table 2 shows the predictions for review numbers 1 through 6 of the validation set. The output of the model is a number between 0 and 1 because the sigmoid activation function is used in the final layer of the model. This output is the likelihood the model gives to a review having a positive label. The predicted label can then be determined by rounding the output of the model. The model is quite confident for most of the reviews, 4 outcomes are very close to 0 or 1. The actual label of the review is also shown in the table. One out of these six reviews is wrongly classified by the model.

Review Number	**Prediction**	**Predicted label**	**Actual label**
1	0.01795	0	0
2	0.99997	1	1
3	0.65761	1	1
4	0.56650	1	0
5	0.99997	1	1
6	0.99971	1	1

Table 2: Predictions validation set

This is a large model with quite a long training time (around 60 minutes), there are lots of weights for the model to train. The results of the model on the validation set are pretty good, an accuracy of 86.75%. The parameters in the model have not yet been trained. Performing a grid search on the parameters, such as the number of neurons, could possibly lead to a better performance of the model. The performance of the model could possibly also be improved by selecting another range of words from the reviews, for example removing the top 50 most frequently occurring words. Words like "the" that occur very frequently in a text, are not good indicators of the sentiment. Optimizing the performance of the model is not part of the scope of this research, and thus is left out of this paper.

6 Conclusion & discussion

Sentiment classification is a task in NLP where documents are classified according to their sentiment. Models that are often used for sentiment analysis are RNN and LSTM. The goal of this research was to find out why these models work well for sentiment analysis and how these models work.

When analyzing a text, the meaning of a word in isolation cannot be given, but only the context of a sentence. This is captured in the principle of compositionality, which states that the meaning of a longer expression depends on the meaning of its predecessors. RNNs are neural networks with loops in them, allowing the network to have a memory. Having a memory in a network is useful because when dealing with sequenced data such as text, the meaning of a word depends on the context of the previous text. One shortcoming of the RNN is that it is only capable of dealing with short-term dependencies. LSTM is a network which addresses this issue by introducing a long-term memory in the network. In an LSTM network sequenced data is processed by using gate vectors at each position to control the passing of information along the sequence. The sets of vectors which control this passing of information are the input gate i_t, a forget gate f_t, an output gate o_t and a memory cell C_t. The literature study showed that the LSTM model resulted in the best performance on several sentiment classification tasks, compared to other classifiers.

To illustrate how an LSTM can be used for sentiment classification, a network is built to classify the sentiment of the IMDB dataset. This dataset contains reviews of 50,000 movies. Each movie is labeled with a 0 or a 1, indicating a negative or positive review. Only the top 10,000 most frequently occurring words are used to make sure rare words are discarded and the maximal length of a review has been set to 500 words.

The LSTM model built, consists of four layers: the input layer, the embedding layer, the LSTM layer and the output layer. Word embedding is a popular method for feature learning in NLP, this technique encodes words as real-valued vectors in a high dimensional space. The model has been trained on the train set and evaluated on the validation set. The model is very sensitive to overfitting, so the model was stopped training after the fifth epoch. The final model resulted in a loss and accuracy on the validation set of 0.4366 and 0.8674, respectively. This is already a pretty good result. The parameters in the model have not yet been trained, performing a grid search on the parameters could possibly lead to a better performance of the model.

References

[1] Elizabeth D. Liddy, *Natural Language Processing* , Syracuse University (School of Information Studies), 2001.

[2] Bo Pang and Lillian Lee, *Opinion Mining and Sentiment Analysis, Foundations and Trends in Information Retrieval*, 2008.

[3] Theo M.V. Janssen, *Frege, contextuality and compositionality*, Computer Science, University of Amsterdam, 2002.

[4] Tsungnan Lin, Bill G. Horne, Peter Tino, and C. Lee Giles, *Learning Long-Term Dependencies in NARX Recurrent Neural Networks*, Transactions on neural networks, 1996.

[5] Xin Wang, Yuanchao Liu, Chengjie Sun, Baoxun Wang and Xiaolong Wang, *Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory*, Harbin Institute of Technology, Harbin, China, 2015.

[6] Duyu Tang, Bing Qin, Ting Liu, *Document Modeling with Gated Recurrent Neural Network for Sentiment Classification*, Harbin Institute of Technology, Harbin, China, 2015.

[7] Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan Cernock and Sanjeev Khudanpur, *Recurrent neural network based language model*, Johns Hopkins University, USA, 2010.

[8] Shuohang Wang and Jing Jiang, *Learning Natural Language Inference with LSTM*, Singapore Management University, 2016.

[9] Christopher Olah, *Understanding LSTM Networks*, 2015.

[10] Maas, Andrew L. and Daly, Raymond E. and Pham, Peter T. and Huang, Dan and Ng, Andrew Y. and Potts, Christopher, *Learning Word Vectors for Sentiment Analysis*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 2011.