Convolutional Neural Networks vs Recurrent Neural Networks for Natural Language Processing

Selina Mead Miller
Tübingen Universität
selinamead@live.com
Matrikelnummer: 4065083

The main goal of this work was to develop a solid understanding of the fundamental make-up and intricate workings of deep neural networks by experimenting with different architectures and models and comparing their performance in natural language processing.

1 Introduction

In recent years, deep learning architectures have revolutionised the way scientists and linguists perform Natural Language Processing tasks (NLP). Deep learning approaches have achieved very high performance across many NLP tasks. The two main deep learning architectures are Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). Vanilla RNN experienced difficulties when used on NLP tasks due to the vanishing gradient problem. The vanishing gradient problem of basic RNN is outside the scope of this paper, however the resulting gated mechanisms which were designed to solve the vanishing gradient will be discussed. The Gated Recurrent Units (GRU) developed by Cho et al. in 2014 [1] and Long-Short-Term-Memory (LSTM) created by Hochreiter and Schmidhuber in 1997 [2] form the basis of the RNN architectures used in NLP tasks.

CNN are designed primarily for spatial data such as images, whilst RNN are primarily designed for sequential data such as text. Intuitively, when choosing a deep learning model for any NLP task one would naturally gravitate towards RNN due to the sequential aspect, however CNN have been very successful in many NLP tasks. So how to know which Deep Neural Network (DNN) to use for which NLP task. There is no definitive answer to this question but it is interesting to compare the differing models and how they interact differently with the data. This work compares the CNN and RNN models in several different NLP tasks. The architectures are pitted against one another systematically on the following NLP tasks; Sentiment Analysis, POS-Tagging and Machine Translation. One key finding of these experiments is that while CNN appear to be disadvantaged in the field of textual, sequential data compared to the RNN, the performance of the CNN models shows how powerful they are at capturing patterns in textual data.

2 Related Work

Yin et al (2017) tests 4 differing neural network architectures on 7 different NLP tasks to provide a detailed comparison. In the paper the authors use a CNN, GRU, LSTM and Bi-LSTM on the following tasks: Sentiment Classification, Relation Classification, Textual Entailment, Answer Selection, Question Relation Match, Path Query Answering and Part of Speech tagging. The GRU model performed the greatest on 4 out of 7 tasks while CNN

had the highest accuracy scores for 2 tasks and Bi-LSTM recording the highest for the remaining task. The authors aim to give basic guidance for DNN selection.

Vu et al. (2016) investigated CNN and RNN but without gated mechanisms for relation classification. The results showed greater performance by the CNN model compared to the RNN model.

Arkhipenko et al (2016) compare CNN, GRU and LSTM in a sentiment analysis task and results showed that the GRU outperforms LSTM and CNN.

3 NLP Tasks

The following natural language processing tasks were used to compare the CNN and RNN models.

3.1 Sentiment Analysis

Sentiment analysis is the interpretation and classification of emotions (positive, negative and neutral) within text data using text analysis techniques. The three main types of algorithms used are; rule based, automatic and a hybrid of the two. In this paper the analysis was conducted using an automatic algorithm by implementing a machine learning approach. Typically the machine learning algorithms were classification models such as Naive Bayes, Linear regression and Support Vector Machines, however with the popularity of neural networks in recent time sentiment analysis has been performed with excellent results using Neural Networks. While CNN models are typically used in computer vision tasks, CNN has been shown to perform very well on sentiment Analysis tasks. RNN and LSTM models are the popular choice for most NLP tasks as they are recurrent so they are ideal for sequential textual data.

In this experiment we tested the different architectures on movie reviews using the IMBD dataset which contains 50,000 labelled reviews.

3.2 POS-Tagging

Part of Speech tagging is the classification of words of a sentence into a list of tuples (where each tuple has the form (word, tag)). The tag signifies whether the word is a noun, adjective, verb, etc. Typically the Hidden Markov Model has been popular for the task however as with all NLP tasks, Neural networks have demonstrated excellent results.

Most of the POS tagging falls under Rule Base POS tagging, Stochastic POS tagging and Transformation based tagging. In this task rule based tagging was implemented using the NLTK Stanford treebank corpus. With a training set containing 2500 samples and 600 for validation.

3.3 Machine Translation

Machine translation is typically one of the more difficult NLP tasks but computer linguists have achieved excellent results using deep learning in recent times. Neural translation models require far less memory than the traditional statistical machine translation (SMT) models. Typically RNN models have been more popular in performing Neural machine translation (NMT) tasks. They have achieved state of the art performance and have taken a dominant position in NMT [6]. According to Yang et al, CNN models have recently been used but for a long while, most of these models don't compete performance wise with RNN based model, especially since the arrival of Attention Mechanism [6].

For this task an English to German dataset containing 152,820 pairs of English to German phrases was used.

4 Deep Learning Models

4.1 CNN

Convolutional Neural Networks (CNN) are traditionally used for computer vision tasks, however recently they have been used with success in Natural Language Processing (NLP) tasks. CNN architecture is perfectly designed to detect features and capture patterns in the features/data, which is why they are at the core of most computer vision systems currently in operation. When it comes to tasks dependant on long semantics CNN's will be found wanting compared to recurrent architectures. Unlike RNN, CNN are not designed for sequential text and intuitively seem an unnatural fit to many NLP tasks. This does not mean that they haven't shown good results in certain tasks

CNN	RNN
Traditionally suitable for spatial data such as images.	RNN is suitable for sequential data.
CNN is considered to be more powerful than RNN.	RNN includes less feature compatibility when compared to CNN.
This network takes fixed size inputs and generates fixed size outputs.	RNN can handle arbitrary input/output lengths.
CNN is a type of feed-forward artificial neural network with variations of multilayer perceptrons designed to use minimal amounts of preprocessing.	RNN unlike feed forward neural networks - can use their internal memory to process arbitrary sequences of inputs.
CNNs use connectivity pattern between the neurons. This is inspired by the organization of the animal visual cortex, whose individual neurons are arranged in such a way that they respond to overlapping regions tiling the visual field.	Recurrent neural networks use time-series information - what a user spoke last will impact what he/she will speak next.

Table 1: *Comparison of CNN and RNN deep learning architectures: http://cs231n.stanford.edu*

such as sentiment analysis where feature capturing is more important than semantic analysis. The main advantage of CNN over RNN in NLP is the speed at which they are able to produce results, they are not fully connected and are therefore far less computational expensive than fully connected networks. CNN are able to process data extremely fast, whereas RNN can be slow and fickle even though they are a more natural approach to language processing [8].

A convolutional neural network is designed to identify indicative local predictors in a large structure, and combine them to produce a fixed size vector representation of the structure, capturing these local aspects that are most informative for the prediction task at hand. [8]

How the convolution and pooling architecture for NLP tasks works is by applying a filter over each instance of a n-word sliding window over the sentence. Then, the pooling operation takes the max value found in each of the different windows and combines them into a vector. This will locate the most important features. The vector is then fed further into the network that is used for prediction. The gradients that are propagated back from the network's loss during the training process are used to tune the parameters of the filter function to find the most important features for the task.

CNN architecture used in these experiments consist of an input layer, a convolutional layer and a max pooling layer, as shown in figure 1.

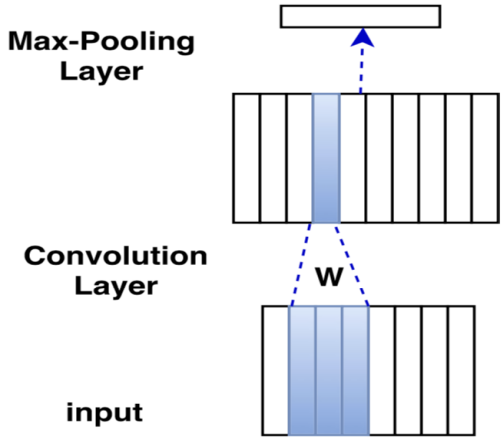


Fig. 1: CNN architecture

4.2 LSTM

Recurrent neural networks are intimately related to sequences and lists. They're the natural

architecture of neural network to use for such data. However, RNN are hard to train effectively because of the vanishing gradients problem. Gradients in later steps of the sequence diminish quickly in the back-propagation process, and do not reach earlier input signals, making it hard for RNN to capture long-range dependencies [7]. The Long Short-Term Memory (LSTM) architecture was designed with this in mind. The main idea behind the LSTM is to incorporate as part of the memory cells, a vector that can preserve gradients. Access to the memory cells is controlled by gates which comprise of functions that simulate logical gates. At each input state, a gate is used to decide how much of the new input should be written to the memory cell, and how much of the current content of the memory cell should be forgotten.

Figure 2 shows the common LSTM unit composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

While LSTM's are a great tool for sequence learning due to their ability to retain important information and to loose unimportant information, they are not necessarily always the best model to use as they do have some drawbacks in comparison with simpler deep learning architectures. Due to the high number of parameters LSTM take much longer to train and require more memory to train. They are also prone to over-fitting. Dropout is much harder to implement in LSTM compared to other RNN or CNN structures and they are sensitive to different random weight initialisation. When applying dropout to an RNN with an LSTM it is crucial to apply dropout only on the non-recurrent connection, i.e. only to apply it between layers and not between sequence positions [9].

4.3 Bi-Directional LSTM

Bi-directional LSTM follow the same architecture of a uni-directional LSTM but with the added benefit of the output layer receiving information from the past (backwards) and future (forwards) states. One would use Bi-LSTM when knowing the past and future information will improve the performance.

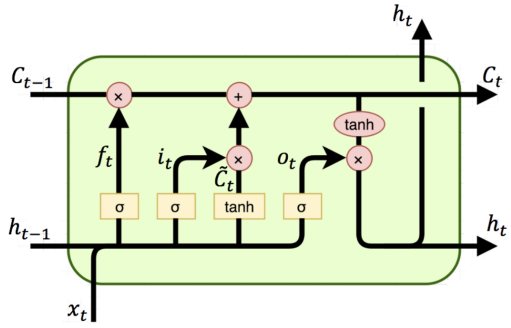


Fig. 2: LSTM architecture

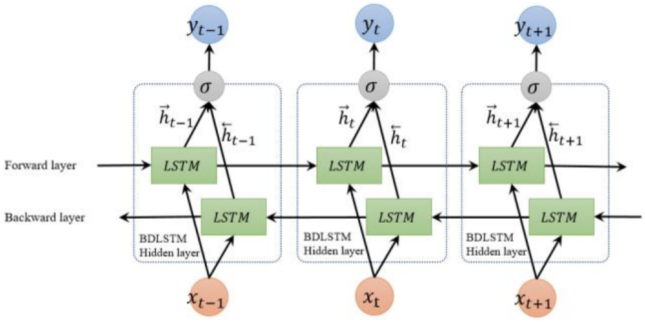


Fig. 3: Bi-LSTM architecture

4.4 GRU

The Gated Recurrent Network (GRU) is similar to an LSTM with a forget gate, but with fewer parameters than LSTM, as it lacks an output gate. While the LSTM models are very successful, they are also computationally expensive, whereas the GRU is generally faster to train due to the fewer parameters [8]. The GRU was introduced by Cho et al. as an alternative to the LSTM. Similar to the LSTM, the GRU is also based on a gating mechanism, but with fewer gates and without a separate memory component. One gate is used to control access to the previous state and compute a proposed update. The updated state is then determined based on an interpolation of the previous state and the proposal, where the proportions are controlled using the gate. The GRU was shown to be effective in language modelling and machine translation. However, the jury is still out between the GRU and the LSTM [8].

5 Experiments

In order to maintain a level of fairness regarding the model comparison between the four differ-

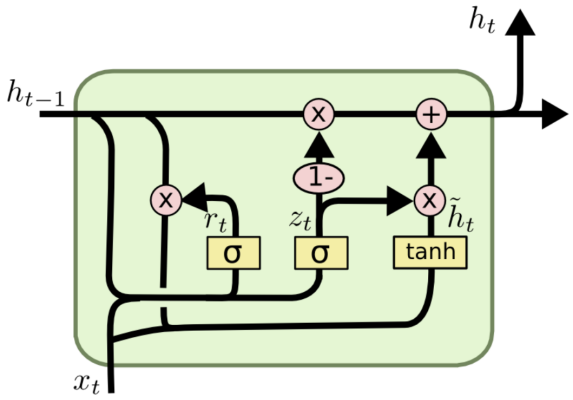


Fig. 4: GRU architecture

ent architectures the experiments followed a uniform setup. This was to ensure there were not significant advantages for any one model. These principles are derived from those used by Yin et al. (2017). All models follow these basic design principles: 1. Always train from scratch without the use of pre-trained word embeddings. 2. Training using the basic model for each architecture, i.e no batch normalisation. 3. Ensure all results for each model and for each task are found using optimal hyperparameters. 4. Use the basic architecture for each model: CNN architecture consists of an embedded layer, a convolutional layer, a global max pooling layer, a dropout layer and a dense layer. GRU and LSTM model the input from left to right and always use last hidden state as the final representation, they consist of embedded layer, LSTM/GRU layer, dropout layer and dense layer. The hyperparameters are tuned on number of epochs, learning rate and dropout.

6 Results

The results of the three NLP tasks along with the hyperparameters and the computational expense are shown on the following page.

The most natural fit for CNN seem to be classifications tasks, such as Sentiment Analysis. Convolutions and pooling operations lose information about the local order of words, so that sequence tagging such as POS Tagging and Machine Translation are more difficult to fit into a pure CNN architecture. It is therefore unsurprising the CNN performed the best in the Sentiment Analysis task, but was outperformed by the RNN models in the Machine Translation task.

6.1 Sentiment Analysis Task

The Sentiment Analysis task produced excellent results from the CNN model. It is a task perfectly suited to the CNN architecture and is an excellent choice for NLP tasks such as this. The GRU achieved satisfactory results while the two LSTM models were unable to produce results inline with the CNN model.

6.2 POS-Tagging Task

For the POS Tagging experiment, again the CNN model outperformed the RNN models. While it was not a surprising result in the Sentiment Analysis task, it is rather surprising considering this task is more heavily reliant on capturing information of a sequential nature. This task caused the GRU, LSTM and Bi-LSTM models to overfit, in order to determine whether this was a case of there being easier samples to predict in the test set or if it was a different problem with the model, k-fold cross validation was performed, the results after fold cross validation produced very similar results to the original output. In the case of both the CNN and RNN models, there appeared to be a significant difference between the model's prediction and the target value, compared to the training data, meaning there was a high variance in all models. This lead to the conclusion that the models were unable to capture relevant relations between features and target outputs - a sign of overfitting. The surest sign of the models overfitting was the poor results produced by the test set in each case, whilst the training set performed well.

The first method implemented to reduce overfitting was regularisation, L2 regularisation in particular. By adding an extra element to the loss function it punishes the model for being too complex or for using too high values in the weight matrix [8]. The next step was to add dropout and early stopping (optimum epoch). The optimal dropout rates and optimum epoch were found by hyperparameter tuning. This removed any significantly damaging overfitting and resulted in the best performance of the models.

Sentiment Analysis					
Model	Accuracy	Loss	Time	Epochs	Dropout
CNN	**94.11**	0.03	1:14 2ms/sample	10	0.2
LSTM	76.73	0.31	11:33 14ms/sample	5	0.3
Bi-LSTM	77.85	0.28	10:36 13ms/sample	5	0.3
GRU	86.54	0.34	9:00 11ms/sample	5	0.3

Table 2: Results for Sentiment Analysis task

POS-Tagging					
Model	Accuracy	Loss	Time	Epochs	Dropout
CNN	**99.18**	0.03	0:26 137us/sample	15	0.2
LSTM	91.73	0.31	7:06 3ms/sample	20	0.1
Bi-LSTM	91.87	0.28	6:46 3ms/sample	20	0.1
GRU	91.54	0.34	5:10 3ms/sample	20	0.3

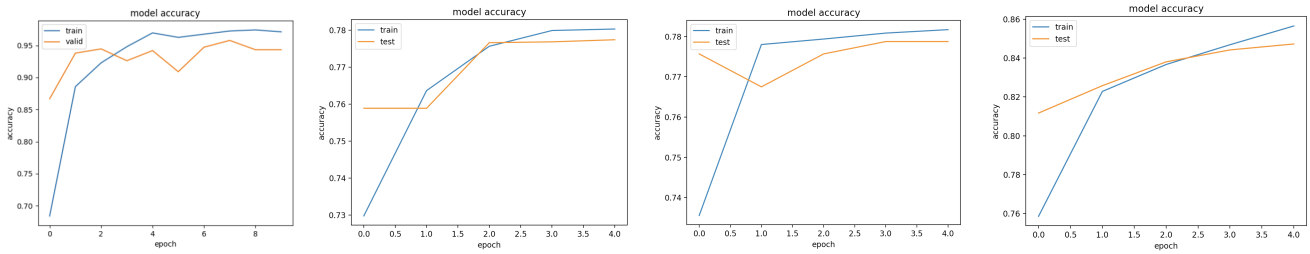
Table 3: Results for POS-Tagging task

Machine Translation					
Model	Accuracy	Loss	Time	Epochs	Dropout
CNN	76.33	0.53	0:42 472us/sample	20	0.2
LSTM	79.26	1.31	1:05 1ms/sample	15	0.2
Bi-LSTM	**85.82**	1.21	1:20 1ms/sample	20	0.2
GRU	81.18	0.78	0:32 1ms/sample	10	0.3

Table 4: Results for Machine Translation task

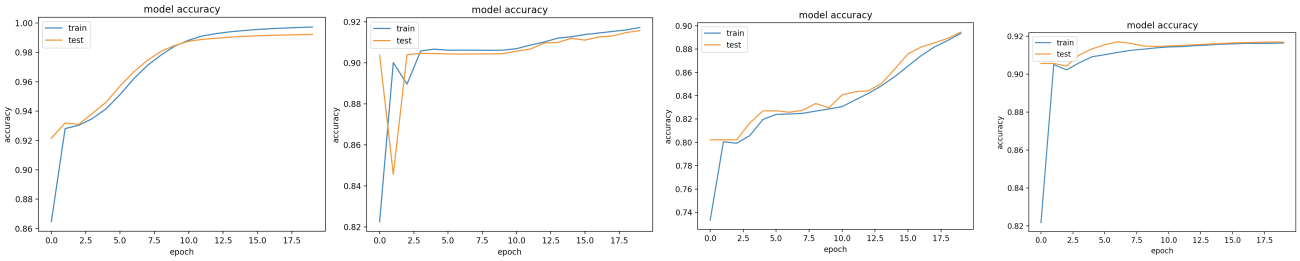
6.3 Machine Translation Task

The RNN models, in particular the Bi-LSTM model, outperformed the CNN model unsurprisingly on the machine translation task. It makes intuitive sense that this would be the case when dealing with a much more complex task compared with the previous tasks. It is interesting to note the computation expense from the three differing RNN when comparing the time taken for training on the POS tagging and sentiment analysis tasks.



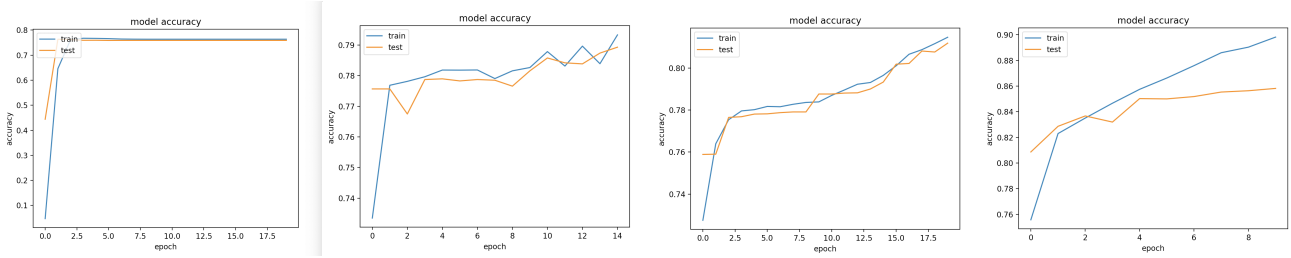
(a) CNN (b) LSTM (c) Bi-LSTM (d) GRU

Fig. 5: Accuracy results for Sentiment Analysis Experiment



(a) CNN (b) LSTM (c) Bi-LSTM (d) GRU

Fig. 6: Accuracy results for POS Tagging Experiment



(a) CNN (b) LSTM (c) Bi-LSTM (d) GRU

Fig. 7: Accuracy results for Machine Translation Experiment

7 Conclusion

This work compared two of the main types of deep learning architectures, convolutional neural networks and recurrent neural networks on three common NLP tasks. The experiment was designed to give some insight as to which architecture is more suited to each NLP task. The results showed that while CNN were not originally designed for NLP tasks, they perform very well and are an ideal choice for spatial data tasks such as Sentiment Analysis where sequence and structure are not as important as compared with tasks such as Machine Translation. When the task is reliant on sequence and structure, then the RNN models are generally better suited. In this experiment however, CNN outperformed the RNN models on both Sentiment Analysis and POS-Tagging.

While these results have given some insight into the pros and cons of both CNN and RNN models and when one may be preferable to the others, ultimately every NLP task will be different so the chosen model will depend on the variables of each task, i.e type and size of training data, computing power and expected outcomes.

The main goal of this work was to develop a solid understanding of the make-up and workings of deep neural networks by experimenting with different architectures on different NLP tasks. I now have a much stronger understanding and appreciation of DNN.

8 Supplemental Material

Please find the code used for this project on github

`https://github.com/selinamead/CNN-vs-RNN-for-NLP-Tasks`

References

[1] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho and Yoshua Bengio. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 2014.

[2] Sepp Hochreiter and Juergen Schmidhuber. *Long Short Term Memory* [Neural Computation] 1997.

[3] Wenpeng Yin, Katharina Kann, Mo Yu and Hinrich Schutze. *Comparative Study of CNN and RNN for Natural Language Processing*. 2016.

[4] Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. *Combining recurrent and convolutional neural networks for relation classification*. 2016.

[5] K. Arkhipenko, I. Kozlov, J. Trofimovich, K. Skorni- akov, A. Gomzin, and D. Turdakov. *Comparison of neural network architectures for sentiment analysis of russian tweets.* 2016.

[6] Shuoheng Yang, Yuxin Wang, Xiaowen Chu. *A Survey of Deep Learning Techniques for Neural Machine Translation.* 2020.

[7] Zhang, Y., Wallace, B. *A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification.* 2015.

[8] Yoav Goldberg. *A Primer on Neural Network Models for Natural Language Processing.* 2015.

[9] Razvan Pascanu, Tomas Mikolov, Yoshua Bengio. *On the difficulty of training Recurrent Neural Networks.* 2012.