Recurrent neural network in NLP

Sütő Evelyne, Group: 246

December 22, 2019

Contents

1 Introduction **1**

2 Theory **1**

2.1 Encoder-Decoder Model . . . 1

2.2 Decoding Algorithms . . . 2

2.2.1 Greedy Decoding . . . 3

2.2.2 Beam-Search algorithm . . . 3

3 Experiments **4**

3.1 Neural Machine Translation . . . 4

3.2 Speech recognition: Speech to text translation . . . 5

3.3 Text summarization . . . 5

3.4 Paraphrase generation . . . 6

4 Conclusion **6**

Abstract

The purpose of this paper is to introduce a few experiments in natural language processing problems using recurrent neural networks.

Before going through the various experiments from this field we need to understand the baseline model used to solve these experiments. We will present a basic Encoder-Decoder model and we will go through the decoding algorithms used to decode the output from the prediction.

In this paper we will explore many complex NLP problems which can be used by applying this baseline model to solve it. In addition we will also see that in most cases there is a need to optimize this model further to fit the actual problem better. The experiments presented here include: neural machine translation, speech recognition, text summarization and paraphrase generation.

1 Introduction

Natural Language Processing (NLP) problems have been researched heavily in the last few decades, however its impact has never been so great as it is today. This is mostly because we are trying to automate more and more processes. However this automation process becomes very difficult once the understanding of natural language is needed. In those cases, such as customer support chatbots, we need to solve problems such as text to speach recognition, language modeling, text generation and more. One architecture in the field of machine learning which is suitable for these problems is Recurrent Neural Networks (RNN).

The aim of this paper is to provide an overview of a popular model used for NLP tasks: Encoder-Decoder model 2.1, which usually uses two seperate RNN for the encoder and the decoder module. In addition we will have a look into two algorithms 2.2 which will help us decode the predictions given by our model: Greedy algorithm and Beam search algorithm.

After this theoretical background we will focus on presenting a variety of applications in which this architecture can be used 3. We will see what additional changes each applications adds to achieve high performence in each tasks.

The tasks mentioned in this paper will be: neural machine translation 3.1, text summarization 3.3, speech recognition 3.2 and paraphrase generation 3.4.

2 Theory

2.1 Encoder-Decoder Model

In order for us to understand the experiements that will be presented in this paper first we need to have a better understanding of the Encoder-Decoder model which will be used in many of these examples. This model is also referred to as the sequence to sequence model. One simple representation of this model can be seen on 1

Since the input can be of varying size which could not be modeled by a recurrent network a simple strategy is to map the input sequence to a fixed sized vector using one RNN and then map the vector to the output sentence with another RNN. Since this model might need to learn really long sentences we need an archictecture that is capable of learning

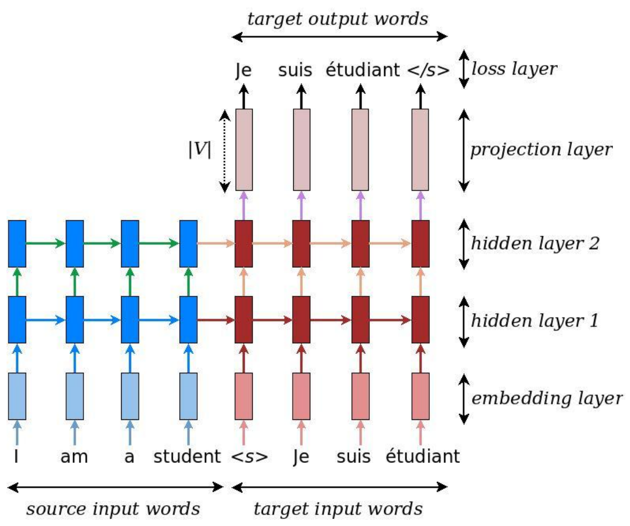

Figure 1: Neural machine translation example of a deep recurrent architecture proposed by for translating a source sentence "I am a student" into a target sentence "Je suis tudiant". Here, "s" marks the start of the decoding process while "/s" tells the decoder to stop. Image source: https://github.com/lmthang/thesis/blob/master/thesis.pdf

long term dependencies such as Long Short-Term Memory Networks. The idea is to use an LSTM to read the input data one time step at a time to obtain a fixed dimensional vector representation, and then to use another LSTM to extract the output sequence from that vector. So the LSTM's goal will be to estimate the probability of $P(y_1, y_2, .., y_l|x_1, x_2, .., x_t)$ where x is the input sequence and y is the output sequence whose length might be different. It is also required for the input sequence to end with a special *Stop* character. The model is based on two components: the first one for the input sequence the a second for the output sequence. [Sutskever et al., 2014]

It is also important to mention that the reasearchers in [Sutskever et al., 2014] have found that reversing the source sentence can lead to better results even though they can't really explain why that might be.

The result of the second layer will be a matrix of probabilities from which we can decode our prediction using two approaches: maximum likelihood method and beam search algorithm which will be discussed in 2.2.

As we can see on Figure 1 a conventional Encoder-Decoder model usually has an Embedding layer which translates the input sentence to a dense vector representation using the Vocabulary that we provide in the beginning. The output of the embedding is then fed to the encoder model which will return the input for the decoder. The output from the decoder layer then can be processed to retrieve the prediction.

2.2 Decoding Algorithms

In the literature usually there is two main algorithms which are used to extract the output of the decoder.

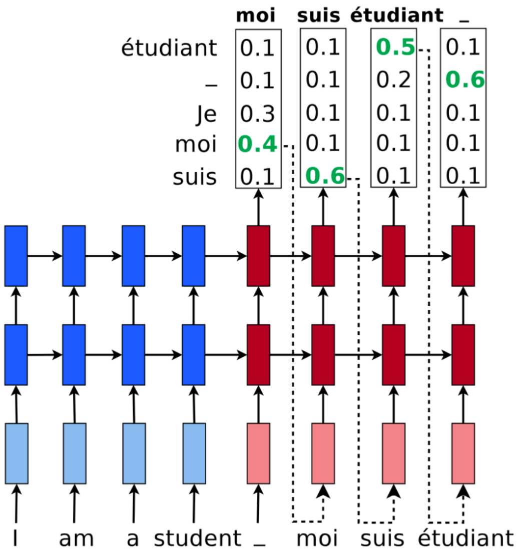

Figure 2: Greedy decoding algorithm example of a deep recurrent architecture proposed by for translating a source sentence "I am a student" into a target sentence "moi suis etudiant" Image source: https://github.com/lmthang/thesis/blob/master/thesis.pdf

2.2.1 Greedy Decoding

Greedy decoding is one of the simplest approach for decoding sentences and it is based on conditional possibility calculations.

In greedy decoding, we follow the conditional dependency path and pick the symbol with the highest conditional probability so far at each node. This is equivalent to picking the best symbol one at a time from left to right in conditional language modelling. [Gu et al., 2017]

However this does not gives us the best results because it only takes into consideration the element at previous time step which means that if it makes a wrong prediction every prediction made after that will be based on false information.

2.2.2 Beam-Search algorithm

Beam search keeps $K > 1$ hypotheses, unlike greedy decoding which keeps only one during decoding. At each time step t, beam search picks K hypotheses with the highest scores

$$\prod_{i=1}^{t} p(y_t|y < t, X)$$

When all the hypotheses terminate (outputting the end-of-the sentence symbol), it returns the hypothesis with the highest log-probability. Despite its superior performance compared to greedy decoding, the computational complexity grows linearly w.r.t. the size of beam K, which makes it less preferable especially in the production environment. [Gu et al., 2017]

The reason these algorithms and model has been presented in detail is because many experiments that will be presented uses them.

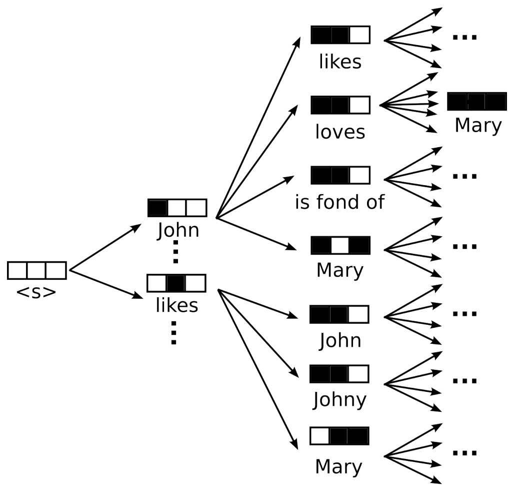

Figure 3: Beam-Search algorithm At each time step each node is expanded with the K most promising new nodes.
Image source: http://mttalks.ufal.ms.mff.cuni.cz/index.php?title=File:Beam-search.png

3 Experiments

3.1 Neural Machine Translation

One of the fields where sequence to sequence models were successfully applied and has yielded state of the art results if machine translation.

One of these researches were conducted for English-French translation in [Sutskever et al., 2014] where researchers have used two connected LSTM models. They have concluded that deep LSTMs can significantly outperform shallow LSTMs. Another important element introduced by that was the fact that reversing the source sentence can improve the model's accuracy. Furthermore they have shown that LSTM sequence to sequence models are able to map even very long sentences to the translation language successfully. This simple unoptimized model was able to produce state of the art results with relatively short training. For example models which use Statistical Machine Translation methods with neural networks produce 37 BLEU score while this models best result was 36.5 BLEU score.

BLEU is an automatic evaluation metric used for machine translations tasks. The way that Bleu metrics work is to compare the output of a machine translation system against reference human translations. The primary reason that Bleu is viewed as a useful stand-in for manual evaluation is that it has been shown to correlate with human judgments of translation quality. [Callison-Burch et al., 2006]

Later on this model's ([Sutskever et al., 2014]) shortcomings are addressed in [Wu et al., 2016] to improve Google's translation system. The main shortcomings of this systems recognized by them was the expensive training, lack of robustness, poor translation when rare words are present and the issue to NMT systems in practical deployments and services. They propose many changes to the architecture itself such deeper encoders and decoders (8 layers), Attention mechanism added between the encoder and decoder module, they introduced residual connections between the LSTM layers to speed up their training and they use

Bi-Directional Encoder to get a better context for the translation. The problem of rare word translation has been solved partially by copying the rare word entirely in the target sentence since these words usually represent names and they also use a more intelligent approach with sub-word units implementation. The evaluation of their model is done on two translation tasks: English-Frech and English-German, where on the English-French dataset the results improved by 1 BLEU score while the English-German with 0.4 BLEU score. It is also presented that such a model can easily be used in production as well improving translation errors by more than 60% even for such challanging tasks as English-Chinese translation.

3.2 Speech recognition: Speech to text translation

Another subfield where LSTM can be used is for speech recognition task.

As used by the authors from [Graves and Jaitly, 2014]. The authors have chosen to use an LSTM to exploit the long range of contexts from the training data and the LSTM's capability to store information. Instead of using a simple RNN they have chosen to use a Bidirectional RNN to exploit the future context as well, not just previous ones. A Bidirectional RNN processes the information in both directions with two separate hidden layers, then calculating the output based on these informations. In order to improve the training of the network they use Connectionist Temporal Classification function which allows the RNN to be trained sequence transcription without requiring prior alignment between target and input data. Their chosen decoding algorithm to decode the sentences was the Beam Search algorithm. In order to train their model they have chosen the Wall Street Journal corpus. The experiments were then scored on word error and character error rate. Their experiments conclude that even though it does not outperform the baseline model, considering that this model does not need data preprocessing it can be viewed as a sucessfull model. Their best scores on the 14 hour dataset was 13.5 while the baseline is 9.4 and on the 81 hour dataset their result was 8.2 while the baseline was 7.8.

3.3 Text summarization

Abstractive text summarization is the task of generating a headline or a short summary consisting of a few sentences that captures the salient ideas of an article or a passage. [Nallapati et al., 2016]

The researchers in [Nallapati et al., 2016] view this problem as mapping a sequence of data to another sequence. As a result they have chosen the Encoder-Decoder model with a bi-directional GRU-RNN as the encoder and a uni-directional GRU-RNN. They also use an Attention mechanism between the two modules. They also address the problem of rare words by using a so called Switching Generator-Pointer model. In this model, the decoder is equipped with a switch that decides between using the generator or a pointer at every time-step. If the switch is turned on, the decoder produces a word from its target vocabulary in the normal fashion. However, if the switch is turned off, the decoder instead generates a pointer to one of the word-positions in the source. The word at the pointer-location is then copied into the summary. The switch is modeled as a sigmoid activation function over a linear layer based on the entire available context at each timestep. [Nallapati et al., 2016] In their experiments they have used 3 datasets: Gigaword corpus, DUC corpus and

CNN/Daily Mail corpus. Their model has outperformed the state of the art results on DUC and CNN/Daily Mail corpus.

3.4 Paraphrase generation

Paraphrasing, the act to express the same meaning in different possible ways, is an important subtask in various Natural Language Processing (NLP) applications such as question answering, information extraction, information retrieval, summarization and natural language generation. [Prakash et al., 2016] Paraphrase generation can be used to improve many NLP tasks such as question answering, translation, text summarization just by generating new forms of the inputs.

Even though paraphrase generation can be used to improve many tasks it was just recently started to spark interest in researchers.

In [Prakash et al., 2016] the authors try to solve this problem by using an Encoder-Decoder model with Deep LSTMs with stacked residual connections. This model is evaluated on three very different datasets. The PPDB is a paraphrase dataset which includes only various short paraphrases, WikiAnswers with 18M question pairs and MSCOCO which contains image captions. The authors have compared 3 other models with their proposed one. On each corpus their model outperformed the other 3: sequence to seqeunce, sequence to sequence with attention and Bi-directional LSTMs.

4 Conclusion

We have explored the baseline model used in NLP tasks. We have presented the Encoder-Decoder model together with the two most used decoding algorithms to decode the output of these architectures.

We have also explored various applications where this architecture can be used by presenting four important research experiemnts from this field.

Having seen these various experiments we can conclude that the use of recurrent neural network is very popular in NLP tasks. One of the main reasons for that is the LSTMs power of capturing long term dependencies in data. The results reported in these researches also show that in most tasks in order to get successfull results there is a need to alter the baseline model to the task in variuos ways e.g. adding Attention mechanism, using residual connections. However after introducing the needed alterations can result in state of the art results as seen in these experiements.

References

[Callison-Burch et al., 2006] Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluation the role of bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

[Graves and Jaitly, 2014] Graves, A. and Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772.

[Gu et al., 2017] Gu, J., Cho, K., and Li, V. O. (2017). Trainable greedy decoding for neural machine translation. *arXiv preprint arXiv:1702.02429*.

[Nallapati et al., 2016] Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B., et al. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

[Prakash et al., 2016] Prakash, A., Hasan, S. A., Lee, K., Datla, V., Qadir, A., Liu, J., and Farri, O. (2016). Neural paraphrase generation with stacked residual lstm networks. *arXiv preprint arXiv:1610.03098*.

[Sutskever et al., 2014] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

[Wu et al., 2016] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.