Machine Translation

A PREPRINT

Anubhav Panda*
Department of Computer Science
University Of Minnesota,Twin Cities
Minnesota,Mn-55414
panda047@umn.edu

May 15, 2019

ABSTRACT

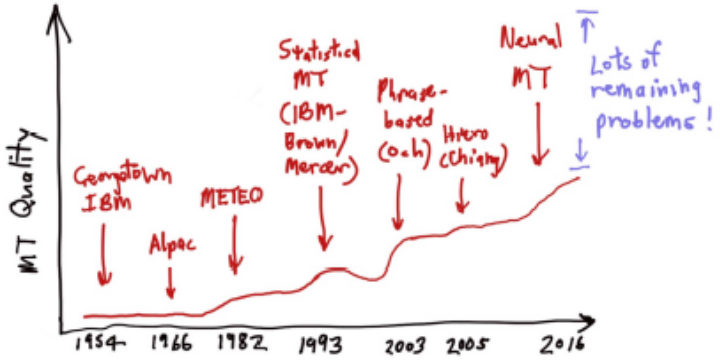
Human Share their thoughts,Ideas,Opinions and emotions by communicating with each other and Being able to communicate with each other using the all sets of human language is an ultimately rewarding goal for an intelligent system.Recently, Neural Machine translation is rising as a solution to the machine translation problems.If you see its Structure it consists of deep neural networks to directly map the source and the target sentences.An attentional mechanism has lately been used to improve neural machine translation(NMT) by selectively focusing on parts of the source sentence during translation.

***K*eywords** Neural Machine translation · Attention · Deep neural Network

1 Introduction

Human languages are diverse with about 6000 to 7000 languages spoken worldwide. As civilization advances, we need to convert one language to the other language for sharing the ideas among the communities. Machine translation (MT), the task of teaching machines to learn to translate automatically across languages, as a result, is an important research area. According to the Wikipedia Machine translation, sometimes referred to by the abbreviation MT is a sub-field of computational linguistics that investigates the use of software to translate text or speech from one language to another.it has significantly improving overtime as in terms of accuracy and output as we can see in the following figure 1. With the increase in accuracy, it is being used in more areas of business, introducing new applications and improved

Figure 1: Machine translation progress – from the 1950s, the starting of modern MT research, until the time of this thesis, 2016, in which neural MT becomes a dominant approach.



*Implementation can be found in https://github.com/anubhavpanda2/NeuralMachineTranslation-With-Attention

machine-learning models. It is being used in the following sectors such as Government, Software and technology, Military and defense, Healthcare, Finance, Legal, E-discovery, E-commerce. A machine translation system can be used to a specific domain by using training data from the same domain. For example, in order to be used an MT system for the legal domain, training data including the most commonly used legal terms, keywords, phrases, terminology, etc., in the legal domain are compiled into corpora, which act as an exhaustive data repository for the MT system to refer to and train on.

One of the applications of machine translation is Online Machine Translation for Consumer Use These applications are usually cloud-based which are typically trained on crowd-sourced data. Machine Translation can be used for the general purpose as well. There are many popular translators like Google Translate and Microsoft Translator which are providing the online applications. These are mostly used by individual consumers. And this kind of application offers the following functionalities

Text-to-text(google Translator), Text-to-speech(Google Assistant), Speech-to-text(Google Assistant,Speech notes), Speech-to-speech etc.

European Commission is the largest institutional user of MT. As they have supported the molto project received more than 2.375 million euros project support from the EU to create a reliable translation tool that covers a majority of the EU languages.it can be used in military services as well.According to the Wikipedia United States have been investing significant amounts of money in natural language engineering. Currently, the military community is interested in translation and processing of languages like Arabic, Pashto, and Dari, etc.

As shown in figure 1, The Translation Quality is improving day by day and it motivates me to learn about the Machine Translation.The goal is to implement a simple NMT model with Attention which will translate The English Statement to French Statement as shown in the example:

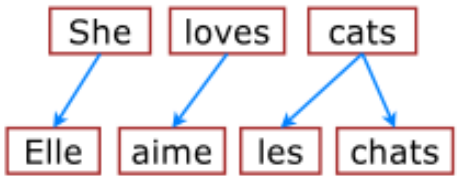
English :" i am a student"

French : "je suis un étudiant"

2 Review of related work

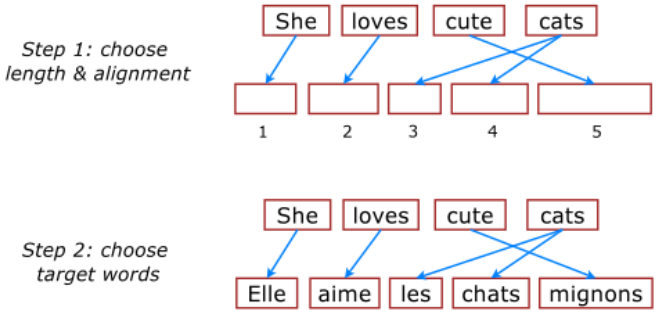
The History of MT begins 1950-1960 Where translation is being done by word to word Replacement Policy.In 1980, With the collaboration of IBM scientists, modern statistical MT has started. instead of hand building bilingual dictionaries which is very costly to build, they proposed to learn these dictionaries or translation models. To accomplish this, they propose a series of 5 algorithms of increasing complexity, often referred as IBM Models 1-5, to learn word alignment between source and target words, as illustrated in the following Figure 2. The idea is simple: the more often two words, e.g., “loves” and “time”, occur together in different sentence pairs, the more likely they have equivalent meanings. Once a translation model, i.e., a probabilistic bilingual dictionary, has been learned, IBM model translates

Figure 2: Word-based alignment – example of an alignment between source and target words. In IBM alignment models, each target word is aligned to at most one source word.



a new source sentence as follows. First, it decides on how long the translation is as well as how source words will be mapped to target words as illustrated in Step 1 of Figure 3. Then, in Step 2, it produces a translation by selecting for each target position a word that is the best translation for the aligned source word according to the bilingual dictionary. Subsequent IBM models build on top of one another and refine the translation story such as better modeling the reordering structure, i.e., how word positions differ between source and target languages. There are, however, two important details that we left out in the above translation story, the search process and the language modeling component. In Step 1, one might wonder among the exponentially many choices, how do we know what the right translation length is and how source words should be mapped to target words? The search procedure informally helps us “browse” through a manageable set of candidates which are likely to include a good translation; whereas, the language model will help us select the best translation among these words. in a nutshell, a language model (LM) learns from a corpus of monolingual text in the target language and collect statistics on which sequence of words are likely to go with

Figure 3: A simple translation story – example of the generative story in IBM Model 1 to produce a target translation given a source sentence and a learned translation model.



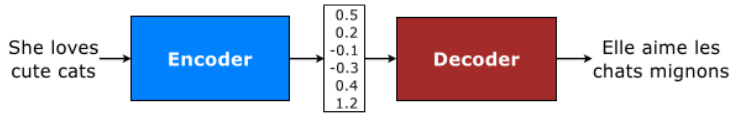
one another. When applied to machine translation, an LM will assign high scores for coherent translations and low scores for bad ones. For the example in the above figure3, if the model happens to choose a wrong alignment, e.g., "cute" go to position 3 while "cats" go to positions 4 and 5, an LM will alert us with a lower score given to that incorrect translation "Elle aime mignons les chats" compared to the translation "Elle aime les chats mignons" with a correct word ordering structure.

Researchers quickly realized that word-based MT is insufficient as words require context to properly translate, e.g., "bank" has two totally different meanings when preceded by "financial" and "river". it does not work very well and suffers from the following two major draw- backs. First, translation decisions are locally determined as we translate phrase-by-phrase and long-distance dependencies are ignored. More problematically, the entire MT pipeline is becoming increasingly complex as more and more features are added to the log- a linear framework such as in recent MT systems.

Neural Machine Translation (NMT) is a new approach that addresses the aforementioned problems. First, NMT is a single big neural network (with millions of artificial neurons) that is designed to model the entire MT process. NMT requires minimal domain knowledge, just a parallel corpus of source and target sentence pairs, similar to SMT, but with far less preprocessing steps before a translation model can be built. The most appealing feature of NMT is that it can be trained end-to-end directly from the learning objective; hence, eliminating the problem of having to learn multiple components in SMT systems.

Unlike those intricate decoders (the search procedure i mentioned earlier) in popular SMT packages the translation story of NMT uses seq to seq models.approach of solving these kinds of problems has been developed recently by apply Recurrent Neural Network models to learn how to convert one sequence to another. Formally, this is modeled with an encoder-decoder framework, where one set of RNN cells are used as an encoder and another set as a decoder. Encoder cells take input sequence as input and generate one vector, formally known as a thought vector, which is fed to the decoder. The decoder takes this vector representation of the input sequence and generates the required output sequence. Vanilla architecture is further improved with techniques like GRU or LSTM, beam search, attention, layer normalization, etc.

Figure 4: Encoder-decoder architecture – example of the general approach for NMT. An encoder converts a source sentence into a meaning vector/thought vector which is passed through a decoder to produce a translation



Recently,the concept of "attention" has been used in training neural networks. It expects the models to learn alignments between different modalities, e.g., between image objects and agent actions in the dynamic control problem, between speech frames and text in the speech recognition task, or between visual features of a picture and its text description in the image caption generation task .In the above mentioned systems attention has helped them to improve the performance successfully.and it is used in NMT to improve the performance as well.

3 Methods(Components used in Algorithms)

In this section, I will discuss about the different method that we have surveyed and implemented in the project. I will start with explaining about vanilla RNN, GRU, Seq to Seq models, models with attention, BLEU Score.

3.1 RNN

Recurrent Neural Networks are being successfully applied for sequential data and in many natural language processing tasks. It takes a sequence as an input vector and generates a sequence of output vector by passing it through a set of nonlinear transformation as defined below. It is normally used for Temporal Sequences.RNN takes as input a sequence of vectors $x_1, x_2, ..., x_n$ and processes them one by one. It updates its memory and to produce a hidden state and that hidden state is used for the next input to predict its output. A simple Vanilla RNN Can Define its output AS

$$h_t = f(x_t, h_{t-1}) \tag{1}$$

In the above formula, f is an abstract function that computes a new hidden state given the current input x_t and the previous hidden state h_{t-1}. The starting state h_0 is often set to 0 though it can take any value as I will see later in the context of NMT decoders. A popular choice of f is provided below with σ(sigmoid) being a non-linear function such as sigmoid or tanh.

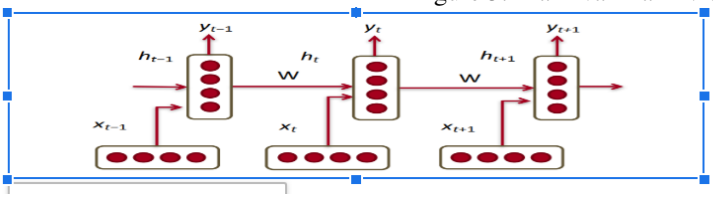
$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1}) \tag{2}$$

At each timestep t, an RNN can (optionally) emit an output symbol y_t which can either be discrete or real-valued. And it is defined as the follwoing equation 3.

$$y_t = softmax(W_s h_t) \tag{3}$$

And it can be shown in the following figure5.

Figure 5: Plain Vanilla RNN Structure



3.2 GRU

It Stands For Gated Recurrent Unit and it is a Variation for LSTM But it has fewer Parameters compared to it. it is used to solve the vanishing Gradient Problem. It is faster than the LSTM.To solve the vanishing gradient problem of vanilla RNN, GRU uses update gate and reset gate. these are two vectors helps it to decide what information should be passed to the output. The special thing about them is that they can be trained to keep information from long ago, without washing it through time or remove information which is irrelevant to the prediction. The following figure 6 explains the GRU System.

It contains 4 type of gates ie update gate,Reset gate,Current memory content and Final memory at current time step.Here i am assuming W_z represents the current weight assigned to state h_t and U_z represents the weight assigned to state h_{t-1}. Update Gate is Defined by the following formula 4 and z_t is defined as the update gate z_t for time step t .It helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future.

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{4}$$

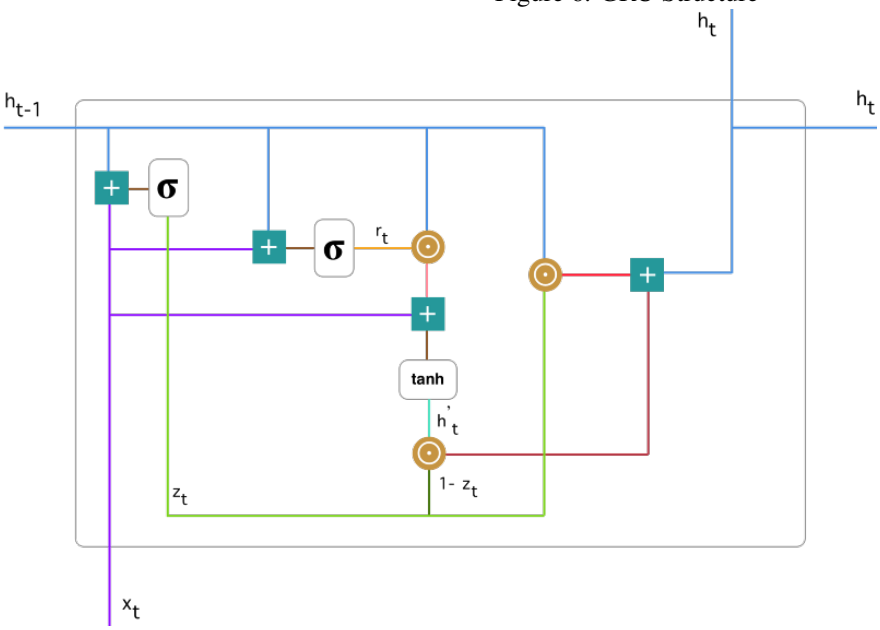
Reset gate is used from the model to decide how much of the past information to forget. To calculate it, following equation 5 can be used:

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{5}$$

Current memory content is used to calculate current hidden state using the following formula and the previous Reset gate

$$h_t = \tanh(W_z x_t + r_t \cdot U h_{t-1}) \tag{6}$$

Figure 6: GRU Structure



· represents element wise multiplication Final memory at current time step represents determines what to collect from the current memory content h_t and what from the previous steps h_{t-1} . That is done as follows:

$$h_t = Z_t \cdot h_{t-1} + (1 - z_t) \cdot h_t \tag{7}$$

3.3 Seq to Seq(Encode Decoder Model)

Sequence to sequence transformation is used to convert one sequence from another sequence.For example it can be used for transformation of one language to other language.And can be explained using the following simple figure 7.If both the input and the output has the same length it can be implemented using a simple LSTM or GRU Layer.If both the input and the output has different Length .Then the following steps need to be followed . As mentioned in the [10] Seq to Seq blog.

- Encode the input sequence into hidden State Vectors.
- Start with a target sequence of size 1 (just the start-of-sequence character).
- Feed the state vectors and 1-char target sequence to the decoder to produce predictions for the next character.
- Sample the next character using these predictions (I simply use argmax or I can use techniques like Beam Search).
- Append the sampled character to the target sequence.
- Repeat until we generate the end-of-sequence character or we hit the character limit.

3.4 Attention

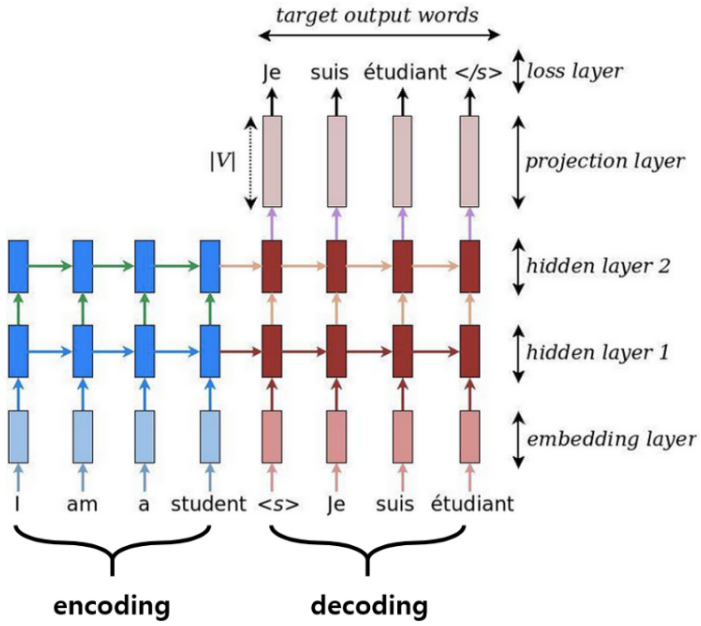
Attention is the Mechanism that helps the model to learn on specific parts of the input sequence while decoding instead of instead of relying only on the hidden vector of the decoder's output.Attention can be divided to two broad categories global and local. As mention in [1] Luong at el 2015,both approaches first take as input the hidden state h_t at the top layer of a stacking GRU. The goal is then to derive a context vector c_t that captures relevant source-side information to help predict the current target word y_t.While these models differ in how the context vector c_t is derived, they share the same subsequent steps. . If we have the output of hidden layer and context vector then we can combine both of them by the following equation 8

$$\overline{h_t} = \tanh(W_c[c_t : h_t]) \tag{8}$$

Then we can calculate the prediction by the following equation

$$p(y_t|y_{<t}, x) = softmax(W_s\overline{h_t}) \tag{9}$$

Figure 7: Encoder And Decoder using seq to seq model



to calculate c_t we need to compute α_t (also known as alignment vector)which is different for both the cases.once we know α_t, c_t can be computed by the equation 10.

$$c_t = \sum_s \alpha_{ts} \overline{h_s} \tag{10}$$

alignment vector α_t Global attention can be computed by the following formula 11

$$\alpha_t(s) = exp(score(h_t, \overline{h_s}))/sum_{s'} exp(score(h_t, \overline{h_{s'}})) \tag{11}$$

α_t can be derived by comparing a hidden state h_t with all the hiddenState h_s.And score(h_t, $\overline{h_s}$) can be derived by the following Figure 8 In the same way the local Attention can be computed by the following formula 12

Figure 8: calculation of score function

$$\text{score}(\boldsymbol{h_t}, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h_t}^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h_t}^\top \boldsymbol{W_a} \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{v_a}^\top \tanh\left(\boldsymbol{W_a}[\boldsymbol{h_t}; \bar{\boldsymbol{h}}_s]\right) & \textit{concat} \end{cases}$$

$$\alpha_t(s) = align(h_t, \overline{h_s}) exp(-(s - p_t)^2 / 2\sigma^2) \tag{12}$$

and p_t can be computed by the following equation

$$p_t = S.sigmoid(v_p^T \tanh(W_p h_t)) \tag{13}$$

W_p and v_p are the model parameters which will be learned to predict positions. S is the source sentence length.

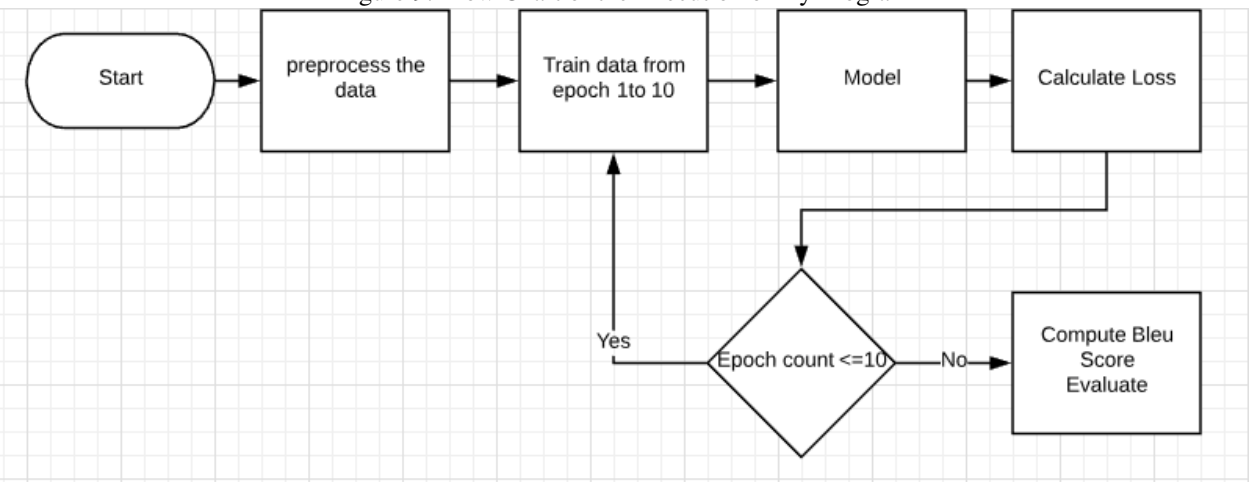
3.5 BLEU Score

It stands for bilingual evaluation understudy. It is used to measure the similarity between the quality of text which has been machine-translated from one natural language.According to the Wiki Scores are calculated for individual translated segments—generally sentences—by comparing them with a set of good quality reference translations. Those scores are then averaged over the whole corpus to reach an estimate of the translation's overall quality. The Bleu Score lies between 0 and 1, with values closer to 1 representing more similar texts.

4 Algorithm and Implementation

In this section I will explain the detailed implementation of the NMT using tensorflow.The code can be found in the following link `https://github.com/anubhavpanda2/NeuralMachineTranslation-With-Attention`. The Implementation can be explained as the following Flowchart(Which contains the Algorithm as described in figure 9).It has the following Sections Data Preparation,Model,Loss Calculation And computing the Bleu Score.I will explain about each module in the Following Sections. I am using a language dataset provided by `http://www.manythings.org/anki/`. This dataset contains language translation pairs in the following format:

Figure 9: Flow Chart of the Execution of my Program



English : I need it.

French :Il me le faut

Apart From That I am using Python 3.0 and and i am using TensorFlow in Google Collab. There are a variety of languages available, but i am using the French - English dataset. For convenience, it can be accessible from Google Cloud as i am running my code on Google-Collab.

4.1 Preprocess the DATA

The Preprocessing of Data is defined as Transforming the data before feeding to thr ML model. Once I have downloaded the Data I process it through three major methods like preprocess sentence,Unicode to ascii and tokenize .preprocess sentence method is used to Remove all the special characters.it also adds a start and end token to each sentence.it creates space between a word and the punctuation following it. It Cleans the sentences.unicode to ascii method is used to convert all the Unicode sequences to the Ascii Sequence.tokenize is used to generate the integer value by turning each text into either a sequence of integers (each integer is the index of a token in a dictionary) or into a vector where the coefficient for each token could be binary, based on word count, based on tf-idf Score.it also pads Each Sentences to its maximum length which will be useful for translation. I have also Limited the no of data processed by my model to 50000 so that it can be trained quickly. Training on the complete dataset of >100,000 sentences will take a long time.

4.2 Model

Here, I have implemented an encoder-decoder model with attention. An Encoder Converts the source sentence to a meaningful vector which will be passed To decoder Unit As explained in the Above Section. At first, it embeds the Data then it encodes it using GRU. The Decoder Computes the Attention At first it also computes the encoder output and then it combines the hidden Value generated by the Encoder And the output generated by the local/Global Attention And it passes it to the GRU Section of it. Then it computes the output(predictions) and attention weights. Briefly, I can say that the Model phase Contains The following Steps.

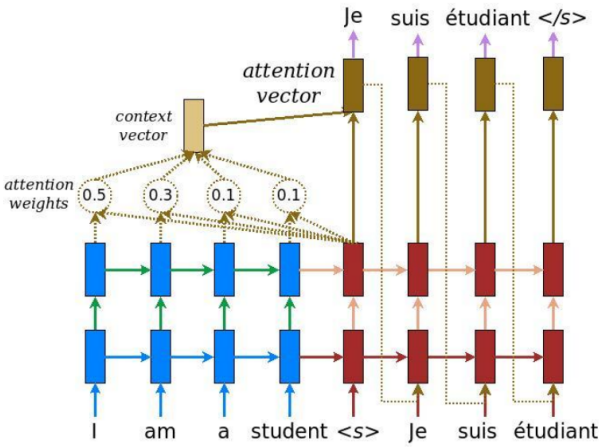
- Pass the input through the encoder which return encoder output and the encoder hidden state.
- The encoder output, encoder hidden state and the decoder input (which is the start token) is passed to the decoder.

- The decoder returns the predictions and the decoder hidden state.
- The decoder hidden state is then passed back into the model and the predictions are used to calculate the loss.

The following diagram (figure 10) shows that each input words is assigned a weight by the attention mechanism which is then used by the decoder to predict the next word in the sentence.

embedding output = The input to the decoder X is passed through an embedding layer. Combined vector = concat(embedding output, context vector) This Combined vector is then given to the GRU

Figure 10: Encoder Decoder Structure With Attention



4.3 Training Epochs And Loss Function

I am computing The Above Mentioned Encoder-Decoder Seq To seq Approach for each Sentence and That prediction and the ground truth are used to calculate the loss using Sparse Categorical Cross-entropy Loss. I am using ADAM Optimizer and teacher forcing to decide the next input to the decoder. I am computing the loss per Batch. The final step is to calculate the gradients and apply it to the optimizer and back-propagate. For the Time being, I am Training my model to 10 Epochs.

4.4 Model parameters

- Epochs count: 10
- num examples computed for training: 50000
- BatchSize : 64
- embedding dim : 256
- Optimizer: AdamOptimizer
- RNNCell: GRU

4.5 Calculating BLEU Score

Once The Training is done I am calculating The Bleu Score using nltk.translate.bleu score.sentence bleu method by passing the data of the test set. It takes two parameters like the predicted value and the ground truth value based on that it calculates The Bleu Score. I am using the Evaluate function to compute the predicted value. The evaluate function is similar to the training loop, except we don't use teacher forcing here. The input to the decoder at each time step is its previous predictions along with the hidden state and the encoder output. i am Stopping the prediction when the model predicts the end token And storing the attention weights for every time step.

5 Results

After Training The model for 50000 lines of French to English Conversion I am getting bleu score =52 for the toy dataset. But According [1]Luong 2015 They have trained the Same model for 10 epochs on the data on the WMT'14

training data consisting of 4.5M sentences pairs (116M English words, 110M German words). They got the Bleu score of 23.0 if they are applying Ensemble methods and unk replace techniques. They have Compared Their Work with The other Equivalent Works done in the NMT And They got The Following Results As shown in the following figure 11. Their code is implemented in MATLAB. When running on a single GPU device Tesla K40,They achieve a speed of 1K target words per second. It takes 7–10 days to completely train a model. They compared their NMT

Figure 11: Comparison of Bleu Score With other models

System	Ppl	BLEU
Winning WMT'14 system – *phrase-based + large LM* (Buck et al., 2014)		20.7
Existing NMT systems		
RNNsearch (Jean et al., 2015a)		16.5
RNNsearch + unk replace (Jean et al., 2015a)		19.0
RNNsearch + unk replace + large vocab + *ensemble* 8 models (Jean et al., 2015a)		**21.6**
My NMT systems		
Base	10.6	11.3
Base + reverse	9.9	12.6 (+*1.3*)
Base + reverse + dropout	8.1	14.0 (+*1.4*)
Base + reverse + dropout + global attention (*location*)	7.3	16.8 (+*2.8*)
Base + reverse + dropout + global attention (*location*) + feed input	6.4	18.1 (+*1.3*)
Base + reverse + dropout + local-p attention (*general*) + feed input	5.9	19.0 (+*0.9*)
Base + reverse + dropout + local-p attention (*general*) + feed input + unk replace		20.9 (+*1.9*)
Ensemble 8 models + unk replace		**23.0** (+*2.1*)

systems in the English-German task with various other systems. The Results include the winning system in WMT'14 [5] (Buck et al., 2014), a phrase-based system whose language models were trained on a huge monolingual text, the Common Crawl corpus. For end-to-end NMT systems, to the best of my knowledge, [6](Jean et al., 2015) is the only work experimenting with this language pair and currently the SOTA system. As shown in Table 1, they have achieved progressive improvements when (a) reversing the source sentence, +1.3 BLEU, as proposed in [2](Sutskever et al., 2014) and (b) using dropout, +1.4 BLEU. On top of that, (c) the attention approach gives a significant boost of +2.8 BLEU, making my model slightly better than the base attentional system of [3]Bahdanau et al. (2015) (row RNNSearch). The local attention model proves to be even better, giving us a further improvement of +0.9 BLEU . It is interesting to observe the trend previously reported in [4] (Luong et al., 2015) that perplexity strongly correlates with translation quality. In total, They achieved a significant gain of 5.0 BLEU points over the non-attentional baseline, which already includes known techniques such as source reversing and dropout.

6 Analysis

I have implemented a model without the attention using GRU and I got the BLEU Score of 48 While With Attention I am getting the BLEU Score of 52 on the Toy Dataset. extensive Analysis has been Done by [1] Luong at el 2015 to better understand the models in terms of learning, the ability to handle long sentences, choices of attentional architectures, and alignment quality. All results reported here are on English-German newstest2014. They compared models as shown in Figure 11. They observed that the non-attentional model takes less test time compared to attentional models when the batch size increases. And they Also observed that Attentional model are good at handling long sentences compared to the non -attentional model and it can be shown in the following figure12. According to The figure, The quality does-not Degrade as the length of the sequence increases. Attention based models are performing well compared to non attention based models in translating names such as "Miranda Kerr" and "Roger Dow" etc and it can be shown in the following figure 13. Non-attentional models, while producing sensible names from a language model perspective, lack the direct connections from the source side to make correct translations. According to [1] loung 2015 there is an interesting case I, which requires translating the doubly-negated phrase, "not incompatible". The attentional model correctly produces "nicht . . . unvereinbar"; whereas the non-attentional model generates "nicht vereinbar", meaning "not compatible" So attentional model are behaving better than non-attentional models. They Also Compared Different Attentional Models(Global,Local)And As we can see in the figure 11 The local Attention Based Model are performing better than Global Attention based models.

Figure 12: Comparison of Bleu Score With long sequences

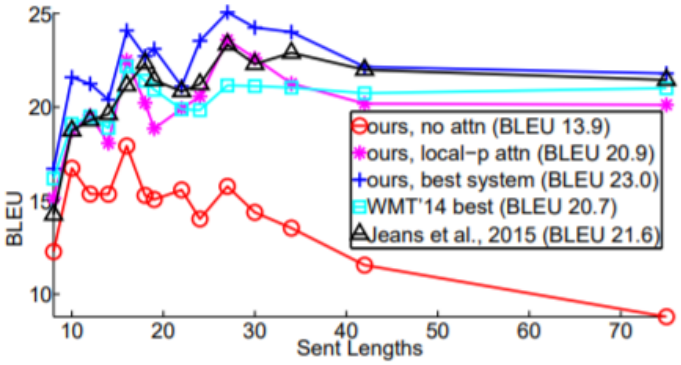


Figure 13: Quality Comparison in names Translation

English-German translations

src	Orlando Bloom and Miranda Kerr still love each other
ref	Orlando Bloom und *Miranda Kerr* lieben sich noch immer
best	Orlando Bloom und *Miranda Kerr* lieben einander noch immer .
base	Orlando Bloom und **Lucas Miranda** lieben einander noch immer .
src	" We ' re pleased the FAA recognizes that an enjoyable passenger experience is not incompatible with safety and security , " said Roger Dow , CEO of the U.S. Travel Association .
ref	" Wir freuen uns , dass die FAA erkennt , dass ein angenehmes Passagiererlebnis nicht im Widerspruch zur Sicherheit steht " , sagte *Roger Dow* , CEO der U.S. Travel Association .
best	" Wir freuen uns , dass die FAA anerkennt , dass ein angenehmes ist nicht mit Sicherheit und Sicherheit *unvereinbar* ist " , sagte *Roger Dow* , CEO der US - die .
base	" Wir freuen uns über die <unk> , dass ein <unk> <unk> mit Sicherheit nicht **vereinbar** ist mit Sicherheit und Sicherheit " , sagte *Roger* **Cameron** , CEO der US - <unk> .

7 Conclusion And Future Work

In this Implementation, I tried to implement SeqtoSeq RNN models with Attention on a toy Dataset. I tried to compare the effectiveness of the models with Attention and Without Attention in terms of their Bleu Score. I also tried to compare the performance of the models based on their test time according to the batch length. i compared the bleu score based on Handling of long sentences and their performance in the translation of the names. with attention, the model yields large gains of up to 5.0 BLEU over non-attentional models.They also examined different attention models (global, local) Due to limited resources, They could not run all the possible combinations. However, the global model can only obtain a small gain when performing unknown word replacement compared to using other alignment functions.14 For contentbased functions, our implementation concat does not yield good performances and more analysis should be done to understand the reason results in Table 4 do give us some idea about different choices. The location-based function does As shown in the following figure 10. My analysis shows that attention-based NMT models are superior to non-attentional ones based on standard WMT 14 English to German Dataset.I am planning to improve the performance of the model by using beam search .I want to explore MultiTask Learning and see whether it can improve the performance of the Existing model.I aspire to learn about Transformer Model which gives a BLEU Score of 31 on the same WMT 14 English to German Dataset and i want to implement it on a toy data set and compare its performance with the existing SeqtoSeq RNN model.

References

[1] Minh-Thang, Luong Hieu Pham, Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 417–422. IEEE, 2014.

[2] [Sutskever et al.2014] I. Sutskever, O. Vinyals, and Q. V. Le. 2014. Sequence to sequence learning with neural networks. In NIPS.

[3] [Bahdanau et al.2015] D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural machine translation by jointly learning to align and translate. In ICLR.

[4] [Luong et al.2015] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba. 2015. Addressing the rare word problem in neural machine translation. In ACL

[5] [Buck et al.2014] Christian Buck, Kenneth Heafield, and Bas van Ooyen. 2014. N-gram counts and language models from the common crawl. In LREC.

[6] [Jean et al.2015] Sebastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In ACL.

[7] machinr traslation wiki `https://en.wikipedia.org/wiki/Machine_translation`

[8] Neural Machine Translation with Attention `https://www.tensorflow.org/alpha/tutorials/text/nmt_with_attention`

[9] MT Luong Thesis `https://github.com/lmthang/thesis/blob/master/thesis.pdf`

[10] Keras Encoding decoder`https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-html`