Real-Time Neural Machine Translation

by

Ashkan Alinejad

M.Sc., University of Tehran, 2016
B.Sc., Shahid Beheshti University, 2014

Depth Report Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
Department of Computer Science

SIMON FRASER UNIVERSITY
Fall 2017

Abstract

Studies on Machine Translation (MT) has a long history, but most of the works in this area assumes we can have access to the entire sentences. As a result, it is not practical to apply them on Real-Time Machine Translation where the objective is to start translation *before* receiving the full sentences. Divergent syntax of different languages makes it a great challenge for both humans and machines to start translating while new inputs are still being received.

Over the past few years, the great success of using deep neural networks in Real-Time translation systems, led this field to evolve in completely new direction and improved the results; However, many of the problems from conventional systems remained unsolved. This report provides a review over the latest methods of utilizing neural attention models for the task of simultaneous machine translation.

Keywords: Neural Machine Translation; Real-Time; SNMT

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Recurrent Neural Networks

1.1.1 Long Short-Term Memory

1.1.2 Gated Recurrent Units

1.2 Evaluation Methods

1.2.1 BLEU Score

1.2.2 ROUGE Score

Chapter 2

Traditional Systems

Chapter 3

Neural Machine Translation

Neural Networks can be seen as an essential component in most of recent approaches in the field of machine translation. In section 1.1 we have reviewed various components of RNNs[1] . During this chapter, we will describe how we can use them to build translation systems that can extract semantic information from source language and follow the syntactical structure of target language in order to produce translated words.

3.1 Neural Translation Models

Most of the state-of-the-art approaches toward solving Machine Translation uses the Encoder-Decoder architecture with attention; However, since adding attention mechanism makes the structure more complex, we will start with simple Encoder-Decoder translation systems in the next section. Later, in 3.1.3 we will demonstrate how attention mechanism affects the translation process.

3.1.1 Encoder-Decoder structure

The very basic neural structure that can generate reasonable translations is what is called the **Encoder-Decoder** model [2, 3]. The concept of this model is to use stacked layers of LSTM cells in order to *encode* the whole source sentence into a real-valued vector. Then we can feed this vector to our second neural network to *decode* it and produce translated words one at a time. see figure 3.2 for illustration.
More mathematically, our network at time step t will receive a numerical representation of word x_t from the source sentence $X = [x_1, \ldots, x_T]$. Then it will combine them with output of encoder at previous time step and passes them to a non-linear function. In other words: $h_t = f(x_t, h_{t-1})$, where f is a non-linear function. Once the encoder receives the <eos> word, it will start to compute the final encoder's context vector h_T.
In decoder component, we will use the output from previous time step y_{i-1}, previous

[1]Recurrent Neural Networks

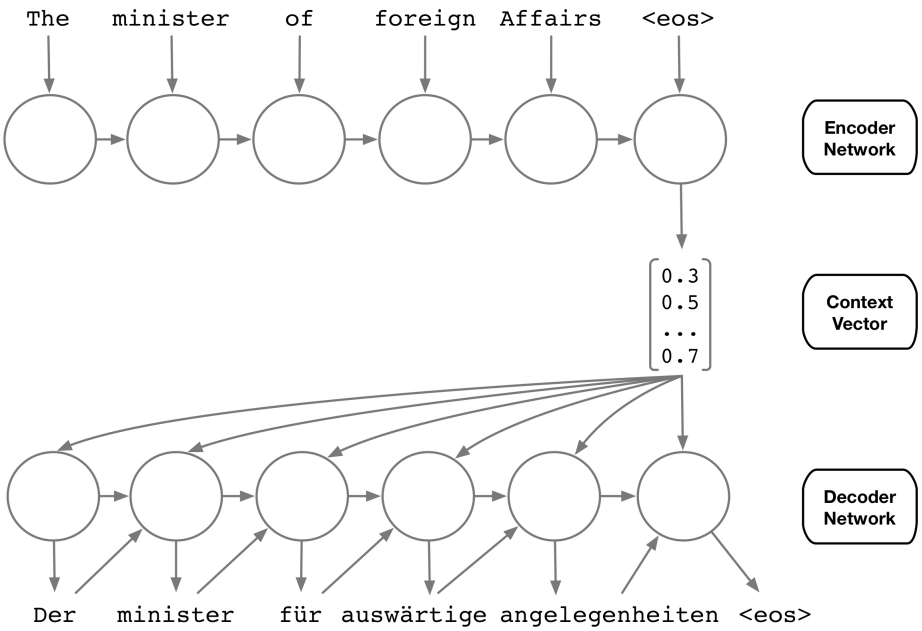

Figure 3.1: Structure of an Encoder-Decoder model.

decoder output s_{i-1} as well as h_T, as an input for decoder's neural network. We will compute s_i, similar to encoder's context vector, with $s_i = g(y_{i-1}, s_{i-1}, h_T)$, where g is a non-linear function. Then we will pass s_i through a softmax layer in order to compute the probability $P(y_i|X, y_{<i})$. We will choose y_i that maximizes the probability. i.e. $i = \arg\max_{y_i} P(y_i|X, y_{<i})$.

This is the very basic, yet powerful structure of the encoder-decoder model and while there are lots of neural models for machine translation, these can be seen as extensions to this model. In [3] it is shown that with some refinements (E.g. using BiLSTM instead of LSTM as encoder), the encoder-decoder structure can beat most of the approaches with many years of research in their background. We will talk about these techniques in the next two sections.

3.1.2 Beam Search

3.1.3 Attention Mechanism

We are only one step away to look at the state-of-the-art translation model which is the encoder-decoder model with attention mechanism. As we have seen in 3.1.1, the encoder-decoder networks forces the encoder to keep all the information required for decoding into a fixed-dimensional context vector. On the other side of this structure, the decoder only have

access to this context vector and it is supposed to produce the whole translation using this fixed representation of input.

Although, with keeping these restrictions in mind, the architecture works well, as the length of sentences grows, the decoder's performance decreases a lot. In order to fix these constraints, Bahdanau et al. [1] proposes the attention mechanism. The main idea is that instead of using the last state of the encoder's context vector, the decoder is able to use a weighted combination of encoder's output at different time steps. As a result, Not only the decoder would be powerful, but it would also be much more easier for encoder to encode input sentences at each time step.

More concretely, the first component of this structure encodes the embeddings of input words $X = \{x_1, \ldots, x_{T_s}\}$ into context vectors $H = \{h_1, \ldots, h_{T_s}\}$. It can be done by utilizing a bidirectional RNN:

$$h_i = \phi_{\text{BiRNN}}(h_{i-1}, x_i)$$

On decoder side we will have:

$$\alpha_i^\tau = \phi_{\text{ATTN}}(z_{\tau-1}, h_i) \tag{3.1}$$

$$c_\tau = \sum_{i=1}^{T_x} \alpha_i^\tau h_i \tag{3.2}$$

$$z_\tau = \phi_{\text{DEC}}(z_{\tau-1}, y_{\tau-1}, c_\tau) \tag{3.3}$$

$$p(y|y_{<\tau}, H) \propto \exp[\phi_{out}(z_\tau)] \tag{3.4}$$

$$y_\tau = \arg\max_y p(y|y_{<\tau}, H) \tag{3.5}$$

In equation 3.1, $z_{\tau-1}$ is the decoder's context vector at previous time step and ϕ_{ATTN} is the attention function which can be any function that measures how well input at position i is related to output at time step τ. The most commonly used function is the original formula presented by Bahdanau et al [1] which employs a multilayer feedforward neural network. We then pass these scores to a Softmax function in order to make their summation equal to one. The output of attention layer in equation 3.2 is using α_i^τ as probability as a means to compute a weighted average over the encoder's context vector.

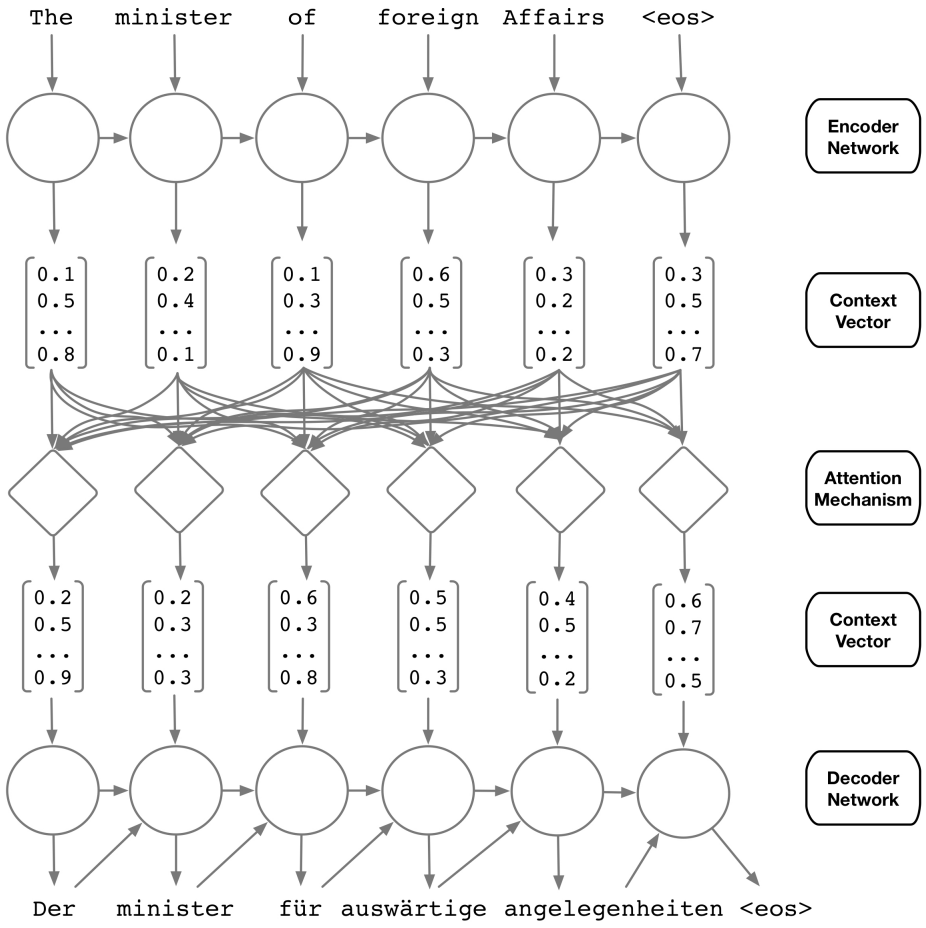

Figure 3.2: Structure of an Encoder-Decoder model with Attention mechanism.

Chapter 4

Real-Time Neural Machine Translation

In the first section of this chapter we will talk about a new approach to solve the task of translation in real-time, presented by Cho et al. We will look at a novel decoding algorithm, called *Simultaneous greedy decoding* which serves as the starting point of new family of algorithms to allow neural translation systems begin translating before receiving the full sentence. Then we will describe another method based on simultaneous greedy decoding in section 4.2, which shows us how we can train our systems to learn when to segment a sentence based on predefined quality and delay criteria.

4.1 Simultaneous Greedy Decoding

4.2 Learning Policy with Trainable Agent

Chapter 5

Results and Analysis

Chapter 6

Conclusion

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[2] Lonnie Chrisman. Learning recursive distributed representations for holistic computation. *Connection Science*, 3(4):345–366, 1991.

[3] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA, 2014. MIT Press.