NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

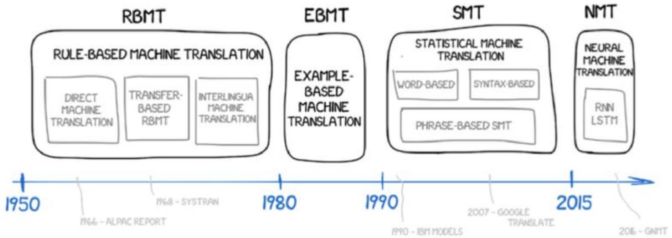
Bootcamp 2 (papers implementation)

Reporter : Maram A.Mohamed

Background NEURAL MACHINE TRANSLATION

- **traditional** phrase-based translation system - consists of many small sub-components that are tuned **separately.**

- neural machine translation attempts to build and train **a single**, large neural network that reads a sentence and outputs a correct translation.

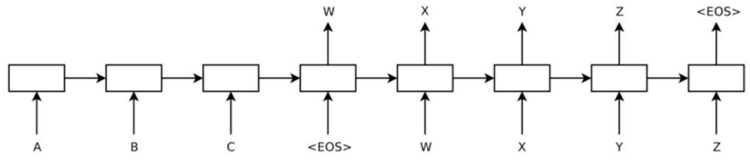


Background NEURAL MACHINE TRANSLATION

- From **a probabilistic perspective**, translation is equivalent to finding a target sentence y that **maximizes** the conditional probability of y given a source sentence x.

$$\arg\max_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x})$$

- In **NMT**, we fit a parameterized **model to maximize** the conditional probability of sentence pairs using a parallel training corpus.
- Consists of **two components**, the first of which **encodes** a source sentence x and the second **decodes** to a target sentence y.



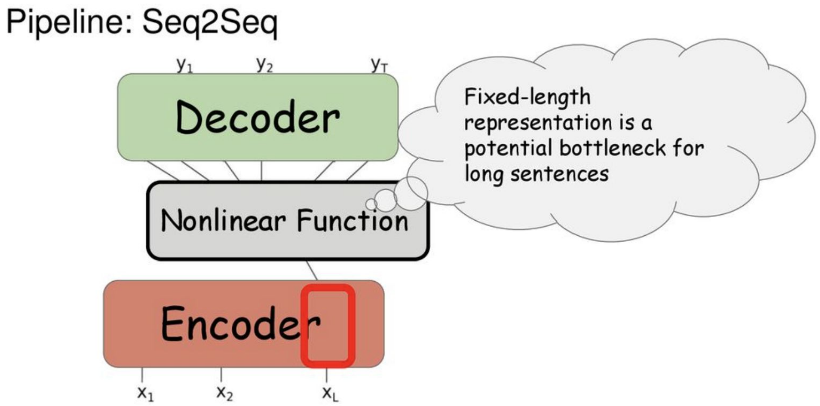
[Encoder – Decoder Architecture]

Basic Encoder-Decoder Using RNN / The main problem

- have done ***pretty well*** on this task.
- but are limited in their ability to track ***long-term dependencies*** .
- lose their ability to translate the end of long sentences correctly.

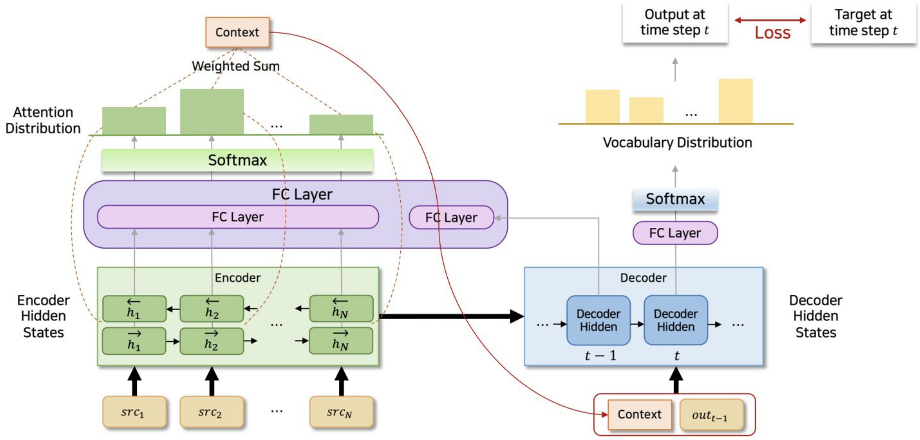
WHY?

- this "basic encoder-decoder" architecture encodes everything about the input sentence in ***a single fixed-length vector.***



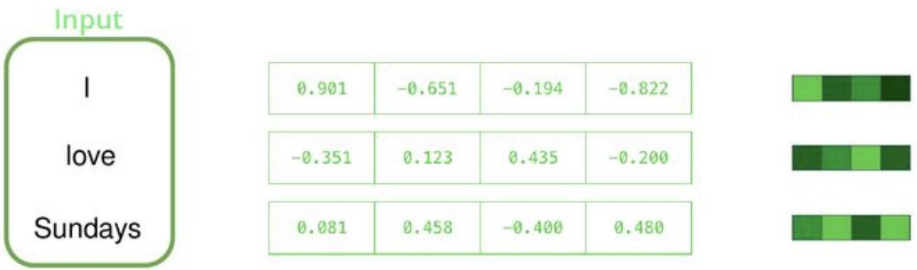
The proposed Model

- by using a **bidirectional LSTM** for input.
- second by introducing an ***alignment model***, a matrix of weights connecting each input location to each output location. This can be thought of as an **attention mechanism** that allows the decoder to pull information from useful parts of the input rather than having to decode a single hidden state.



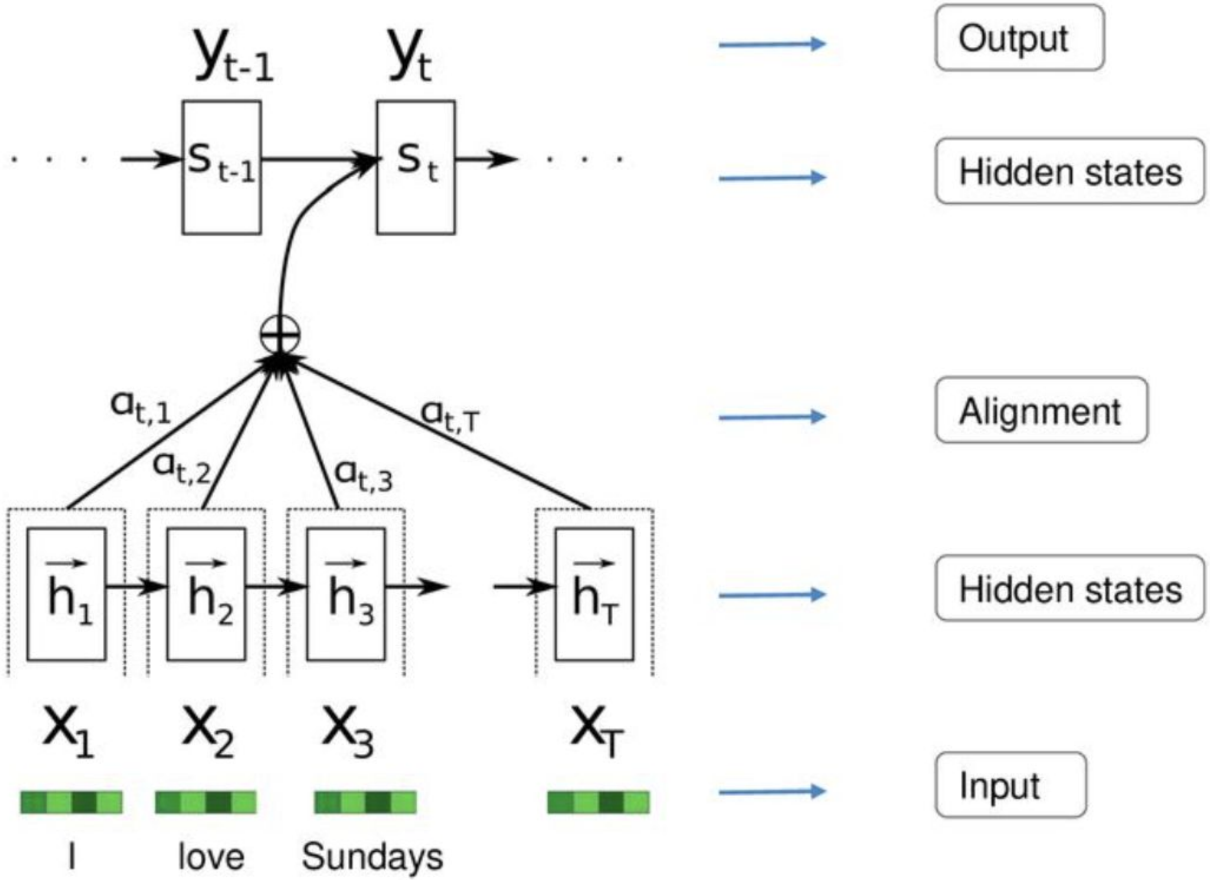
The full pipeline

- ***Word Embedding :***



The Full Pipeline

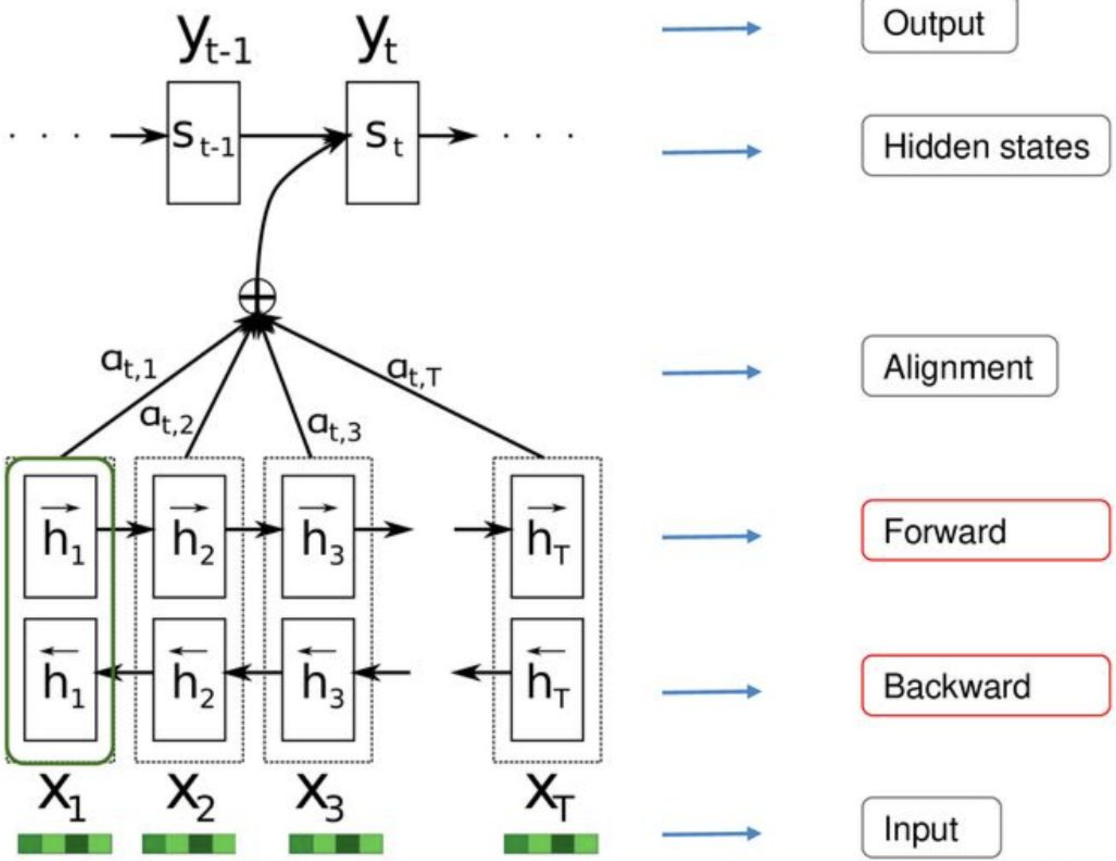
Annotations: each word only summarizes the information of its preceding words



The Full Pipeline

Bidirectional RNNs for the annotation hidden states

True annotations can be obtained by concatenating the forward and backward annotations



Alignment Model

In a new model architecture, we define each conditional probability in Eq. (2) as:

$$p(y_i|y_1, \ldots, y_{i-1}, \mathbf{x}) = g(y_{i-1}, s_i, c_i), \tag{4}$$

where s_i is an RNN hidden state for time i, computed by

$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

It should be noted that unlike the existing encoder–decoder approach (see Eq. (2)), here the probability is conditioned on a distinct context vector c_i for each target word y_i.

The context vector c_i depends on a sequence of *annotations* $(h_1, \cdots, h_{T_x})$ to which an encoder maps the input sentence. Each annotation h_i contains information about the whole input sequence with a strong focus on the parts surrounding the i-th word of the input sequence. We explain in detail how the annotations are computed in the next section.

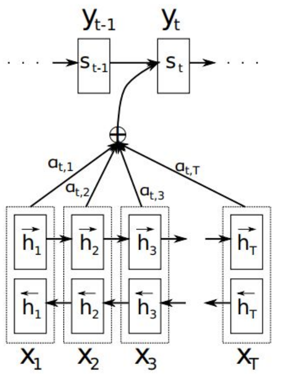


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

The context vector c_i is, then, computed as a weighted sum of these annotations h_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j. \tag{5}$$

The weight α_{ij} of each annotation h_j is computed by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \tag{6}$$

where

$$e_{ij} = a(s_{i-1}, h_j)$$

is an *alignment model* which scores how well the inputs around position j and the output at position i match. The score is based on the RNN hidden state s_{i-1} (just before emitting y_i, Eq. (4)) and the j-th annotation h_j of the input sentence.

We parametrize the alignment model a as a feedforward neural network which is jointly trained with all the other components of the proposed system. Note that unlike in traditional machine translation,

Experiments

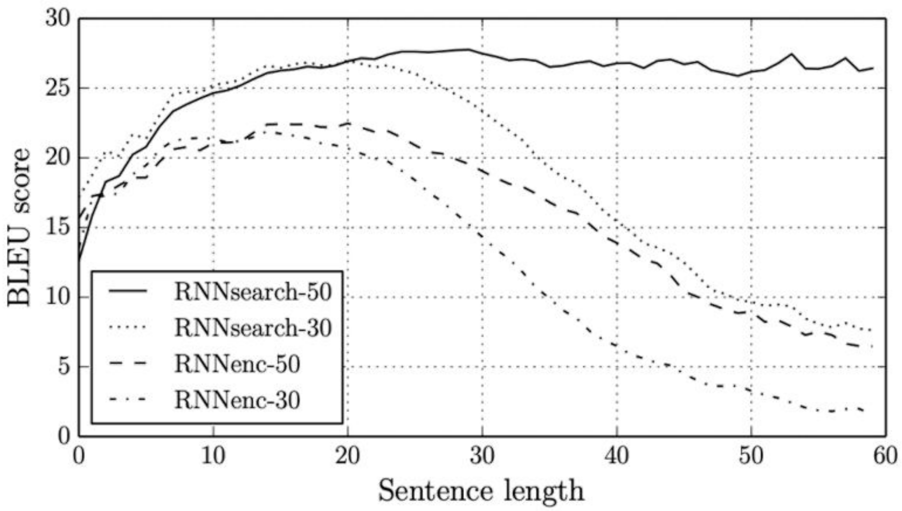
Quantative Results

- WMT'14 (English → French)
- **No UNK**: Sentences without any unknown word

  RNNsearch-50*: Trained much longer until the performance on the dev set stopped improving

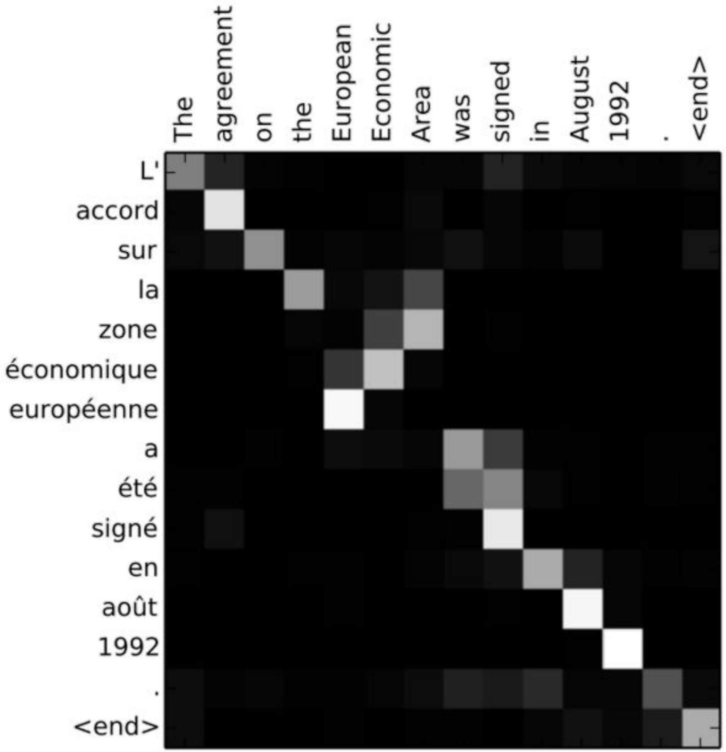
  Moses: Conventional phrase-based translation system (using separate monolingual corpus)

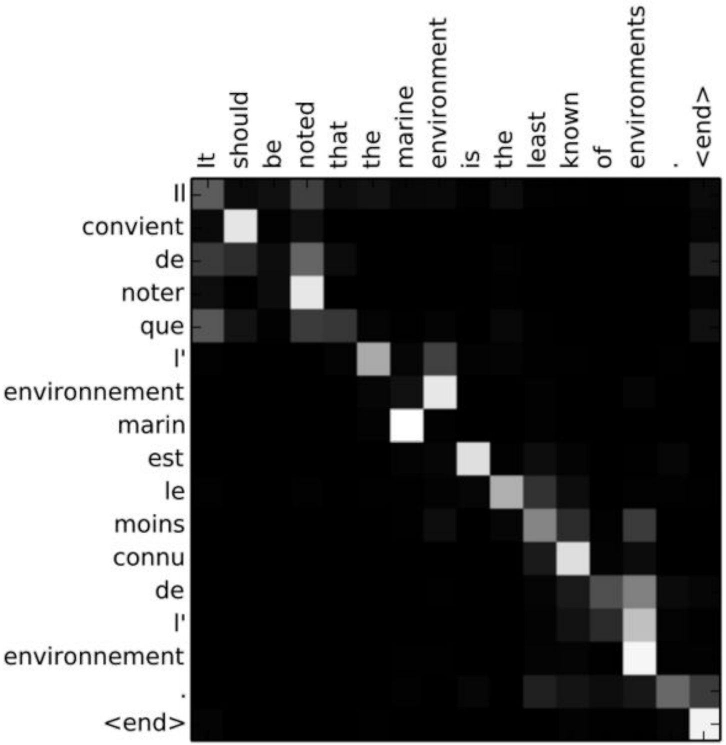
Model	All	No UNK°
RNNencdec-30	13.93	24.19
RNNsearch-30	21.50	31.44
RNNencdec-50	17.82	26.71
RNNsearch-50	26.75	34.16
RNNsearch-50*	28.45	36.15
Moses	33.30	35.63



Experiments

Qualitative Analysis (RNNsearch-50)





Conclusion

- Attention mechanism allows the network to refer back to the input sequence, instead of forcing it to encode all information into one fixed-length vector.

- Pros: Soft access to memory, Model interpretation
- Cons: Computational expensive

What I did So Far :

- French - English translation.
- Using IWSLT2016 from Torchtext.
- Spacy for tokenization.
- Built seq2seq model with BiRNN in Encoder part.
- RNN in Decoder part + Alignment Model using Attention Mechanism .
- I did not get the result yet - check the code
- To be continued for the next week if I am allowed to do so .