UE21CS343BB2

Topics in Deep Learning

Dr. Shylaja S S
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
shylaja.sharath@pes.edu

Ack:Anashua Dastidar, Teaching Assistant

Pitfalls in the Encoder Decoder model

In the previous lecture we saw the encoder decoder architecture , we can say that the encoder generates a representation for the entire sentence which is the output of the last encoder unit better known as the context vector.This vector is then taken in as input to the decoder which generates a step by step output .

This is similar to the reading a whole paragraph in one go and trying to translate it from memory and most of us would agree that this is a bad method to opt for while translating . Instead a more common experience while translating is one in which a text is broken down into small parts and each part is translated one after the other .

The same is the issue with the encoder decoder architecture.

Pitfalls in the Encoder Decoder model

- The context vector used here becomes a bottleneck with increase in sentence length as it is forced to represent more information using a fixed size vector.

- Moreover at any time step the decoder may not need information about the entire input sentence to generate the translation but the knowledge of only some specific words may be essential . Like in the following example to generate the word मिलकर it was enough to know that the word meet occurs in the corresponding english sentence.

Nice to meet you → आपसे मिलकर अच्छा लगा

- But in our encoder decoder model the decoder is provided with a representation of the whole sentence via context vector which does not allow the architecture to attend to specific words at specific timesteps .

The Attention Mechanism

Thus in order to overcome the pitfalls of the traditional encoder - decoder model the attention mechanism was proposed in the 2016 paper titled "**Neural Machine Translation by Jointly Learning to Align and Translate"** by Bahdanau et al., 2014 and further refined by Luong et al., 2015. We shall first discuss the idea discussed by by Bahdanau et al.

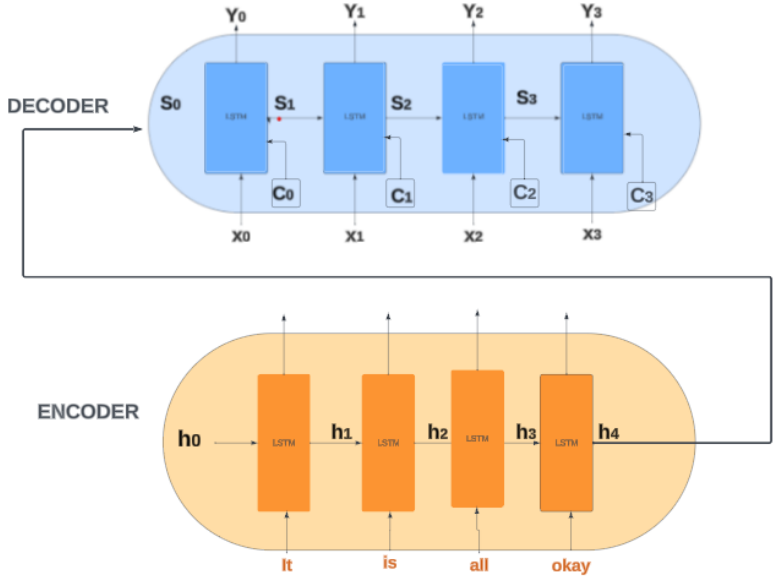
→ The authors proposed a simple solution which is instead of passing a single context vector i.e the last hidden state of the encoder to the decoder model , the encoder passes all of its hidden states to the decoder model .

→ Second, an attention decoder does an extra step before producing its output. In order to focus on the parts of the input that are relevant to this decoding time step, the decoder does the following:

1. Look at the set of encoder hidden states it received – each encoder hidden state is most associated with a certain word in the input sentence
2. Give each hidden state a score (let's ignore how the scoring is done for now)
3. Multiply each hidden state by its soft maxed score, thus amplifying hidden states with high scores, and drowning out hidden states with low scores

The Attention Mechanism

Assume we have the following model let h_0 - h_4 be the hidden states of the encoder , Let S_0 - S_3 be the decoder's hidden states , let x_0 - x_3 be the inputs to the decoder's cells and Y_0 - Y_3 be the output generated by the decoder. Thus to each LSTM unit in the decoder we add a third input called the attention input C_0 - C_3.



- The attention input C_i depends on a sequence of annotations ($h_1, \cdots, h_T$;T is the number of units) to which an encoder maps the input sentence.
- Each hidden layer output 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.
- Thus C_i is a weighted average of all h_i values. So for the i^{th} decoder timestep and j^{th} encoder timestep

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j .$$

The Attention Mechanism (Bahdanau attention)

Now , how to compute this value α_{ij}?

- is an alignment score which tells us the weight of the j^{th} hidden encoder state corresponding to the i^{th} decoder timestep.
- α_{ij} depends on h_j and S_{i-1} i.e the previous hidden state of the decoder ; this is necessary as we must find out how useful h_j is given that i-1 parts of the sentence have already been translated.

* softmax normalization

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \quad \text{where,} \quad e_{ij} = a(s_{i-1}, h_j) \quad \text{Here a is a function}$$

We know that ANNs are universal function approximators , thus the value of eij can be approximated using an neural net , this neural net is jointly trained with other components of the proposed system . Here alignment is not considered a latent variable but instead this model computers a soft alignment which allows the gradient from the loss function to be back propagated through.This gradient can be used to train the alignment neural net as well as the whole translation model jointly.

Does attention really work results of Bahdanau et al.

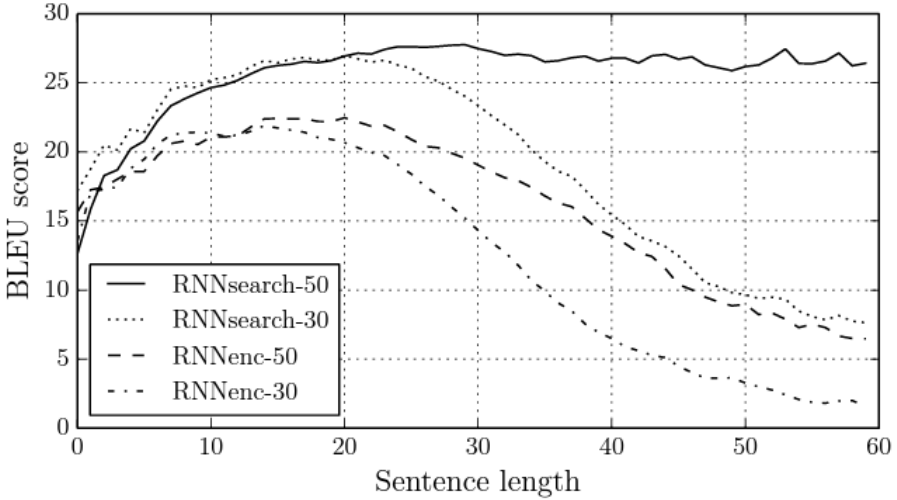
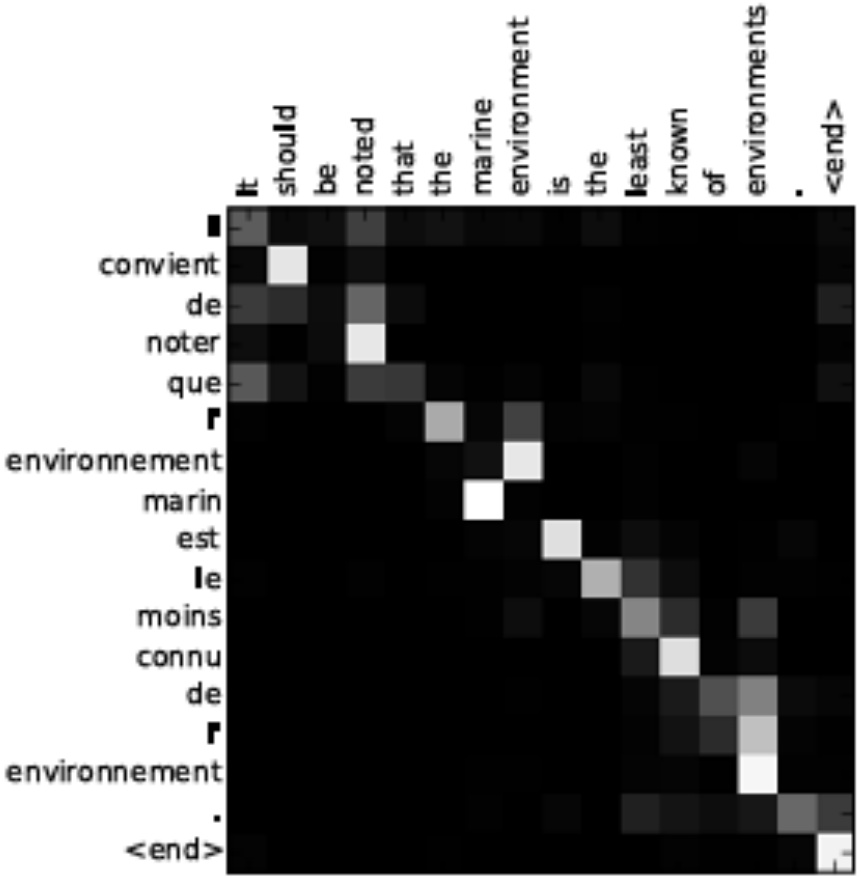


Figure 2: The BLEU scores of the generated translations on the test set with respect to the lengths of the sentences. The results are on the full test set which includes sentences having unknown words to the models.

The model introduced by Bahdanau et al. is called RNN search 50 and RNN search 30 which have been trained on sentence of length 50 and 30 respectively . Here we can see that many models show diminishing performance as sentence length increases while RNN search 50 plateaus which is a big improvement o the pre-existing models



Sample alignments found by RNN search-50.

The x-axis and y-axis of each plot correspond to the words in the source sentence(English)and generated translation(French), respectively.

Each pixel shows the weight αij of the annotation of the j-th source word for the i-th target word in grayscale (0: black,1:white).

Luong Attention

The second type of Attention was proposed by Thang Luong in this paper[3]. It is often referred to as Multiplicative Attention and was built on top of the Attention mechanism proposed by Bahdanau. The two main differences between Luong Attention and Bahdanau Attention are:

1. The way that the alignment score is calculated.
2. The procedure for calculating α_{ij} value .
3. The position at which the Attention mechanism is being introduced in the decoder, instead of being added to the LSTM cell the C_i value is concatenated to the output.

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \quad \text{where,} \quad e_{ij} = a(s_i \quad , h_j) \quad \text{Here a is a function}$$

Here α_{ij} is calculated based on the current decoder hidden state value the authors argue that the current hidden state contains the most recent information and thus must be used

Luong Attention

In Luong Attention, there are three different ways that the alignment scoring function is defined- dot, general and concat. These scoring functions make use of the encoder outputs and the decoder hidden state produced in the previous step to calculate the alignment scores.

- **Dot:** The first one is the dot scoring function. This is the simplest of the functions; to produce the alignment score we only need to take the hidden states of the encoder and multiply them by the hidden state of the decoder.

$$e_{ij} = S_i^T \cdot h_j$$

- **General:** The second type is called general and is similar to the dot function, except that a weight matrix is added into the equation as well.

$$e_{ij} = S_i^T \cdot (W * h_j)$$

Luong Attention

- **Concat:** The last function is slightly similar to the way that alignment scores are calculated in Bahdanau Attention, whereby the decoder hidden state is added to the encoder hidden states. However, the difference lies in the fact that the decoder hidden state and encoder hidden states are concatenated together first before being passed through a Linear layer.

  This means that the decoder hidden state and encoder hidden state will not have their individual weight matrix, but a shared one instead, unlike in Bahdanau Attention. After being passed through the Linear layer, a *tanh* activation function will be applied on the output before being multiplied by a weight matrix to produce the alignment score.

$$score_{alignment} = W \cdot tanh(W_{combined}(H_{encoder} + H_{decoder}))$$

Luong attention strives to reduce the dependence on neural nets while calculating similarity as the authors argue that adding a neural net increases the number of parameters which in turn makes the model bulky and slow.

References

[1]D. Bahdanau, K. Cho, and Y. Bengio, ‘Neural Machine Translation by Jointly Learning to Align and Translate’, *arXiv [cs.CL]*. 2016.
[2]https://www.youtube.com/watch?v=rj5V6q6-XUM&list=PLKnIA16_RmvYuZauWaPlRTC54KxSNLtNn&index=69
[3] M.-T. Luong, H. Pham, and C. D. Manning, ‘Effective Approaches to Attention-based Neural Machine Translation’, *arXiv [cs.CL]*. 2015.
[4]https://machinelearningmastery.com/the-luong-attention-mechanism/

UE21CS343BB2

Topics in Deep Learning

Dr. Shylaja S S
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
shylaja.sharath@pes.edu

Ack:Anashua Dastidar, Teaching Assistant