Alexandre Zajac
zajac.alexandre@gmail.com

Lab session # 2
ALTEGRAD 2020

11/28/20

1 Question 1

In the lab, we employed a greedy decoding strategy for decoding the translations. Indeed, we use the the index of the word with the highest probability after the GRU decoding layer at time t as input for the decoding layer at time $t+1$, until the EOS token:

$$\tilde{x_t} = \arg\max_x log\ p(x|x_{<t}, Y) \tag{1}$$

It's computationally efficient but has the drawback of affecting a lot the dependence between predicted words. If the first word is poorly detected, the next prediction will be greatly influenced.

One way to overcome this would be to use teacher forcing, where we use the next word in the correct output sentence as target instead of the newly predicted word. Here we might also over-fit and fail to capture complex language patterns, so we could do a kind of weighted teaching forcing, where we begin the training with teacher forcing and after a certain loss/epoch threshold ϵ, we use the predicted words as input for each new decoder layer.

Another technique that could be employed is to use the Beam search decoding strategy, mentioned in the ACL tutorial, which considers K paths with the top-k predicted words at each time-step and expands them if there are interesting compared to the correct output prediction. A fine-tuning on K must be used to achieve good results.

One could also consider ancestral sampling also mentioned in the ACL tutorial, which samples a word from the probability distribution given by the $softmax$ layer used to predict the decoder's current output. This technique can generate more natural sequences.

2 Question 2

Here are some translations generated after training the model and using $is_{prod} = True$:

- I did not mean to hurt you $->$ je n ai pas voulu intention de blesser blesser blesser blesser blesser blesser . blesser . blesser
- I can't help but smoking weed $->$ je ne peux pas empêcher de de fumer fumer fumer fumer fumer fumer fumer fumer fumer urgence urgence urgence urgence urgence urgence . urgence urgence . urgence urgence .
- The kids were playing hide and seek $->$ les enfants jouent cache cache cache cache caché caché caché caché caché caché caché caché caché caché caché caché caché caché caché dentifrice perdre caché risques rapide caché risques éveillés

We can see that the major problem is that the translated sentence presents repetitions, especially with the words near the end of the sentence.

To solve this, one solution would be to have a system for penalizing the word repetition, and we could make this system even have a larger penalization when repeating the word a 3rd time, 4th time, etc... This is similar in essence to what is suggested in [4], with the idea of a context vector. A set $x = \{x_1, x_2, ..., x_n\}$ of input words and a coverage set $C = \{0, 0, ..., 0\}$ is maintained to keep track of which words where translated. A coverage model can be formed where it will discourage attention to already translated words, and could solve the problem we are facing here.

One more robust method would be to use a technique suggested in [2]: "local attention". It requires to learn attention weights and the context vector not on the entire set of previous encoder outputs, but on a window of size D and select a position p_t to pay attention to words in the windows $[p_t - D, p_t + D]$ only. Since French

and English languages are different in structure we can't really assume that a position t of a word in the input is the same as the in the output. We might thus use "predictive alignment" where we have:

$$p_t = T_x * sigmoid(v_p^T * tanh(W_p h_t)) \tag{2}$$

where T_x is the size of the input sequence, and the value in the sigmoid is the score we calculated for the alignment vector $\alpha_{t,i}$.

3 Question 3

After modifying the existing code to add the generation of the alignements visualizations, here are some examples:

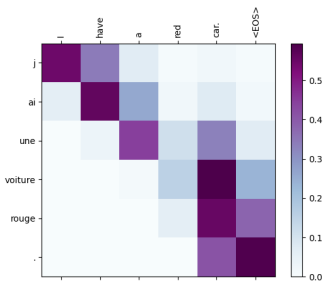

Figure 1: Attention weights for 'I have a red car.'

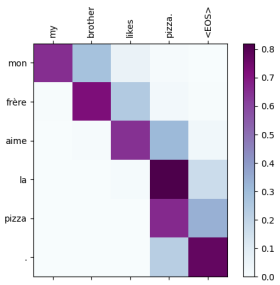

Figure 2: Attention weights for 'This river is full of fish.'

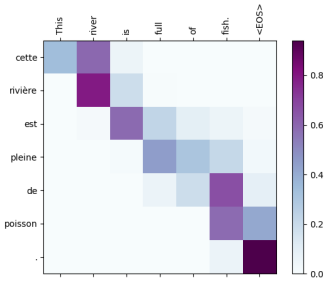

Figure 3: Attention weights for 'my brother likes pizza.'

To achieve this, a tensor of the attention scores from the attention mechanism in the seq2seqAtt module was returned, as well as a tensor of the overall attention from the forward() method in the seq2seqModel. We then implemented a custom $visualize\_attention(input\_sentence, output\_sentence, attention\_weights)$ method. Note that all $attention\_weights$ are cut to their limits in input ('EOS' token) and output sentences ('.' token).

We can see that the network successfully understood the adjective-noun inversion for the car example 1, because it gave more attention to "voiture" for "red" than "rouge", meaning it understood that is was actually the car that was red. We can see for 2 that there was a lot of attention for "la" and "pizza", meaning the network understood the semantic relation between the pronoun and the noun. It's the same phenomenon for 3, where there is more attention for the pronoun referring to "fish" rather than to "poisson" itself.

4 Question 4

Here are the visualizations and translations of attentions for both of the sentences:

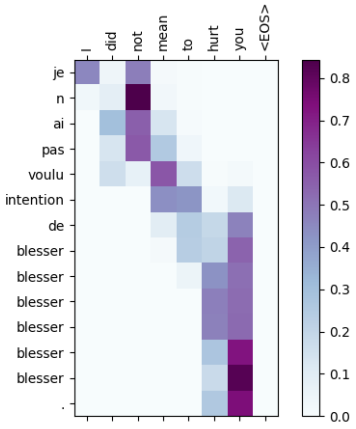

Figure 4: Attention weights for 'I did not mean to hurt you.'

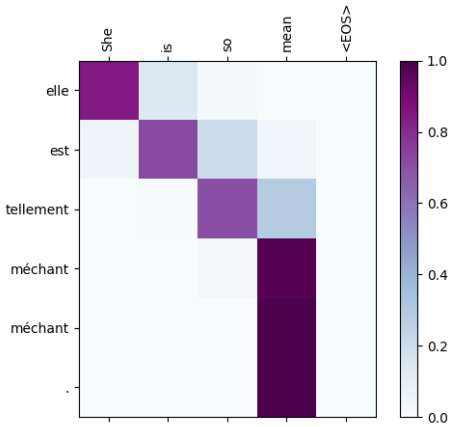

Figure 5: Attention weights for 'She is so mean.'

- I did not mean to hurt you $->$ je n ai pas voulu intention de blesser blesser blesser blesser blesser blesser . blesser . blesser
- She is so mean $->$ elle est tellement méchant méchant . EOS

We can see here that the model successfully understood the difference between the adjective/verb alignment of ”mean” in terms of the context, even if there is a difference between the French and English syntax.

However, for more complex semantics and contexts of sentences, our model might have limitations to identify the difference between adjective/verbs. To solve this issue, one option to consider would be to use an architecture based on [1], with transformers, that would train to obtain ”deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers”.

Another option would be to keep the sequential-like architecture (RNN, LSTM or GRU) but have a bidirectional language model [3] so that we have both:

$$p(t_1, t_2, ..., t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2, ..., t_{k-1}) \tag{3}$$

$$p(t_1, t_2, ..., t_N) = \prod_{k=1}^{N} p(t_k | t_{k+1}, t_{k+2}, ..., t_N) \tag{4}$$

This way, the network would be able to learn and infer from both directions and it would greatly help if the semantics and the context of certain entities are more difficult to dissect.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[3] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[4] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*, 2016.