BENYAMINA Elyas
elyas.benyamina@ens-paris-saclay.fr

Question 1

The presentation mentioned will help us answer this question.
We use a **greedy strategy**, which lacks efficiency but is inexpensive in terms of resources and time.
The output is computed word by word, and the word chosen as each step is the most probable word in the vocabulary. The word t in the sentence is chosen so as to maximize the probability, which is approximated by the GRU neural network $\tilde{x}_t = \arg\max_x \log p(x \mid x_{<t}, Y)$. The neural network takes as input the last prediction (and also a vector h, which retains some information from previous inputs) and returns the probabilities for the next word, selecting the word with the highest probability. This continues until the EOS token is encountered. The translation done in this way is not concerned with sentence coherence, and does not allow to correct any grammatical errors afterwards.

For a more efficient translation, we can use **beam research**. The output sentence is still generated word by word: We select the k (for k a chosen integer) most probable words, and for each of them, we add each word of the vocabulary (of cardinal V), so as to obtain for each k word, V sentences (of two words), i.e. $k \cdot V$ sentences in total. We then keep only the k most probable sentences (i.e. the $\tilde{X}$ sentences that maximize $\log p(\tilde{X} \mid Y)$), and we repeat the process, adding to each of the k saved sentences the V words from the vocabulary, to obtain $k \cdot V$ sentences again, from which we'll retain only the k most probable, and so on. This translation extends the greedy strategy (it's the same for $k = 1$), but produce more coherent and exact translations as it's comparing different sentences at each step and keep the best one (by comparing them to the sentence we want to translate).
This high-performance strategy is very costly when k is large, and is difficult to parallelize.

We should also mention the **ancestral sampling** method, which randomly generates words according to the probability vector returned by the softmax layer. However, this method suffers from high variance and is inefficient.

Question 2

The main problem is repetition in sentences. The same words appear several times. Sentences in which words are repeated are mostly those without a dot. Our strategy doesn't know when to end the sentence.

Here are some translations that illustrate this :

1. I love playing video games. -> j adore jouer à jeux jeux jeux vidéo . . . . . .
2. I did not mean to hurt you -> je n ai pas voulu intention de blesser blesser blesser blesser blesser blesser . blesser . blesser . . . . . .
3. I can't help but smoking weed -> je ne peux pas empêcher de de fumer fumer fumer fumer fumer fumer fumer fumer fumer fumer urgence urgence urgence urgence urgence urgence . urgence urgence . urgence urgence .
4. The cat fell asleep in front of the fireplace -> le chat s est en du du pression peigne peigne cheminée portail portail portail portail portail portail portail portail indépendant oiseaux oiseaux oiseaux oiseaux oiseaux oiseaux oiseaux oiseaux oiseaux oiseaux

Possible hypothesis for the last two sentences is that the model is not sufficiently trained, making it impossible to obtain good translations, or that the sentences are too complex for our model.

Here are two methods to help the model translate better and avoid the overtranslation problem (and thus stop sentences at the right moment).

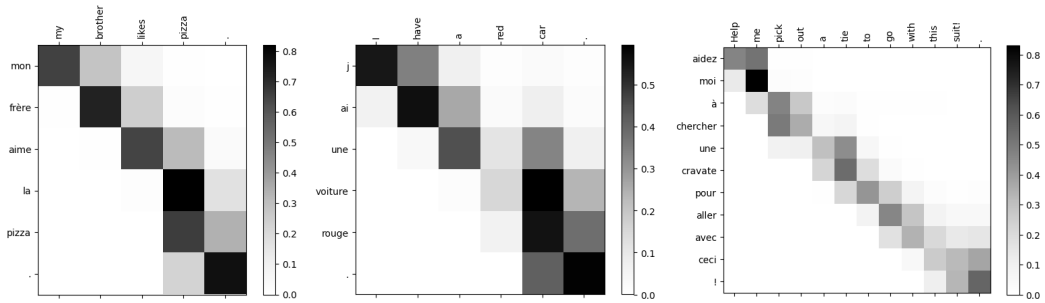
1. **Coverage model.** One method proposed in [4] is to use a coverage vector that indicates which words have already been translated, so that the neural network is less likely to take them into account. This method is called coverage model. A word can be required to be used only once in the translation (hard coverage). But for NMT, it's often better to use a "continuous" coverage can be used, which discourages

(but doesn't forbid) the use of words that have already been used. In particular, this coverage can be included in the calculation of the context vector, and thus influence the translation of subsequent words to limit the use of words already translated and thus limit the problem of overtranslation.
The problem with this method is that it favors the translation of each individual word, which is sometimes irrelevant, especially when translating long sentences.

2. **Local attention method.** One idea is to use the local attention method mentioned in [3], which generates at each step t, the context vector from the words $[p_t - D, p_t + D]$ (where D is chosen empirically according to observed performance) of the original sentence (where p_t can be chosen equal to t if we assume that the words in the translated sentence will have a similar position to those in the original sentence, or we can use a neural network to generate this if this is not the case).

Question 3

We modify our code to return the $norm\_scores$ variable as output from the forward function of the $seq2seq$ class. We then use matplotlib to model the links between the translated words and the source words (we'll restrict some translations for the sake of visualization).
Our algorithm takes into account noun-adjective inversion. For example, the most frequently used word to obtain "voiture" in output is "car" and to obtain rouge is "red", with the intervention of voiture (which can be explained by the fact that to spell the adjective correctly, we need the noun).
It also knows that in french noun comes before adjective, and that prepositions depend on nouns, and we can see the strong use of nouns when he translates prepositions (such as "la", which uses the word "pizza" in the first plot, "une" which uses "a" and "car" in the second, and "une" which uses "a" and "tie" in the third). Here are some relevant plots.



(a) (b) (c)

Figure 1: Examples of source/target alignments

Question 4

Here are the translations obtained:

- 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 notice that there are several possible uses for the word "mean", and that it can occupy several different forms (adjective or noun). The model language must take into account the context in which the word is used. The words that precede and follow the word we want to translate must therefore be used to produce a relevant translation. Our model succeeds in translating this word well in the two different contexts, but the quality of the overall translation is not optimal.

[2] and [1] mention a bidirectional approach.
Indeed, the approaches presented above perform word-by-word translation using only past words. This does not take into account the end of the sentence to be translated.
The bidirectional mentionned in [2] approach uses a second neural network (GRU, LSTM or other) similar to the first, which processes the words of the sentence in reverse order (so for each word translated, the end of the sentence is used, not the beginning) and combines the outputs of the two approaches to obtain a

translation that takes into account the beginning and end of the sentence.
The use of a bidirectional model could be expected to improve sentence quality.
Note that the BERT model in [1] uses transformers which take the set of words in the sentence to be translated, and associate a probability vector (corresponding to the meaning of the word) with each word. The encoding of each word uses all the words in the sentence.

References

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

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

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

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