1 Question 1

The answer to this question is based on [5].
Greedy search is a computationally cheap version of decoding, because it assumes selecting the output with the highest probability after the softmax. Despite its low computational cost, this method yields outputs which lack in variety and diversity. This means that some specific constructions which are seen very frequently in the language will be chosen over the choices which are less frequent. Therefore, after the translation, we will end up obtaining very similar phrases which consists of very common words. More precisely, by choosing the outputs with the highest probability after the softmax (greedily), we don't necessarily choose the most probable sequence overall. For example, suppose we wanted to output a sentence of nine words. By choosing the most probable softmax output at the second timestep, we might end up with a sentence with lower probability than that of the most likely sentence of nine words. Furthermore, since this greedy procedure is deterministic, we don't get sample diversity (no matter how many times we repeat this procedure, we always get the same outputs).
One strategy that can be used to address the disadvantages of greedy search is beam search, which has an increased computational cost, but can be more effective at outputting sentences with potentially higher total probability. As presented at slide 94 in [5], beam search assumes that we keep an ensemble of K hypotheses at time t, $H_t = \{(x_1^1, x_2^1, ..., x_t^1), (x_1^2, x_2^2, ..., x_t^2), (x_1^K, x_2^K, ..., x_t^K)\}$, where $(x_1^k, x_2^k, ..., x_t^k)$ is the k-th hypothesis of H_t. K is a hyper-parameter denoted as the beam width, which we are free to choose (by e.g. cross-validation). At time $t+1$, we consider each potential continuation for each of the K hypothesis. For example, $\{(x_1^k, x_2^k, ..., x_t^k, v_1), (x_1^k, x_2^k, ..., x_t^k, v_2), (x_1^k, x_2^k, ..., x_t^k, v_{|V|})\}$ are all the possible continuations for the k-th hypothesis, and $\{v_1, v_2, ..., v_{|V|}\}$ are the words of the vocabulary. After this step, the probabilities of the expanded hypotheses are computed (by using Bayes' rule) and only the top K hypotheses with the highest probabilities are kept. This procedure is repeated recursively, until e.g. the end of sequence (EOS) special character is predicted (or, for example, until a maximum sentence length is reached). At the end, the sentence with the highest likelihood is usually chosen. It is easy to observe that this procedure is computationally more expensive, but the output sentence can have higher likelihood (than in the case of greedy search), as beam search is able to explore and analyse more possible outputs. The beam search algorithm provides a good trade-off between the greedy decoding strategy (not very effective, but it requires little computation) and exhaustive search (which provides the most likely outcome with certainty, but with a very high computational cost). By setting the beam width equal to 1, the beam search algorithm is equivalent to the greedy search algorithm and by making it asymptotically large, we will find the exhaustive search method (e.g. for sentences of length t and beam search $|V|^t$, we recover exhaustive search). If we set the beam width somewhere in between these values, we may come up with a very good decoding strategy. Also, by sampling from the K final hypotheses (with e.g. a uniform distribution) rather than choosing the one with the highest probability, we can obtain more diverse samples.

2 Question 2

The answer of this question is based on the references [3] and [6].
The main issue with the obtained translations is that some words are translated multiple times (the over-translation problem), while other words are not translated at all (under-translation). For example, when translating the sentence "I am a busy student", the model outputs "je suis un occupée", so the word "student" is mistakenly missing from the translation. On the other hand, when translating the example "I love playing video games", the result is "j adore jouer à jeux jeux jeux vidéo", so the word "games" is translated multiple times. This may seem very contradictory, but both problems actually have the same source: the attention model used in the code does not keep track of past alignment information. This is a very important aspect, because keeping track of past alignments is necessary when we want the model to keep track of the source words that have already been translated. This can be improved by having our model use a history of previous alignments. A method for resolving this issue is proposed in [6], where a coverage vector is used to keep track of the attention history. This vector is computed using the alignment weights. By using this coverage vector, the decoder knows whether a specific word has already been translated or not. In the paper, the authors confirmed that this approach helped significantly in solving the over-translation and under-translation issues. [3] proposes a similar approach, denoted as the input-feeding approach, in which the attentional vectors (obtained by weighing the encoder's outputs with the alignments weights) are used.

3 Question 3

Four alignment visualizations of the attention weights are presented in Figures 1, 2, 3, 4. The model of the visualizations is adapted from [1]. As we can observe from the images, the noun-adjective inversions are not well represented in the visualisations, even when the translations do contain the noun-adjective inversion (like in Figure 3). As we can see from Figure 3, the attention weight corresponding to the translation 'red - rouge' is lower than the attention weight corresponding to the translation 'car - rouge'. The translation is correct with regard to the noun-adjective inversion, because the adjective is placed after the noun in French ("red car" is translated as "voiture rouge"), even if in the source sentence the grammar rule is different (in English grammar, the adjective is placed before the noun). The visualization weights are also different from what we would expect in Figure 1, where the attention weight corresponding to the translation 'busy - occupée' is lower than the weight corresponding to 'student - occupée'. However, in this case, the translation as a whole is wrong ('student' is not translated and the gender of 'un' is different from the gender of 'occupée').

4 Question 4

The answer of this question is based on [4] and [2].
The translation of the sentence "I did not mean to hurt you" is "je n ai pas voulu intention de blesser blesser blesser blesser blesser blesser . blesser . blesser . . . . . .", while the translation of the sentence "She is so mean" is " elle est tellement méchant méchant .". We can observe from these translations that a specific word - "mean" -, which can have multiple meanings in different English contexts, appears with the same form in both of the source sentences and it is translated differently. Therefore, by inspecting these translations, we can say that the model is able to capture the context of the word and translate it accordingly to the meaning it might have in that specific context. In addition, the word "mean" has different syntactic functions (it can represent different parts of speech) in the two sentences. In the first phrase, it is a verb, while in the second one, it is an adjective. In other words, the language model can properly deal with complex characteristic of words, such as syntax and the polysemy property. It means that the model does not analyse only the appearance of words, it also captures context-dependent details about the semantic and the syntax of the words. Word-sense disambiguation can be a very important property of a machine translation model in order to provide high-quality results. Although [4] and [2] use bidirectional language models (bidirectional LSTMs and bidirectional Transformers, respectively), the left-to-right GRU unidirectional language model used in our code seems sufficient to capture some syntactic and semantic word properties.

References

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

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

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

[4] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. Deep contextualized word representations. *ArXiv*, abs/1802.05365, 2018.

[5] Christopher Manning Thang Luong, Kyunghyun Cho. Neural Machine Translation - ACL 2016 Tutorial. `https://nlp.stanford.edu/projects/nmt/Luong-Cho-Manning-NMT-ACL2016-v4.pdf`, 2016. [Online].

[6] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Coverage-based neural machine translation. *CoRR*, abs/1601.04811, 2016.

A Figures

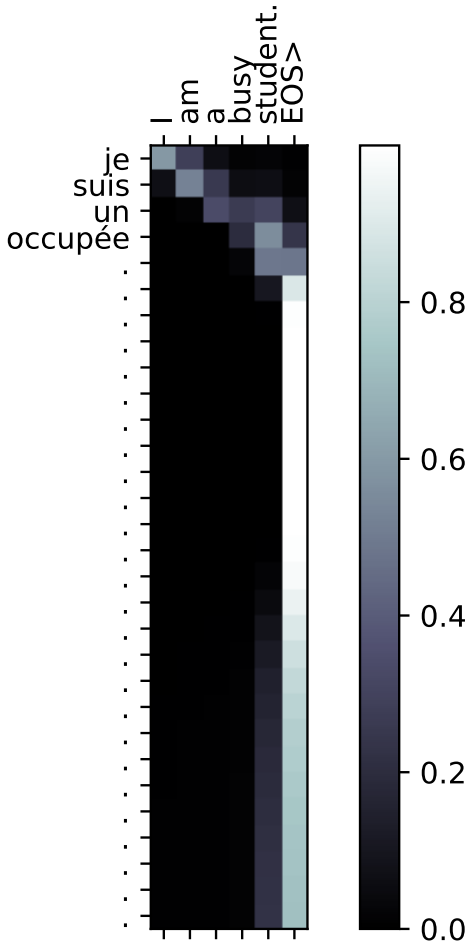

Figure 1: Alignment visualizations of the attention weights for the sentence ”I am a busy student.”.

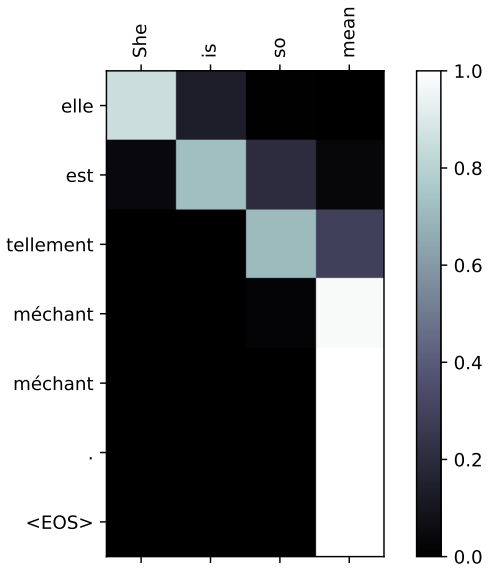

Figure 2: Alignment visualizations of the attention weights for the sentence ”She is so mean”.

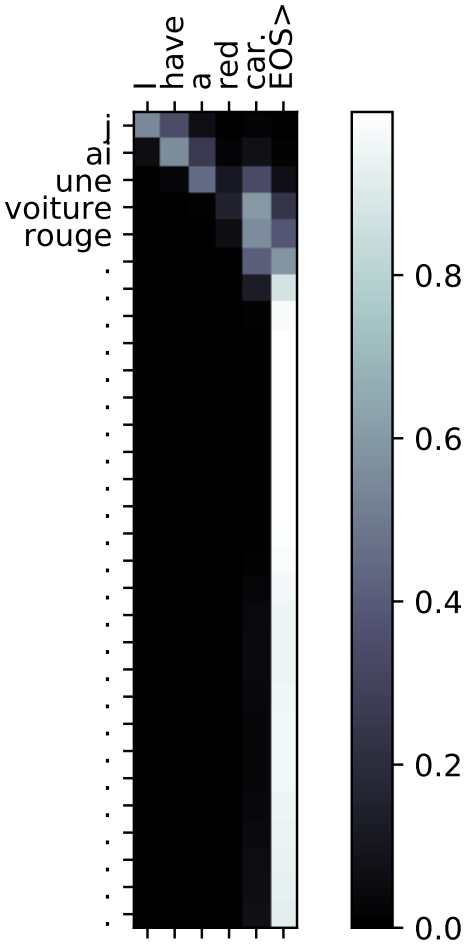

Figure 3: Alignment visualizations of the attention weights for the sentence ”I have a red car.”.

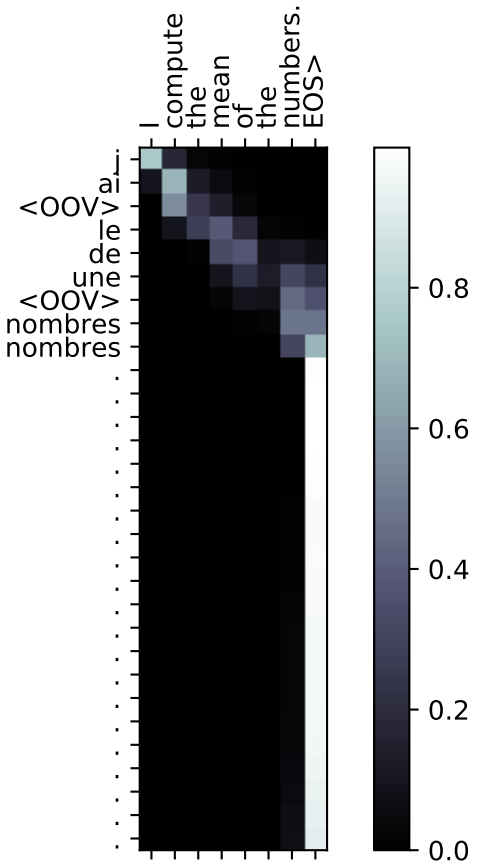

Figure 4: Alignment visualizations of the attention weights for the sentence ”I compute the mean of the numbers.”.