CS291 Structured Prediction in NLP — Project1 Machine Translation

Sourabh Raja Murali
UC San Diego
srajamurali@ucsd.edu

May 1, 2022

1 Problem Description

In this paper we discuss about the machine translation problem by training a seq2seq architecture with an attention mechanism. The dataset we will be using for the experimentation is Multi30k machine translation dataset.
We will look at a bunch of decoding strategies like greedy decoding, beam search decoding and nucleus sampling and compare and contrast these methods to look at how they perform in comparison.

2 Task 1 Seq2Seq model

The task of machine translation in our case from German to English is achieved through seq2seq model and the model and its implementation is defined in the source code that I have submitted. We train the model on the Multi30k dataset as mentioned earlier. We run the model for 10 epochs and we get the loss curve for the training and validation as seen below in the figure1.

We can infer from the graph that the training loss slowly decreases with the increasing epochs. However the validation loss starts reducing before flattening. I have used the trained model to translate sample sentences from the validation set into English. Two such sentences where the source sentence is in German, target sentence is in English and the translated sentence which is the model prediction are shown here.
Example1
src: ”ein schwarzer hund und ein gefleckter hund kämpfen .”
trg: ”a black dog and a spotted dog are fighting ”
model prediction: ”a black dog and. a spotted dog fighting .”

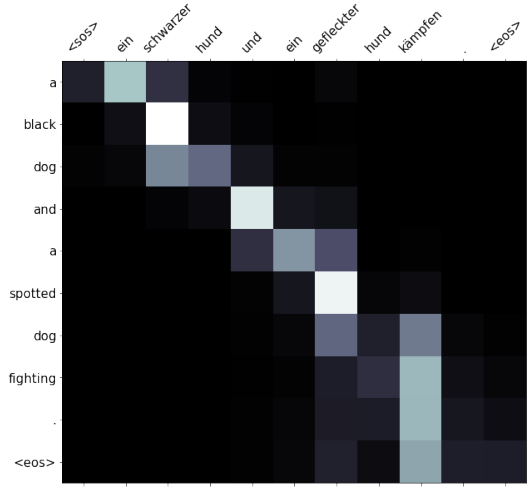

Figure 2: Attention visualization for each trg token

Example2
src: ”eine frau spielt ein lied auf ihrer geige .”
trg: ”a female playing a song on her violin .”
model prediction: ”a woman is a song on her violin .”

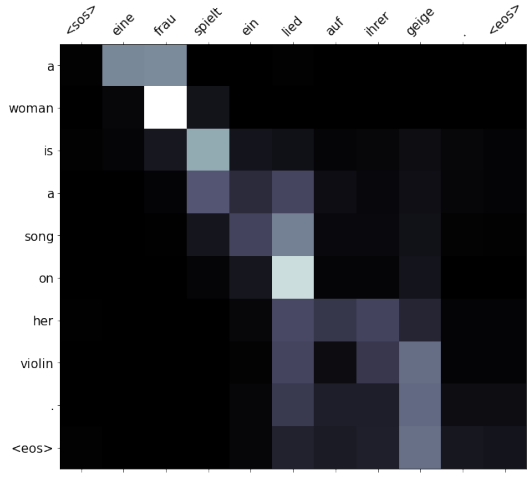

Figure 3: Attention visualization for each trg token

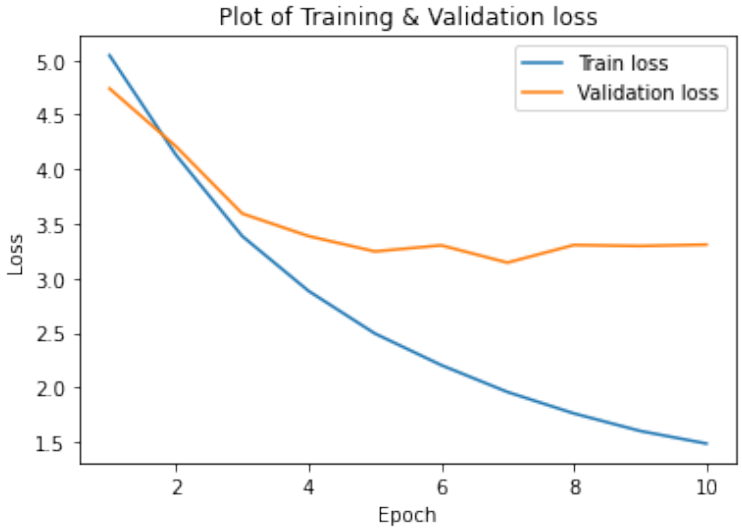

Figure 1: Training & Validation Loss vs Epochs

When we look at the heat map of attention of source sentence to each token in the target sentence we can evidently make out when words related are translated to target tokens. Like for example black dog is influencing schwarzer hund and so on. If I knew the German language I could have made much more sense out of the heat map however my knowledge is nil. When we will look at these model predictions we can make out that the model is doing a pretty good job translating sentences from German to English. However there would be cases where the translation might not make sense or is slightly off the track. The test loss for the model that we trained is **3.158** and the test perplexity turned out to be **23.52**. The bleu score using the greedy sampling method was **29.49**.

2.1 Task 3: Beam Search

As we have seen the seq2seq model leverages an encoder and decoder framework with LSTM or GRU as the basic building blocks. Encoder map a source sequence, encodes the source information and passes it to the decoder. The decoder takes the encoded data from the encoder as input and along with the ¡sos¿ token as the initial input and produces an output sequence. We do not want any random translation but the best and most likely words to be chosen for the translation into the target language. In order to select the most likely words like we saw the most easiest approach is the greedy approach where we pick the maximum probability of the word that is spit out by the decoder at each time step. Unlike greedy, beam search algorithm selects multiple alternatives from the input sequence at each time step based on the conditional probability. The count of this multiple alternatives depends on the beam width (k) that is passed as a parameter to the beam search function. At each time step, the algorithm selects k number of alternatives with the highest probabilities as the most likely probable choices for the time step. We also prune the branches of the beam search if we encounter end of sentence token as the most probable one at that time step. Therefore beam search considers multiple best options based on the beam width using conditional probability, which is better than the sub-optimal Greedy search. I have implemented this greedy search algorithm in PyTorch which can be seen in the notebook attached. I am maintaining a data structure which captures the log probability, hidden state output from the decoder and the indices list of target token till that time step. With the for loop we loop through each of this list data structure and take the top k probable sequences and branch from there. Finally we would end up in k sequences which have the highest probability values among all the possible sequences. It runs fairly quickly as I have used mostly PyTorch functions for faster computations and avoiding loops as much as possible. The samples generated by this method with beam size set to 5 is shown below and is compared to the ones obtained from greedy method. In my observation for some samples beam search gave good variations in the translations but

for few cases the other candidates from he beam search contain repetition in words. Following are 3 different examples picked from the test dataset.

Example1:
src: "die person im gestreiften shirt klettert auf einen berg ."
trg: "the person in the striped shirt is mountain climbing. "
greedy strategy translation: "the person in the striped shirt is climbing a mountain."
beam search translations:
1. the person in the striped shirt is climbing a mountain.
2. the person in the striped shirt is climbing on a mountain.
3. the person in the striped shirt is rock climbing.
4. the person in the striped shirt is rock climbing a mountain.
5. the person in the striped shirt is climbing up a mountain.

Example2:
src: "ein mann schneidet äste von bäumen ."
trg: "a man cutting branches of trees "
greedy strategy translation: "a man is chopping some trees."
beam search translations:
1. a man is chopping some trees.
2. a man is chopping some trees trees.
3. a man is chopping some some trees.
4. a man is chopping bubbles surrounded by trees.
5. a man is chopping some debris.

Example3:
src: "vier weiße hunde mit maulkörben springen über eine rote wand."
trg: "four white dogs with muzzles jump over a red wall. "
greedy strategy translation: "four white dogs with muzzles jumping over a red wall."
beam search translations:
1. four white dogs with muzzles jumping over a red wall.
2. four white dogs dogs jumping over a red wall.
3. four white dogs wearing muzzles jumping over a red wall.
4. four white dogs with muzzles over a red wall.
5. four white dogs with muzzles over over a red wall.

When we look at these sample translations we can see a diversity in most of the cases in translation from source to target. In some cases we can also see that the candidate output from greedy is also one among the beam search output translations. But that does not have to be the case always. In some cases we see repeated words whereas in some cases we see alternate meanings of the same words which is a good thing. There are also cases where there is a totally different meaning being conveyed in the beam search output sentence. This is the qualitative analysis of beam search compared to the greedy search.

Now lets look at the bleu scores on the test data. As mentioned earlier the bleu score using greedy strategy on the test data is 29.49. With the beam search strategy when the beam size is 1 I get the same bleu score of 29.49 which indicates that greedy search is a special case of beam search with beam size 1. We can see the variation of bleu score for different beam sizes from the below table and graph. I have just multiplied the bleu score by 100 to depict on the graph.

beam size	Bleu scores on the test data
1. 1	29.49
2. 3	30.81
3. 5	30.62
4. 7	30.60
5. 10	30.40

Table 1: Blue score for different beam sizes

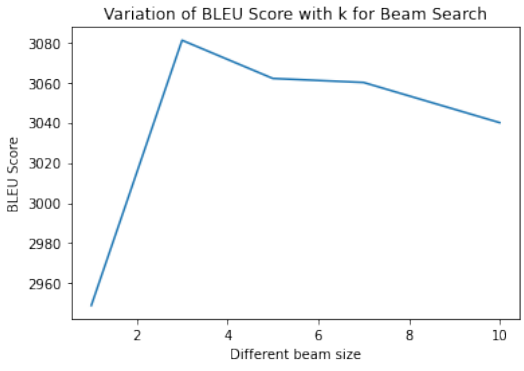

Figure 4: bleu score vs beam size

The reason for decrease in the bleu score for higher beam sizes could be that with higher beam sizes the chances of a diverse word getting picked which qualifies the top probability is more and due to this diverse set of translation words, there could be mismatches in the ngram count which the bleu

score uses and hence we might be getting a dip. As we go to higher beam sizes the bleu score might decrease and it might give better diverse translations however it will have higher computational cost.

3 Nucleus Sampling

In this method of decoding, instead of looking at just the maximum probability token, nucleus sampling focuses on the smallest possible sets of top-V words so that the sum of their probabilities are greater than or equal to p. The tokens that do not fall in this set are dropped by masking their indices and the rest are re-scaled just to ensure that they sum up to 1. Thereby the size of the set of words can dynamically increase or decrease as per the next word's probability distribution. The best part here is that with this method we not only get diverse word choices through the stochasticity of sampling, but also preserves the quality by focusing only on the top-p mass. The samples generated by nucleus sampling with p set to 0.9 are shown here and compared with the greedy search result. I referred to [1] for implementing the nucleus sampling method.

Example1
src: "eine frau spielt ein lied auf ihrer geige ."
trg: "a female playing a song on her violin "
Greedy search result: "a woman is a song on her violin ."
Nucleus Sampling result: "a woman fiddles is a song on her violin on her concert. "

Example2
src: "drei kleine kinder stehen um ein blau-weißes fass herum ."
trg: "three young children stand around a blue and white barrel "
Greedy search result: "three small children standing around a a blue elephant."
Nucleus Sampling result: "hree small children standing around a a yellow hallway. "

By observing the translations from the nucleus sampling method what we can observe is that even though sometimes we do not get nearest translation as that of greedy search we get a more diverse words and sentences which might not be the exact translation of the source sentence. Now let us look at the bleu score variation for different values of top p values. This is summarized in the below table and depicted on the below graph.

top p value	Bleu scores on the test data
1. 0.4	27.85
2. 0.5	26.84
3. 0.6	25.10
4. 0.7	23.65
5. 0.8	20.75
6. 0.9	18.64

Table 2: Blue score for different top p values

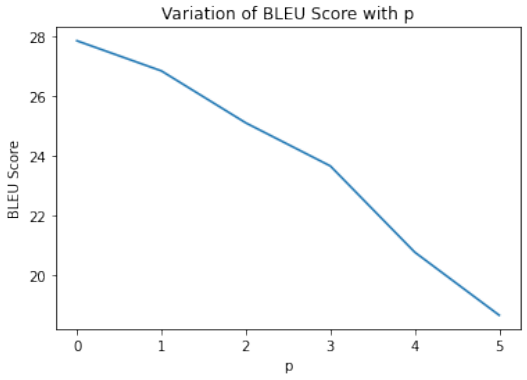

Figure 5: bleu score vs top p value

We can observe the variation of bleu score as we increase the top p value which is a parameter for the nucleus sampling function, the bleu score decreases and it gives a bleu score of 18.64 for the top p value of 0.9. The reason for this could be that bleu score involves evaluating how well the ngrams are matching between the two. But like I mentioned in nucleus sampling we are looking at diverse sentences hence there might not be an exact match in the ngrams. However for the lower values of top p, it mostly behaves like closer to greedy search as it has lesser number of high probability words to pick from. Hence the chances of picking the actual translation word is higher, therefore chances of ngrams match is higher and thereby might be getting a higher bleu score at lower values of top p.

4 Optional Tasks

I have attempted the task of training the seq2seq model on another machine translation dataset for the language pair Spanish, English. The dataset has been downloaded from the following URL "http://storage.googleapis.com/download.tensorflow.org/data". I

have used the SpaCy's Spanish tokenizer to tokenize the sentences and split the data into train, test and validation and following a similar pipeline using PyTorch's TabularDatasets. I have changed the dropout value from 0.5 to 0.6 and all other model parameters have been retained the same. The training loss and validation loss can be seen from the figure 6

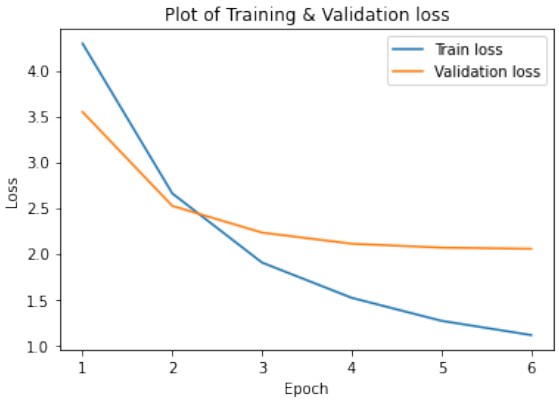

Figure 6: Training & Validation Loss vs Epochs for Spanish to Eng

The model has been trained for 6 epochs. The test loss that I obtained was 2.03 and test perplexity was 7.61.

I have tried to decode from Spanish to English using both greedy search and beam search method. Below are some sample translations. The bleu score from the greedy strategy on test data was 43.03.

Example1:
src: "¿ por qué tom querría lastimar a maría ?"
trg: "why would tom want to hurt mary ? "
greedy strategy translation: "why would tom want to hurt mary ? "
beam search translations:
1. why would tom want to hurt mary ?
2. why would tom want to kill mary ?
3. why would tom want to hurt ? ?
4. why would tom want to wait?
5. why would tom want to drive mary ?

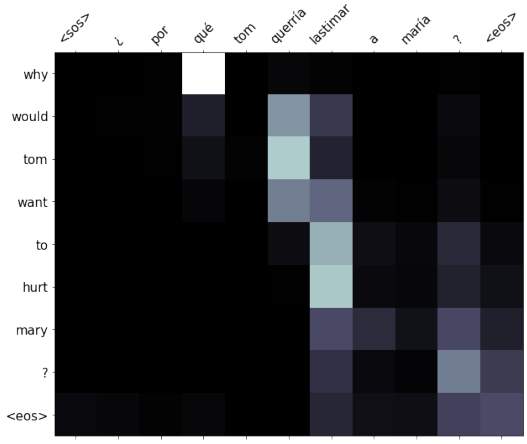

Figure 7: Attention visualization for Spanish translation

Example2:
src: "este sombrero es tuyo ."
trg: "this hat is yours . "
greedy strategy translation: "this hat is yours . "
beam search translations:
1. this hat is yours .
2. this hat is mine .
3. this hat 's yours .
4. this hat is yours . .
5. that hat is yours .

I obtained a bleu score of 44.82 for a beam size of 5 on the test data.

References

[1] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.