1g) before encoding step, we padded sentences to make them have equal lengths by using special padding token which does not convey any meaning with it. The mask is vector to determine the padded location by setting the corresponding padding indices to 1 and non-padded to 0.

In step function these masks were used to set attention values corresponding to pads to -inf.

mask = [0 0 0 1 1 1]
et = [- - inf, -inf, -inf, -]
at = [- 0 , 0 , 0 , 0]

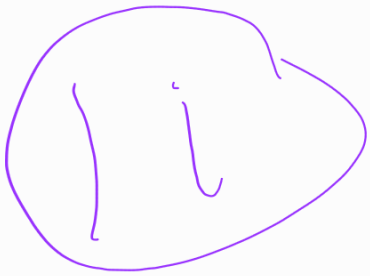
This approach will lead to 0 attention scores for those locations containing pads which means the model will not pay attention to them.

1g

1h

12.49

BLEU



Dot product VS Multiplicative
Advantage:
Dot product computation is faster than multplicative because it does not require to calculate extra weight matrix. (W in multiplicative is growing quadratic ally with tiime)

Dis:
Dot product uses the whole hidden states but we may not need to do it, multiplicative has the flexibility to control the usage of h by incorporating W.

Additive VS Multiplicative
It is more flexible than multiplicative by adding an extra W matrix to control S_t, it also incorporate non-linearaty (tanh) and it uses v vector which is going to add to the flexibility.
But it is more expensive to calculate since we have to compute extra W for S_t and also applying tanh.

r1: love can always find a way
r2: love makes anything possible
c1: the love can always do
c2: love can make anything possible

C_1 : a unigram

	Count$_p$ (r_1, r_2)	Count$_c$
the	(0, 0)	1
love	(1, 1)	1
can	(1, 0)	1
always	(1, 0)	1
do	(0, 0)	1

5 ⟹ $P = \frac{3}{5}$

3

C_1 : bigram

	Counter V_1	V_2	Count C
the love	0	0	1
love can	1	0	1
can always	1	0	1
always do	0	0	1

$\Rightarrow$ $P = \frac{2}{4}$

C_2 unigram make = makes

	Count$_r$	Count$_r$
love	(1, 1)	1
can	(1, 0)	1
make	(0, 1)	1
anything	(0, 1)	1
possible	(0, 1)	1

5

5

$P_1 = \frac{5}{5}$

C_2 bigram

	Count$_1$	Count$_2$
love Con	(1, -)	1
Can make	(0, 0)	1
make anything	(0, 1)	1
anything possible	(0, 1)	1
		4

$$\Rightarrow p_2 = \frac{3}{4}$$

$len(c_1) = 5$

$len(r_{c_1}) = 5$

$len(c_2) = 5$

$len(r_{c_2}) = 5$

anything → one token

$BP_{c_1} = 1$

$BP_{c_2} = 1$

BLUE$_{c_1}$

$$1 \times e^{\left(\frac{1}{2}\log\frac{3}{5} + \frac{1}{2}\log\frac{2}{4}\right)}$$

$$= 0.55$$

BLUE$_{c_2}$

$$1 \times e^{\left(\frac{1}{2}\log\frac{5}{5} + \frac{1}{2}\log\frac{3}{4}\right)}$$

$$= \boxed{0.87}$$

better interpretation

Yes I agree with bleu score because the second translation is very very similar to second reference but the first translation is not that similar and also we have all the words from second translation in references but not all words from first translation.

(ii)

C_1 to unigram

	Count p_1	Count c
the	0	1
love	1	1
can	1	1
always	1	1
do	0	1
	3	5

$$p_1 = \frac{3}{5}$$

C_1 : bigram

	Count(x)	Count(c)
the love	(o) . o	1
love can	(l) , l	1
can always	(l) , l	1
always do	(o) . o	1/4

2

$$P_2 = \frac{2}{4}$$

C_2 anagram make = makes

love	Counter	Counter
can	1	1
make	1	1
anything	0	1
possible	0	1
		5

2

$$P_1 = \frac{2}{5}$$

C_2 bigram

	Count$_1$	Count$_2$
love Con	1	1
Can make	0	1
make anything	0	1
anything possible	0	1

$$p_2 = 1/4$$

$len(c_1) = 5$

$len(r_{c_1}) = 5$

$len(c_2) = 5$

$len(r_{c_2}) = 5$

$BP_{c_1} = 1$

$BP_{c_2} = 1$

BLUE$_{c_1}$

$$1 \times e^{\left(\frac{1}{2}\log\frac{3}{5} + \frac{1}{2}\log\frac{2}{4}\right)}$$

$$= 0.55$$

better interpretation

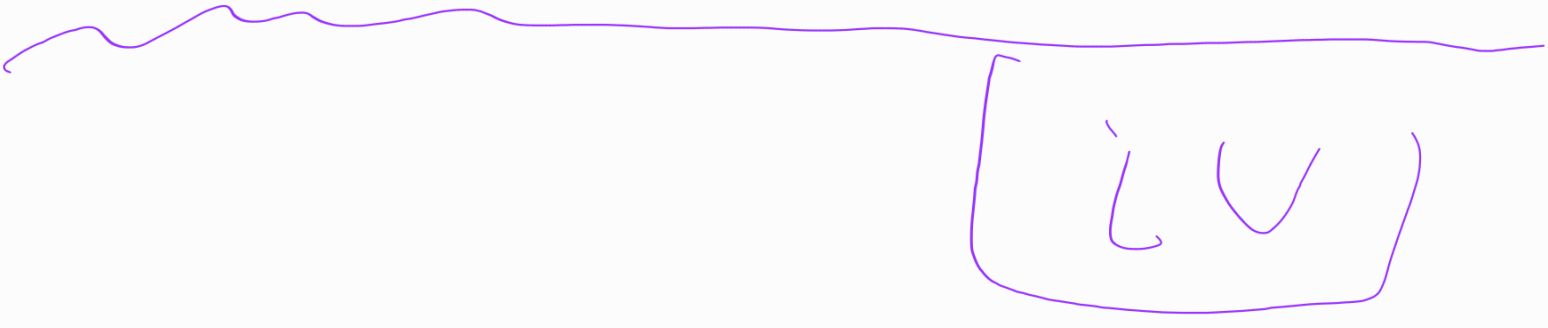
BLUE$_{c_2}$

$$1 \times e^{\left(\frac{1}{2}\log\frac{2}{5} + \frac{1}{2}\log\frac{1}{4}\right)}$$

$$= 0.32$$

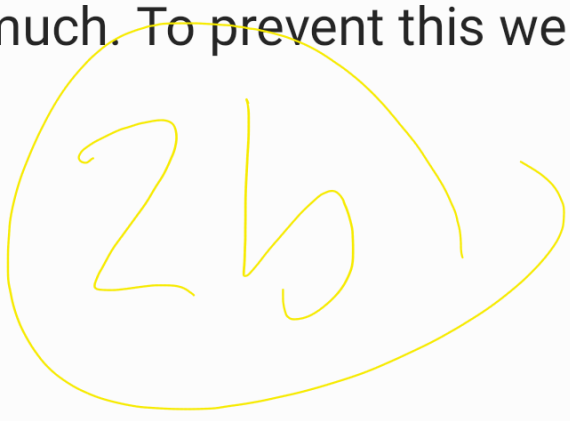
No this time I still think the second translation is better but first translation got bigger bleu score, I think this shows us the importance of reference sentences for accurate calculation of BLEU score.

In previous question we saw the dépendance of bleu accuracy on good references, so with only one reference at hand, we reduce bleu score for those good translation that convey the meaning but in other words.



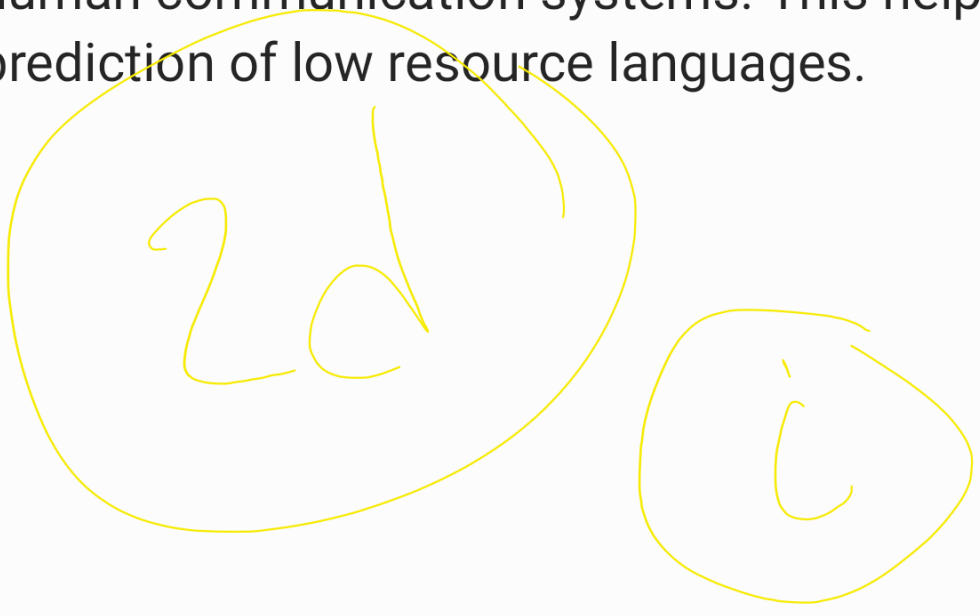
The advantage is it is simple to compute and it some how look like the way human evaluates a translation, but the disadvantage is it's high dépendance on tokens and it can not accurate score for those translation that have the same meaning but in other words, but humans can understand these types of translation.

Languages like chrokee are called polysynthesis, this means they can contain very long words that are constructed by concatenation smaller words that have meaning but does not appear stand alone in a sentence. Not using sub words cause a big problem when it comes to embeddding. Since long words are just multiple small words, we can not have good embeddings for them because they may not appear as much. To prevent this we use sub word embedding.

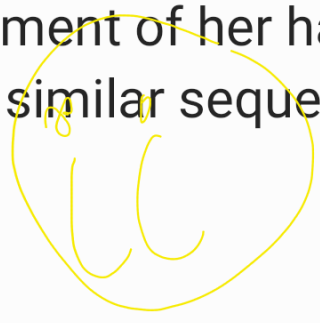


Stand alone characters do not convey any special meaning in many cases (except maybe in cases like math chars: e and etc.) so there is no need for big vectors in order to embedd them. Sub words do not need big embedding vectors too, since sub words are like building block of bigger words that convey more meaning, for example pre and occupy both have their indépendant meaning but when they combine they have a new meaning which is a combination of both meaning, so instead of embedding big wods, we can embedd smaller building block words, since those building block does not convey as much meaning, we can use smaller embedding.

In multilingual NMT, we train a huge model on all available pair of sentences despite their language and dataset size, Google researches shown large improvement over both language with scarce data and big data. The idea is to learn common human signals behind languages that used in all human communication systems. This helps to improve the prediction of low resource languages.



Since we face scarce data, maybe due to lack of examples, crown and hair found similar vectors. Also it can be a result of greedy decoding (wrong placement of her hair). We can add some penalty for generating similar sequences.



Maybe chrokee does not have all English pronouns (it is a gender neutral language) in that case we can use corefrence to automatically detect pronoun gender or use a gender

neutral token for English pronouons.

I sense two mistakes here, one is the wrong translation of swim and the other one is to not understanding that Little fish is an indépendant entity. So adding more data can solve these problems.