1.

(a)

Self-attention in Transformer is to generate the output based only on learning which input words are important, where attention weight as the dot-product of each input token with every other token.

let x be the input and y be the output.

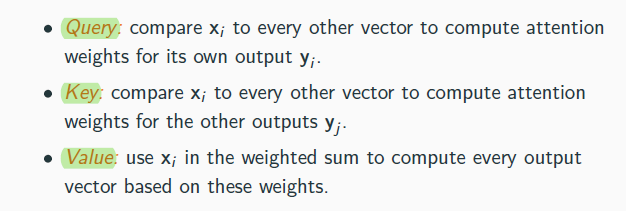
Y\_i = sum(w\_ij x\_j)

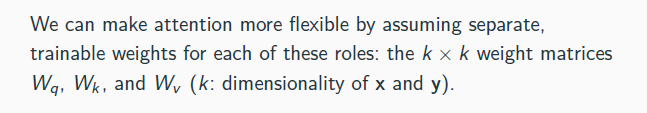
W\_ij’ = x\_i \* x\_j

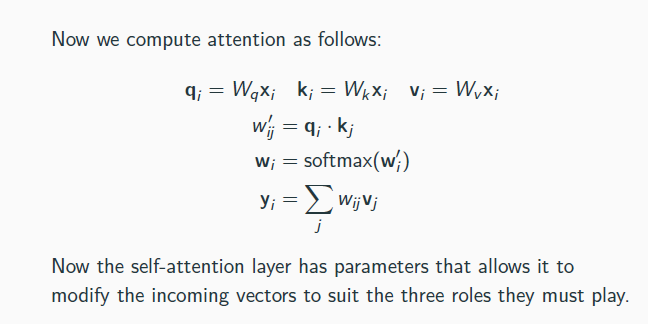
W = softmax(w\_i’)

(b)

Each input vector xi is used in three ways in self-attention:



(why flexible)



(c)

No. Because the objective is topic classification, where some specific word could have important weight on classification, such as ‘football’ is the representative of ‘sport’. It doesn’t matter whether the sentence is ‘I like football, but I don’t like basketball’ or ‘I don’t like football, but I like basketball’. From the above example, we can see the position of the word ‘don’t’ doesn’t change the topic of the sentence. Thus, position embedding is unnecessary in this situation.

2.

(a)

Input: sentence. -> Vector. (one-hot encoding, subword embedding?)

Output: the semantic role tagging of each phrase in the sentence. -> Vector.

(b)

Replace the final output layer to suit the task.

MLP / logstic regression

Instead of output the vector representation of the input sentence, we use the hidden states before the output layer to predict the corresponding semantic role tagging of each phrase through softmax.

(c)

Models like Bert have already trained a general good set of parameters from another large dataset. By truncating the last layer and replacing with a new one that suits the task. Then, we freeze the weight and use small learning rate to let Bert learn this semantic role tagging task.

The parameter in the last output layer should be tuned (randomly initialized).

Objective is maximizing the NLL of the final whole tagging sequence.

Minimize cross-entropy.

(d)

After the final Bert output layer, we put the output contextual embedding as another conditional language model’s input to predict the tagging result.

(e)

Add attention.

(use RNN)

Our model will achieve multi-predicate in following steps:

1. We need to find all the predicate in the input sentence
2. Depending of the predicate, we apply multi-head attention on each predicate to capture the most likely A0, A1, AM for each predicate
3. Finally, we predict the tagging based on the softmax

考虑到多tag的情况。

3.

(a)

~~i. RNN has gradient vanishing, while CNN don’t. (??????????????)~~

ii. CNN focus on local feature extraction and parameter sharing by using filter, while RNN’s structure where the input of time step t needs the state of step t-1 and input x\_t, allow it to capture the time order of the input, such as NL.

iii. CNN’s input and output are strictly formed in size, while RNN’s input could either be fixed or flexible (dynamic RNN)

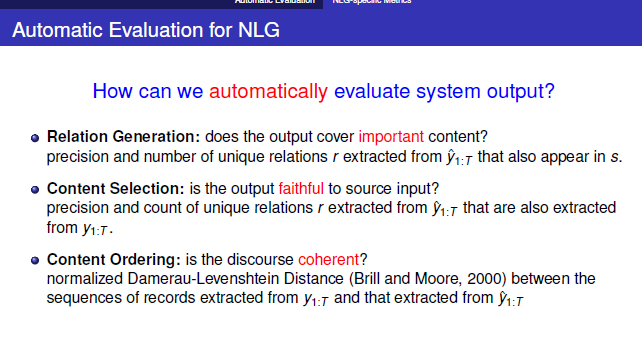
iv. RNN is sequence input CNN is spatial input.

v. rnn’s hidden state only contain one direction, CNN contains information around

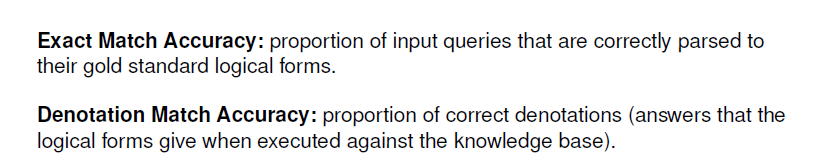
(b)

i. ROUGE: evaluates n-gram overlap of the generated text (candidate) with a reference

ii. BLEU: compares n-grams of a candidate text (e.g. that generated by an algorithm) with the n-grams of a reference text and also penalize the output by its length



iii.

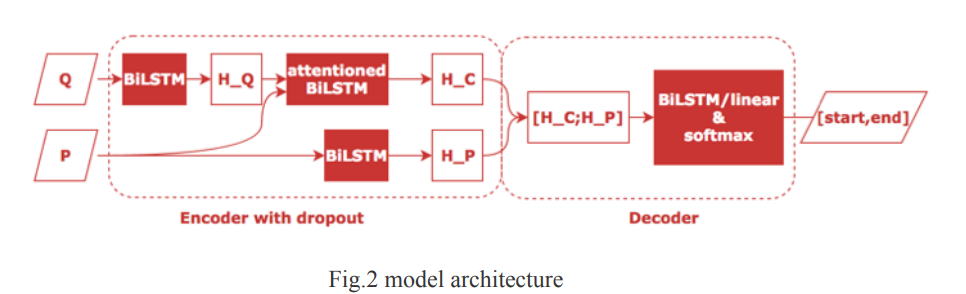


4.

(a) Given the answer text passage x and the answer a, we predict the word of question one by one.

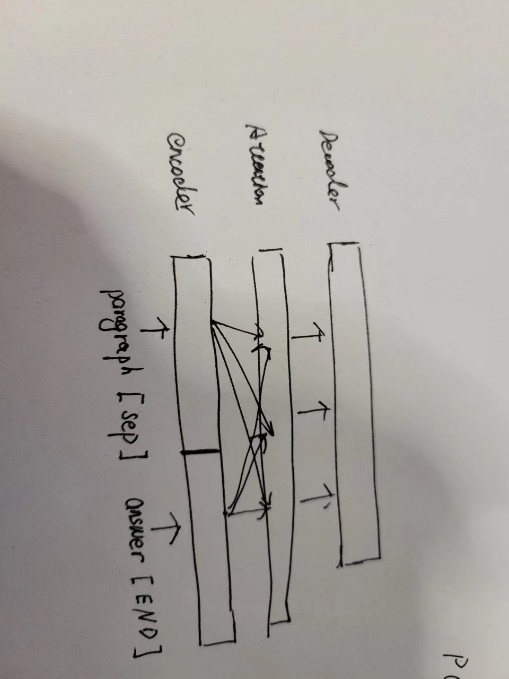
P(y\_t|y\_<t, x, a)

(b)



Encoder’s input: paragraph, answer h\_i = RNN(x, a, h\_i)

Decoder’s output: s\_i and softmax(W concat(si+h\_i)+b)

(delete attention)

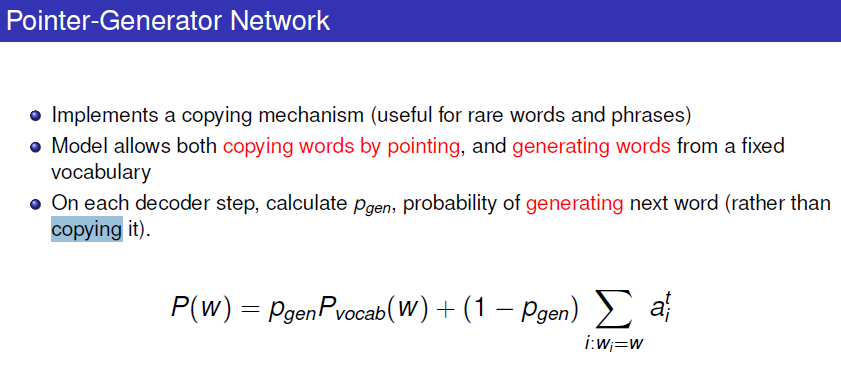
(c)

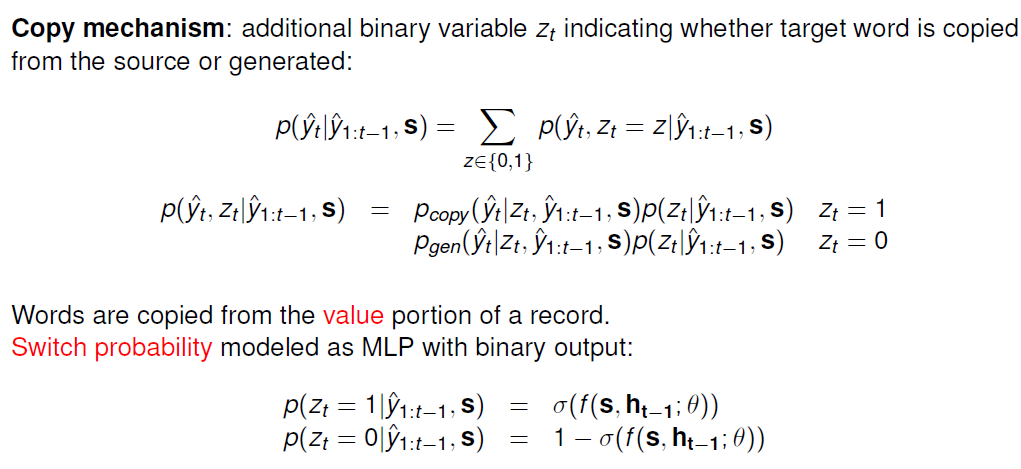
In the encoder, we calculate the paragraph’s attention on the answers.

In the decoder, we use multi-head attention to calculate the importance of questions on the paragraph and answer separately.

(d)

Yes, as seen from the example, we can see that a large amount of text in question are extracted from the paragraph.





(e)

Minimize the training loss (negative likelihood of target word in question)

Loss = -SUM(log p(question))

(f)

ROUGE:

BLEU:

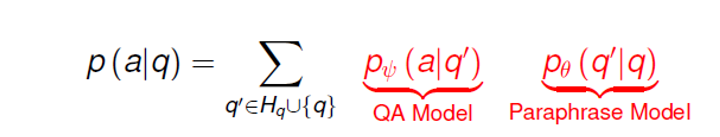
Cons: 正确性不足

见slides

COMET:

(g)

We could use MT system to generate the pivot of training data in English, also includes the answer piviot. Then our new model for German question generation’s objective changes to

，which contains two parts, one is QA model, one is for en-de translation model.

A(de)-> A(en)->Q(en)->Q(de)