1.

(a)

i. No. Because there is no linear boundary that makes the output y linear separatable.

ii. a XOR b = (-a AND b) OR (a AND -b)

(b)

(1) n-gram: P(xi | x1, . . . , xi−1) ~ P(xi | xi-n+1, . . . , xi−1). The probabilities are non-negative and sum to one. Markov assumption.

(2) feedforward: Basic idea is same as n-gram, but all the data use vector instead. Softmax could map any vector into a probability distribution. Y = softmax(Wx+b) Markov assumption.

(3) RNN: without Markov assumption. Any time step’s output only depends on its previous hidden state and current input x.

(c)

i. No. subword machine translation model could learn the subword translation. Thus, even the words never appears in the training data, we could also translate the word by its subwords and combine them.

ii. Yes. Because self-attention only cares about the weight of importance each word contributed to the target word, instead of concerning about the order of those words. While RNN itself structure has the property of capturing the order information in the sequence data, although cannot capture very long-distance dependency because of gradient vanishing.

(d)

Because the right-to-left information is missing in the decoder layer, where only has the encoder output and left-to-right history decoder output.

(e)

It depends.

If the MT system is fluency-oriented or lack of human references, we could use MTeRater instead. Besides fluency, adequacy is also important for MT system, for general MT tasks, I would prefer BLEU. If only maximize the fluency, the adequacy may decrease, the metrics of MT system should balance these two.

Penalty of the length

2.

(a)

Standard encoder-decoder cannot capture the order importance among the words in the input, and the last word always have the highest weight.

Use position encoding to know how important each hidden state contributes to the prediction

(bidirectional encoder)

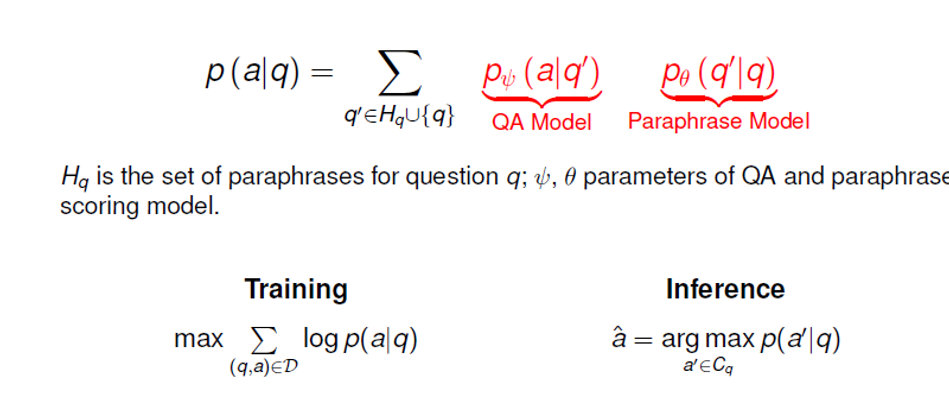
(b)

Encoder-decoder usually generates sequence as output, while this task is classification, which is not fit by using this architecture.

We could drop the decoder layer and add a MLP with the final hidden state of encoder. Or use BERT as contextual embedding and finetuning the output to fit the classification task.

(c)

Q-> logic form -> query from database-> records-> A



3.

(a)

Encoder: takes whole doctor report as input, using a single layer bidirectional LSTM produce a sequence of hidden states.

Decoder: single layer unidirectional LSTM receive word embedding of precious word emitted by decoder and has decoder states. Copy mechanism is used to copy words in the report or generated from vocabulary with the highest probability.

Prompt:

(b)

The data contains all the vocabulary model output needed.

Weakness: only works well on medical codes usually happens, those codes with rare opportunity to happen are hard to generated correctly.

Weakness: prompt could only focus on specific task.