Ass_4_seq2seq

June 22, 2020 9:35 PM

$$\mathbf{h}_i^{\text{enc}} = [\overleftarrow{\mathbf{h}_i^{\text{enc}}}; \overrightarrow{\mathbf{h}_i^{\text{enc}}}] \text{ where } \mathbf{h}_i^{\text{enc}} \in \mathbb{R}^{2h \times 1}, \overleftarrow{\mathbf{h}_i^{\text{enc}}}, \overrightarrow{\mathbf{h}_i^{\text{enc}}} \in \mathbb{R}^{h \times 1} \qquad 1 \le i \le m \tag{1}$$

$$\mathbf{c}_i^{\text{enc}} = [\overleftarrow{\mathbf{c}_i^{\text{enc}}}; \overrightarrow{\mathbf{c}_i^{\text{enc}}}] \text{ where } \mathbf{c}_i^{\text{enc}} \in \mathbb{R}^{2h \times 1}, \overleftarrow{\mathbf{c}_i^{\text{enc}}}, \overrightarrow{\mathbf{c}_i^{\text{enc}}} \in \mathbb{R}^{h \times 1} \qquad 1 \le i \le m \tag{2}$$

We then initialize the Decoder's first hidden state $\mathbf{h}_0^{\text{dec}}$ and cell state $\mathbf{c}_0^{\text{dec}}$ with a linear projection of the Encoder's final hidden state and final cell state.[1]

$$\mathbf{h}_0^{\text{dec}} = \mathbf{W}_h[\overleftarrow{\mathbf{h}_1^{\text{enc}}}; \overrightarrow{\mathbf{h}_m^{\text{enc}}}] \text{ where } \mathbf{h}_0^{\text{dec}} \in \mathbb{R}^{h \times 1}, \mathbf{W}_h \in \mathbb{R}^{h \times 2h} \tag{3}$$

$$\mathbf{c}_0^{\text{dec}} = \mathbf{W}_c[\overleftarrow{\mathbf{c}_1^{\text{enc}}}; \overrightarrow{\mathbf{c}_m^{\text{enc}}}] \text{ where } \mathbf{c}_0^{\text{dec}} \in \mathbb{R}^{h \times 1}, \mathbf{W}_c \in \mathbb{R}^{h \times 2h} \tag{4}$$

With the Decoder initialized, we must now feed it a matching sentence in the target language. On the t^{th} step, we look up the embedding for the t^{th} word, $\mathbf{y}_t \in \mathbb{R}^{e \times 1}$. We then concatenate $\mathbf{y}_t$ with the *combined-output vector* $\mathbf{o}_{t-1} \in \mathbb{R}^{h \times 1}$ from the previous timestep (we will explain what this is later down this page!) to produce $\overline{\mathbf{y}_t} \in \mathbb{R}^{(e+h) \times 1}$. Note that for the first target word (i.e. the start token) $\mathbf{o}_0$ is a zero-vector. We then feed $\overline{\mathbf{y}_t}$ as input to the Decoder LSTM.

$$\mathbf{h}_t^{\text{dec}}, \mathbf{c}_t^{\text{dec}} = \text{Decoder}(\overline{\mathbf{y}_t}, \mathbf{h}_{t-1}^{\text{dec}}, \mathbf{c}_{t-1}^{\text{dec}}) \text{ where } \mathbf{h}_t^{\text{dec}} \in \mathbb{R}^{h \times 1}, \mathbf{c}_t^{\text{dec}} \in \mathbb{R}^{h \times 1} \tag{5}$$

$$\tag{6}$$

We then use $\mathbf{h}_t^{\text{dec}}$ to compute multiplicative attention over $\mathbf{h}_0^{\text{enc}}, \ldots, \mathbf{h}_m^{\text{enc}}$:

$$\mathbf{e}_{t,i} = (\mathbf{h}_t^{\text{dec}})^T \mathbf{W}_{\text{attProj}} \mathbf{h}_i^{\text{enc}} \text{ where } \mathbf{e}_t \in \mathbb{R}^{m \times 1}, \mathbf{W}_{\text{attProj}} \in \mathbb{R}^{h \times 2h} \qquad 1 \le i \le m \tag{7}$$

$$\alpha_t = \text{Softmax}(\mathbf{e}_t) \text{ where } \alpha_t \in \mathbb{R}^{m \times 1} \tag{8}$$

$$\mathbf{a}_t = \sum_i^m \alpha_{t,i} \mathbf{h}_i^{\text{enc}} \text{ where } \mathbf{a}_t \in \mathbb{R}^{2h \times 1} \tag{9}$$

We now concatenate the attention output $\mathbf{a}_t$ with the decoder hidden state $\mathbf{h}_t^{\text{dec}}$ and pass this through a linear layer, Tanh, and Dropout to attain the *combined-output* vector $\mathbf{o}_t$.

$$\mathbf{u}_t = [\mathbf{a}_t; \mathbf{h}_t^{\text{dec}}] \text{ where } \mathbf{u}_t \in \mathbb{R}^{3h \times 1} \tag{10}$$

$$\mathbf{v}_t = \mathbf{W}_u \mathbf{u}_t \text{ where } \mathbf{v}_t \in \mathbb{R}^{h \times 1}, \mathbf{W}_u \in \mathbb{R}^{h \times 3h} \tag{11}$$

$$\mathbf{o}_t = \text{Dropout}(\text{Tanh}(\mathbf{v}_t)) \text{ where } \mathbf{o}_t \in \mathbb{R}^{h \times 1} \tag{12}$$

Then, we produce a probability distribution $\mathbf{P}_t$ over target words at the t^{th} timestep:

$$\mathbf{P}_t = \text{Softmax}(\mathbf{W}_{\text{vocab}} \mathbf{o}_t) \text{ where } \mathbf{P}_t \in \mathbb{R}^{V_t \times 1}, \mathbf{W}_{\text{vocab}} \in \mathbb{R}^{V_t \times h} \tag{13}$$

Here, V_t is the size of the target vocabulary. Finally, to train the network we then compute the softmax cross entropy loss between $\mathbf{P}_t$ and $\mathbf{g}_t$, where $\mathbf{g}_t$ is the 1-hot vector of the target word at timestep t:

[1] If it's not obvious, think about why we regard $[\overleftarrow{\mathbf{h}_1^{\text{enc}}}, \overrightarrow{\mathbf{h}_m^{\text{enc}}}]$ as the 'final hidden state' of the Encoder.

(g) (3 points) (written) The `generate_sent_masks()` function in `nmt_model.py` produces a tensor called `enc_masks`. It has shape (batch size, max source sentence length) and contains 1s in positions corresponding to 'pad' tokens in the input, and 0s for non-pad tokens. Look at how the masks are used during the attention computation in the `step()` function (lines 295-296).

First explain (in around three sentences) what effect the masks have on the entire attention computation. Then explain (in one or two sentences) why it is necessary to use the masks in this way.

```
def generate_sent_masks(self, enc_hiddens: torch.Tensor, source_lengths: List[int]) -> torch.Tensor:
        """ Generate sentence masks for encoder hidden states.

        @param enc_hiddens (Tensor): encodings of shape (b, src_len, 2*h), where b = batch size,
                                     src_len = max source length, h = hidden size.
        @param source_lengths (List[int]): List of actual lengths for each of the sentences in the batch.

        @returns enc_masks (Tensor): Tensor of sentence masks of shape (b, src_len),
                                    where src_len = max source length, h = hidden size.
        """
        enc_masks = torch.zeros(enc_hiddens.size(0), enc_hiddens.size(1), dtype=torch.float)
        for e_id, src_len in enumerate(source_lengths):
            enc_masks[e_id, src_len:] = 1
        return enc_masks.to(self.device)
```

Enc_mask has the same dimension as enc_hidden

Src_len: =1 identifies the paddings in the encoder

In step, it assigns negative infinity score on the paddings, which tells the model that paddings are not important;
  If not, the attention score calculation would only use the encoder att projection and the decoder hidden layer for attention score calculation

```
# Set e_t to -inf where enc_masks has 1
    if enc_masks is not None:
        e_t.data.masked_fill_(enc_masks.byte(), -float('inf'))
```

https://docs.google.com/document/d/1z9ST0IvxHQ3HXSAOmpcVbFU5zesMeTtAc9km6LAPJxk/edit#

```
load model from model.bin
Decoding: 100%|██████████████████████████████████████████████████|
8064/8064 [06:14<00:00, 21.52it/s]
Corpus BLEU: 22.677545638804883
```

(j) (3 points) In class, we learned about dot product attention, multiplicative attention, and additive attention. Please provide one possible advantage and disadvantage of each attention mechanism, with respect to either of the other two attention mechanisms. As a reminder, dot product attention is $\mathbf{e}_{t,i} = \mathbf{s}_t^T \mathbf{h}_i$, multiplicative attention is $\mathbf{e}_{t,i} = \mathbf{s}_t^T \mathbf{W} \mathbf{h}_i$, and additive attention is $\mathbf{e}_{t,i} = \mathbf{v}^T(\mathbf{W}_1 \mathbf{h}_i + \mathbf{W}_2 \mathbf{s}_t)$.

Dot product:
- Only depends on encoder and decoder hidden states
- Less flexible

✓ Compute & Memory efficient, easy to interplate state similarity

✗ Little flexibility

Multiplicative:
- Can modify the weight to emphasis using covariance

✓ : Learnable parameter matrix W, able to transform source hidden state

Addictive:
- Can modify the weight of emphasis on the encoder or the decoder

✓ : flexibility

✗ : slower

2. Analyzing NMT Systems (30 points)

(a) (12 points) Here we present a series of errors we found in the outputs of our NMT model (which is the same as the one you just trained). For each example of a Spanish source sentence, reference (i.e., 'gold') English translation, and NMT (i.e., 'model') English translation, please:

1. Identify the error in the NMT translation.
2. Provide a reason why the model may have made the error (either due to a specific linguistic construct or specific model limitations).
3. Describe one possible way we might alter the NMT system to fix the observed error.

Below are the translations that you should analyze as described above. Note that out-of-vocabulary words are underlined.

i. (2 points) **Source Sentence:** *Aquí otro de mis favoritos, "La noche estrellada".*
Reference Translation: *So another one of my favorites, "The Starry Night".*
NMT Translation: *Here's another favorite of my favorites, "The Starry Night".*

Didn't use pronoun

ii. (2 points) **Source Sentence:** *Ustedes saben que lo que yo hago es escribir para los niños, y, de hecho, probablemente soy el autor para niños, ms ledo en los EEUU.*
Reference Translation: *You know, what I do is write for children, and I'm probably America's most widely read children's author, in fact.*
NMT Translation: *You know what I do is write for children, and in fact, I'm probably the author for children, more reading in the U.S.*

language order (Alignment).
learn more samples with such difference?

iii. (2 points) **Source Sentence:** *Un amigo me hizo eso – Richard Bolingbroke.*
Reference Translation: *A friend of mine did that – Richard Bolingbroke.*
NMT Translation: *A friend of mine did that – Richard <unk>*

Unknown rare words.
Add to training Corpus.

iv. (2 points) **Source Sentence:** *Solo tienes que dar vuelta a la manzana para verlo como una epifanía.*
Reference Translation: *You've just got to go around the block to see it as an epiphany.*
NMT Translation: *You just have to go back to the apple to see it as a epiphany.*

?

Idiom

v. (2 points) **Source Sentence:** *Ella salvó mi vida al permitirme entrar al baño de la sala de profesores.*
Reference Translation: *She saved my life by letting me go to the bathroom in the teachers' lounge.*
NMT Translation: *She saved my life by letting me go to the bathroom in the women's room.*

Sex oriented word property?

vi. (2 points) **Source Sentence:** *Eso es más de 100,000 hectáreas.*
Reference Translation: *That's more than 250 thousand acres.*
NMT Translation: *That's over 100,000 acres.*

different unit under Context $\Rightarrow$ also learn the unit system Under Certain language environment

b) <unk> in line 4: unknown rare words,
Use more words in the training samples shortlist

Line 13/14: pronoun is not correct

(c) (14 points) BLEU Score is the most commonly used automatic evaluation metric for NMT systems. It is usually calculated across the entire test set, but here we will consider BLEU defined for a single example.[3] Suppose we have a source sentence **s**, a set of k reference translations $\mathbf{r}_1, \ldots, \mathbf{r}_k$, and a candidate translation **c**. To compute the BLEU score of **c**, we first compute the *modified n-gram precision* p_n of **c**, for each of $n = 1, 2, 3, 4$:

$$p_n = \frac{\sum_{\text{ngram} \in \mathbf{c}} \min\left(\max_{i=1,\ldots,k} \text{Count}_{\mathbf{r}_i}(\text{ngram}),\ \text{Count}_{\mathbf{c}}(\text{ngram}) \right)}{\sum_{\text{ngram} \in \mathbf{c}} \text{Count}_{\mathbf{c}}(\text{ngram})} \tag{15}$$

Here, for each of the n-grams that appear in the candidate translation **c**, we count the maximum number of times it appears in any one reference translation, capped by the number of times it appears in **c** (this is the numerator). We divide this by the number of n-grams in **c** (denominator).

Next, we compute the *brevity penalty* BP. Let c be the length of **c** and let r^* be the length of the reference translation that is closest to c (in the case of two equally-close reference translation lengths, choose r^* as the shorter one).

$$BP = \begin{cases} 1 & \text{if } c \geq r^* \\ \exp\left(1 - \frac{r^*}{c}\right) & \text{otherwise} \end{cases} \tag{16}$$

Lastly, the BLEU score for candidate **c** with respect to $\mathbf{r}_1, \ldots, \mathbf{r}_k$ is:

$$BLEU = BP \times \exp\left(\sum_{n=1}^{4} \lambda_n \log p_n \right) \tag{17}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are weights that sum to 1.

i. (5 points) Please consider this example:

Source Sentence **s**: **el amor todo lo puede**
Reference Translation $\mathbf{r}_1$: *love can always find a way*
Reference Translation $\mathbf{r}_2$: *love makes anything possible*
NMT Translation $\mathbf{c}_1$: *the love can always do*
NMT Translation $\mathbf{c}_2$: *love can make anything possible*

Please compute the BLEU scores for $\mathbf{c}_1$ and $\mathbf{c}_2$. Let $\lambda_i = 0.5$ for $i \in \{1, 2\}$ and $\lambda_i = 0$ for $i \in \{3, 4\}$ (**this means we ignore 3-grams and 4-grams**, i.e., don't compute p_3 or p_4). When computing BLEU scores, show your working (i.e., show your computed values for p_1, p_2, c, r^* and BP).

Which of the two NMT translations is considered the better translation according to the BLEU Score? Do you agree that it is the better translation?

C_1: 1 gram:
2 gram: the love, love can. can always. always do

C_2: 1 gram
2 gram: love can, can make, make anything, anything possible.

P_n: C_1:

ngram = 1

$\max_{i=1..k}$	$\text{Count}_{r_i}(\text{ngram})$	n=1	$\text{Count}_c(\text{ngram})$
0	0	the	1
1	1	love	1
1	1	can	1
1	1	always	1
			1

$\max_{i=1..k}$	$\text{Count}_{r_i}(n)$	n=2	$\text{Count}_c(n)$
0	0	the love	1
1	1	love can	1
1	1	can always.	1
0	0	always do.	1

$P_1 = \frac{3}{5}$

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

1	1	can	1
1	1	always	1
0	0	do	1

1	1	can always.	1
0	0	always do.	1

$p_2 = \frac{1}{4}$

C_2:

1	1	love	1
1	1	can	1
0	0	makes	1
1	0	anything	1
1	0	possible	1

1	1	love can	1
0	0	can make	1
1	0	make anything	1
1	0	anything possible.	1

$P_1 = \frac{4}{5}$

$P_2 = \frac{3}{4}$

$BP = 1$ Since $len(c) = len(r_t)$.

$BLEU = BP \times \exp\left(\sum_{n=1}^{2} \lambda_n \log p_n\right)$ ($\lambda_n = 0.5$)

$C_1 = 1 \times \exp\left(0.5 \times \left(\frac{3}{5} + \frac{2}{4}\right)\right) = 1.32$ → $\log\frac{3}{5} + \log\frac{2}{4} = 0.77$

$C_2 = 1 \times \exp\left(0.5\left(\frac{4}{5} + \frac{3}{4}\right)\right) = 1.48$ ✓ → $\log\frac{4}{5} + \log\frac{3}{4} = 0.8195$

Better. Agree

(ii) Without r_2 BLEU:

C_1: Same ✓ higher score, Not agree

C_2: $BLEU = 1 \times \exp\left(0.5\left(\log\frac{2}{5} + \log\frac{1}{4}\right)\right) = 0.6065$ much less

(iii) only single reference translation:
The machine doesn't know there's another way or translating the text, then the score might not be high in this way, but it could be a really good way of expressing the same meaning

(iiii) Two advantages and disadvantages of BLEU:

a. Good:
   i. provide a base line
   ii. Fast to calculate
b. Bad:
   i. Depends on the number of reference translation × Semantics
   ii. Slow to calculate? × Structure.