# Applied Deep Learning Assignment 4

Tung-Chun, Chiang R05922027

December 9, 2016

## 1 Data Preprocessing

In this assignment, we have two task: machine translation and natural language generation. For machine translation, we have a sequence of input words in English and the corresponding sequence of output words in Spanish. For natural language generation, we have an input function name, some entities, contents of entities and two corresponding sentences. To simplify the task, we replace the contents of entities by $\_[entityname]\_$, as Eq. 1. When predicting, we mapping the $\_[entityname]\_$ to original contents, as Eq 2.

$$\begin{aligned} &inform(name =' trattoriacontadina'; pricerange = moderate) \\ &\rightarrow inform\ name\ \_name\_\ pricerange\ \_pricerange\_ \end{aligned} \tag{1}$$

$$\begin{aligned} &\_name\_\ is\ a\ nice\ restaurant\ in\ the\ \_pricerange\_\ price\ range \\ &\rightarrow trattoria\ contadina\ is\ a\ nice\ restaurant\ in\ the\ moderate\ price\ range \end{aligned} \tag{2}$$

## 2 Recurrent Neuron Network

In both the tasks, we use two layers RNN whose input is English word sequence for translation and parsed function for generation, while the output is Spanish word sequence for translation and word sentence for generation and both use sampled softmax loss as loss function. Figure 1 shows the model structure. And the objective function is to maximize the probability of an output label given all output labels before it(Eq. 3).

$$\max_{\theta} \prod_{t=1}^{T} P(y_t | y_{t-1}, y_{t-2}, ..., y_1, x; \theta) \tag{3}$$

$y_t$ means the label output as time step $t$, $x$ means input words, $\theta$ is the RNN model.

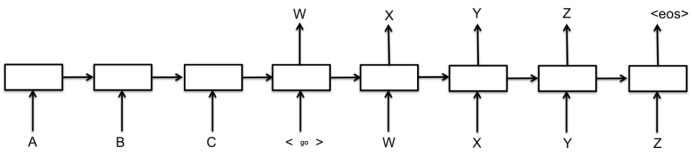


Figure 1: Recurrent Neuron Network

# 3 Improvement

## 3.1 Attention

To improve the model, we add attention mask in the model. In Eq 4, the $c$ is attention value and it is a linear combination of the outputs of input word sequence(length $T$) and the weights $\alpha$ are also learned in training. And then, concatenate $c$ and $h$ as decoder inputs.

$$c_i = \sum_{j=1}^{T} \alpha_{i,j} h_j \tag{4}$$

## 3.2 Buckets

In Tensorflow sample seq2seq model, they use "buckets" to split the training data by sequence lengths. If we use more buckets, less training time we need, but we get worse target score. It is a trade-off.

## 3.3 BLEU score

If we don't map $\_[entityname]\_$ to original contents, we map $\_[entityname]\_$ to $[entityname]$. The BLEU score increases significantly. Because when computing BLEU, the contents with multi-words may lead to N-gram computation error.

# 4 Learned

- Because of gradient explosion problem, we should clip the gradients.
- For gradient vanish problem, we use LSTM to control the back propagation flow.
- We can also use drop out to avoid overfitting.
- Use buckets to control the training speed and the power of modeling.

# References


[1] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, *Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation*, EMNLP 2014.

[2] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, *BLEU: a Method for Automatic Evaluation of Machine Translation*, ACL 2002.