Conditional Sequence-to-Sequence VAE

Shih-Po, Lee
mapl0756051.cs07g@nctu.edu.tw

1 Lab Description

In this lab, you need to implement a conditional seq2seq VAE for English tense conversion and generation. VAE [4] has been applied to many NLP generation task such as text summarization and paraphrase. Specifically, your model should be able to do English tense conversion and text generation. For example, when we input the input word 'access' with the tense (the condition) 'simple present' to the encoder, it will generate a latent vector z. Then, we take z with the tense 'present progressive' as the input for the decoder and we expect that the output word should be 'accessing'. In addition, we can also manually generate a Gaussian noise vector and feed it with different tenses to the decoder and generate a word those tenses.

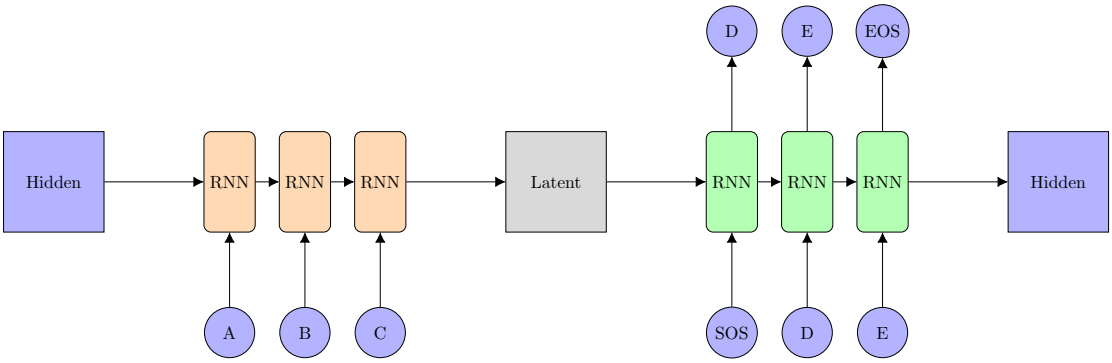

Figure 1: The illustration of sequence-to-sequence architecture.

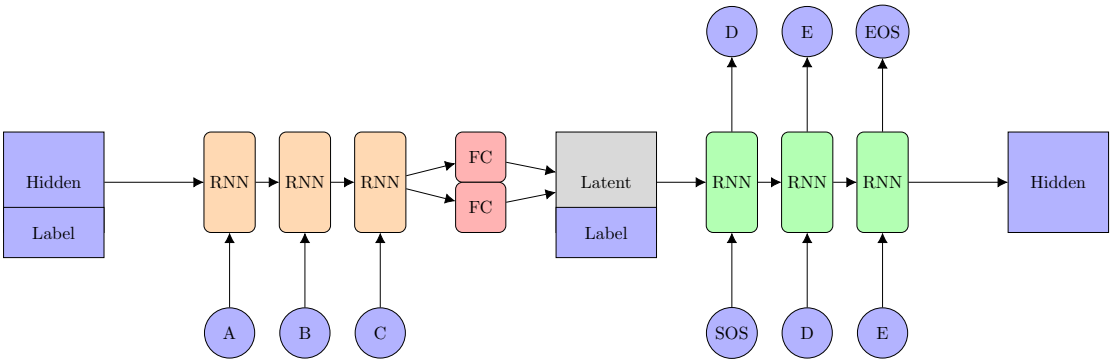

Figure 2: The illustration of sequence-to-sequence VAE architecture.

2 Requirements

- **Implement a seq2seq model**
  - Modify encoder, decoder, and training functions
  - Implement evaluation function and dataloader, and reparameterization trick
- **Plot the CrossEntropy training loss and BLEU-4 testing score curves during training**
  - Teacher forcing ratio
  - KL annealing schedules (two methods)
- **Output the conversion results between tenses (from tense A to tense B)**
- **Output the results generated by a Gaussian noise with 4 tenses**

3 Implementation Details

3.1 Variational Autoencoder

Recall that the loss function of VAE

$$L(X, q, \theta) = E_{z\sim q(Z|X;\phi)}\log p(X|Z;\theta) - KL(q(Z|X;\phi)||p(Z)) \tag{1}$$

where $q(Z|X;\phi)$ is considered as encoder and $p(X|Z;\theta)$ as decoder.
To train the model end-to-end, we adopt the reparameterization trick. (see in Fig. 3). The output of reparameterization trick should be log variance instead of variance directly.

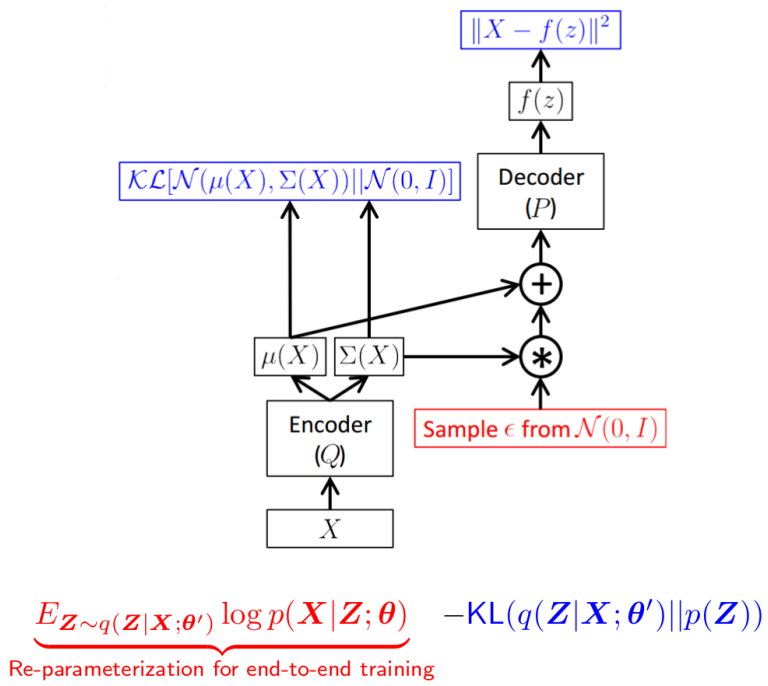

Figure 3: The illustration of reparameterization trick.

3.2 Conditional VAE

The objective function of conditional VAE is formulated as

$$L(X, c, q, \theta) = E_{z\sim q(Z|X,c;\phi)}\log p(X|Z, c;\theta) - KL(q(Z|X, c;\phi)||p(Z|c)) \tag{2}$$

where both the encoder $q(Z|X, c; \phi)$ and the decoder $p(X|Z, c; \theta)$ need to take c as part of their input. There several ways to add the conditional part to your VAE model. In the figure of model architecture, we concatenate the condition part with the initial hidden part as input of encoder. Similarly, we concatenate the condition part with the latent vector z as input of decoder. Before the concatenation, we construct condition embeddings via projection. You can adopt nn.Embedding and decide the size of your condition embeddings. You can also try to convert your condition into one-hot vector.

3.3 KL cost annealing

This is a simple approach adopted by [3]. We add a variable weight to the KL term in the loss function. We initially set the weight to 0. The maximum value is 1. Fig. 4 shows monotonic and cyclical annealing method. You should adopt these two methods and compare their results.

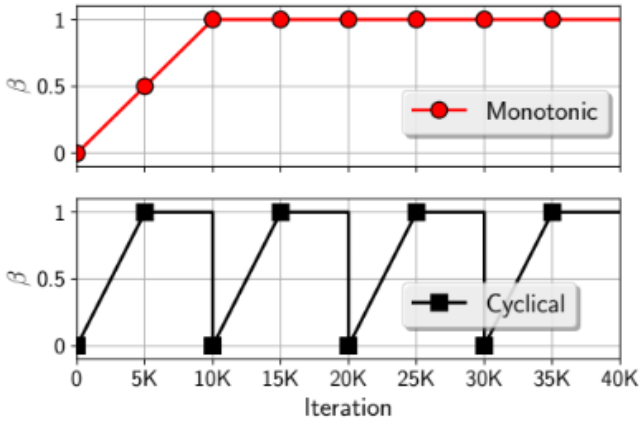

Figure 4: The design of KL annealing schedule.

3.4 Other implementation details

- The encoder and decoder must be implemented by nn.GRU, otherwise you will get no point on your report.
- You cannot use *attention mechanism* in your implementation.
- The loss function is nn.CrossEntropyLoss() and the optimizer is SGD.
- Adopt BLEU-4 score function in NLTK [1] and *Gaussian_score*() to compute the conversion and generation scores respectively. (provided in the sample code.)
- If the dimensions between layers are mismatched, you can adopt extra fully-connected layers to transform the channle size.
- Your KL annealing schedules only follow the similar notions of monotonic and cyclical methods; that is, it is not necessary to implement the exactly the same methods.
- While training, your input and output words must have the same tense. For example, if your input is 'accessing'+'progress', then your output should also be 'accessing'+'progress'.
- Hyper-parameters and model setting.
  - RNN hidden size: 256 or 512
  - Latent size: 32
  - Condition embedding size: 8
  - KL weight: $0 \sim 1$
  - Teacher forcing ratio: $0 \sim 1$
  - Learning rate:: 0.05

4 Derivation of Conditional VAE

Derive the objective function of conditional VAE 2. Start from the EM algorithm (L13, page 23)

5 Dataset Descriptions

You can download the .zip file from new e3. There are three files in the .zip: readme.txt, train.test, and test.test. All the details of the dataset are in the readme.txt.

6 Scoring Criteria

1. Report (50%)
   - Introduction (5%)
   - Derivation of CVAE (5%)
   - Implementation details (15%)
     - Describe how you implement your model (encoder, decoder, reparameterization trick, dataloader, etc.). Note that you must prove that your text generation is produced by Gaussian noise or you will get no point at this part. (paste/screenshot your code)
     - Specify the hyperparameters (KL weight, learning rate, teacher forcing ratio, epochs, etc.)
   - Results and discussion (25%)
     - Show your results of tense conversion and generation and Plot the Crossentropy loss, KL loss and BLEU-4 score curves during training (5%)
     - iscuss the results according to your setting of teacher forcing ratio, KL weight, and learning rate. Note that this part mainly focuses on your discussion, if you simply just paste your results, you will get a low score. (20%)
2. Demo (50%)
   - Capability of tense conversion on testing data. (10%)
     score = BLUE-4 score (Average your score with 10 testing data)

Accuracy	Grade
score ≥ 0.7	100%
$0.7 >$ score ≥ 0.6	90%
$0.6 >$ score ≥ 0.5	80%
$0.5 >$ score ≥ 0.4	70%
$0.4 >$ score ≥ 0.3	60%
score < 0.3	0%

   - Capability of word generation. (Gaussian noise + tense) (20%)
     score = Gaussian_score (100 words with 4 tenses)

Accuracy	Grade
score ≥ 0.3	100%
$0.3 >$ score ≥ 0.2	90%
$0.2 >$ score ≥ 0.1	80%
$0.1 >$ score ≥ 0.05	70%
score < 0.05	0%

   - Questions (20%)

7 Output Examples

1. English tense conversion with BLEU-4 score (test.txt)

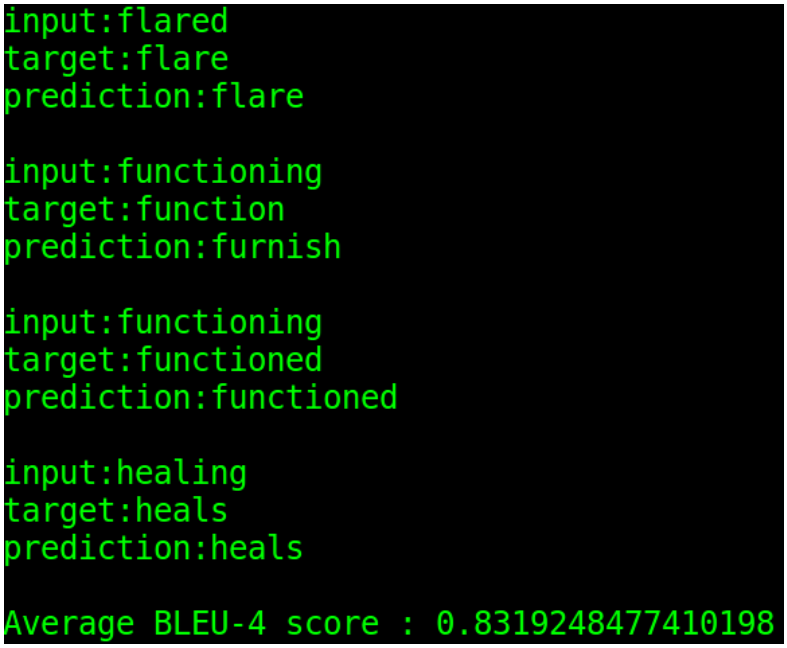

Figure 5: The output of tense conversion.

2. Gaussian noise with 4 tenses with Gaussian score

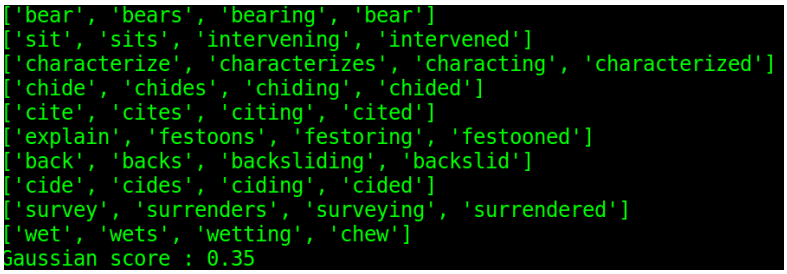

Figure 6: The output of text generation.

8 Useful Hints

1. Sequence-to-sequence model is very sensitive to the previous hidden input especially when regularizing the hidden state. Hence, I strongly suggest you save your model weights after each epoch so that you can decide which weight you like to use.
2. The teacher forcing ratio and KL weight are very important for training this model and significantly influence the performance.
3. You should know how traditional VAE works [2] before you start to build the model.

References

[1] Natural language toolkit. `https://www.nltk.org/`.
[2] Vae reference code. `https://github.com/pytorch/examples/tree/master/vae`.
[3] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space, 2015.

[4] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.