Lab 5: Conditional Sequence-to-sequence VAE

- **Lab objective**

In this lab, you need to implement a conditional seq2seq VAE for English tense conversion and generation.

- **Important Date**

1. Deadline: 5/12 (Tue.) 11:59 a.m.
2. Demo date: 5/12 (Tue.)

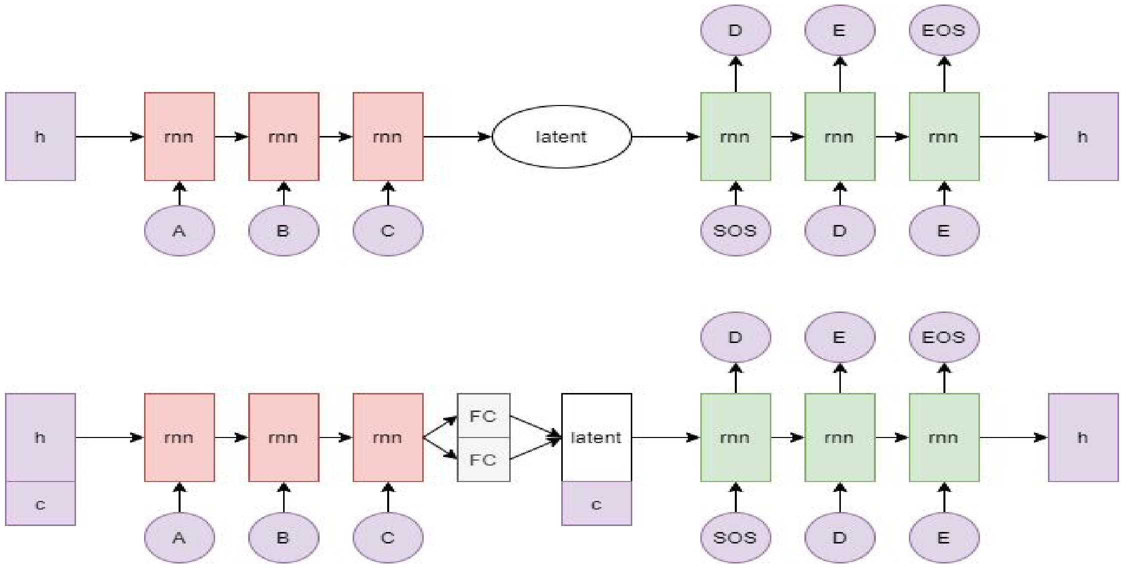
- **Turn in**

1. Experimental Report (.pdf) and Source code

Notice: zip all files in one file and name it like DLP_LAB5_your studentID_name.zip. e.g. DLP_LAB5_0756051_李仕柏.zip

- **Lab Description**

VAE has been applied to many NLP generation task such as text summarization and paraphrase. In this lab, your model should be able to do English tense conversion and generation. When we feed the input word 'access' with the tense (the condition) 'simple present' to the encoder, it will generate a latent vector z. Then, we feed z with the tense 'present progressive' to 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.



- **Requirements**

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

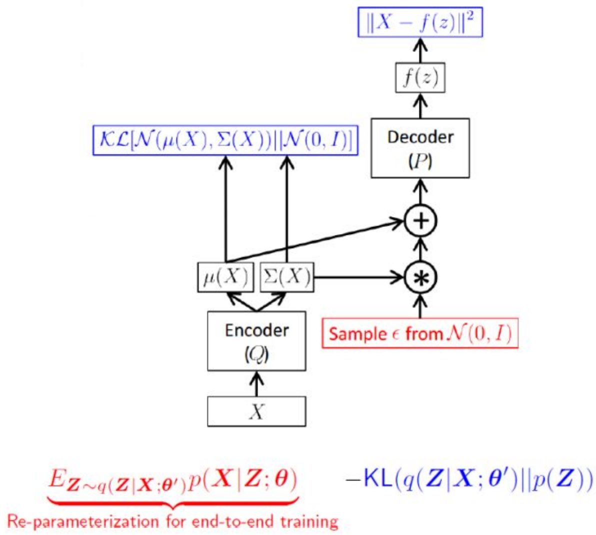
- **Implementation details**

1. VAE
   A. Recall that the loss function of VAE:

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

   where $q(Z|X;\phi)$ is considered as encoder and $p(Z;\theta)$ as decoder.

   B. Reparameterization trick:
   Train the encoder and decoder jointly.



   C. Log variance:
   The output of reparameterization trick should be log variance instead of variance directly. (see more information in [Diederik P. Kingma et al. 2014])

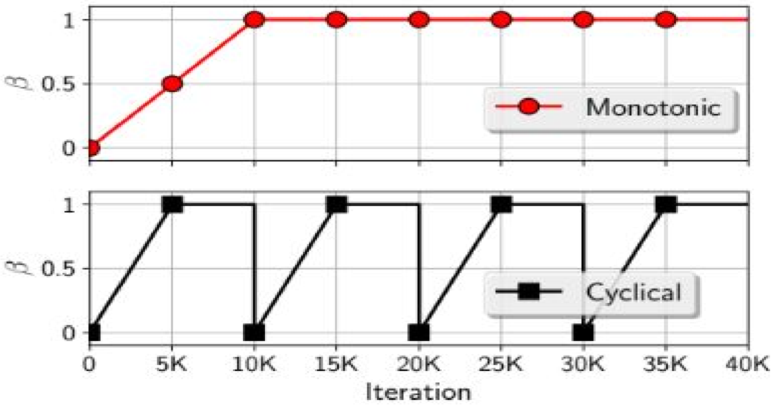
   D. Conditional VAE:

$$E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X},c;\boldsymbol{\theta}')} \log p(\boldsymbol{X}|\boldsymbol{Z},c;\boldsymbol{\theta}) - \mathrm{KL}(q(\boldsymbol{Z}|\boldsymbol{X},c;\boldsymbol{\theta}')||\,p(\boldsymbol{Z}|c))$$

   Both the encoder $q(Z|X,c;\phi)$ and the decoder $p(Z,c;\theta)$ 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.

E. KL cost annealing:

This is a simple approach adopted by [Samuel R. Bowman et al. 2016]. 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 (at most). There are two ways to implement the annealing schedule: **Monotonic method** and **Cyclical method.** You should adopt these two methods and compare their results.



2. Other implementation details
   - The encoder and decoder must be implemented by LSTM. If not, you will get no point at this part.
   - You should not adopt attention mechanism.
   - The loss function is nn.CrossEntropyLoss().
   - The optimizer is SGD
   - Adopt BLEU-4 score function in NLTK [4] and **Gaussian_score**() to help you compute your generation score. Both sample codes will be provided.
3. Hyper-parameters and model setting
   - LSTM hidden size: 256 or 512
   - Latent size: 32
   - Condition embedding size: 8
   - Teacher forcing ratio: 0~1
   - Learning rate: 0.05
   - KL weight: 0~1

- ## Derivation of Conditional VAE

  Derive the objective function of CVAE. Start from the EM algorithm (L-13, page 23).

$$E_{\boldsymbol{Z}\sim q(\boldsymbol{Z}|\boldsymbol{X},c;\boldsymbol{\theta}')} \log p(\boldsymbol{X}|\boldsymbol{Z},c;\boldsymbol{\theta}) - \mathrm{KL}(q(\boldsymbol{Z}|\boldsymbol{X},c;\boldsymbol{\theta}')||\,p(\boldsymbol{Z}|c))$$

- ## Dataset Descriptions

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

- **Scoring Criteria**

1. Report (50%)

◆ Introduction (5%)

◆ Derivation of CVAE (5%)

◆ Implementation details (15%)

A. Describe how you implement your model (encoder, decoder, reparameterization trick, dataloader, etc.). Notice: 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)

B. Specify the hyperparameters (KL weight, learning rate, teacher forcing ratio, epochs, etc.)

◆ Results and discussion (25%)

A. Show your results of tense conversion and generation and Plot the Crossentropy loss, KL loss and BLEU-4 score curves during training (5%)

B. Discuss the results according to your setting of teacher forcing ratio, KL weight, and learning rate. (20%) Notice: This part mainly focuses on your discussion, if you simply just paste your results, you will get a low score.

2. Demo (50%)

A. Capability of tense conversion on testing data. (10%)

score = **BLUE-4 score (**Average your score with 10 testing data)

- score >= 0.7 ---- 100%
- 0.7 > score >= 0.6 ---- 90%
- 0.6 > score >= 0.4 ---- 80%
- score < 0.4 ---- 0%

B. Capability of word generation. (Gaussian noise + tense) (20%)

score = **Gaussian_score**(100 words with 4 tenses)

- score >= 0.3 ---- 100%
- 0.3 > score >= 0.2 ---- 90%
- 0.2 > score >= 0.05 ---- 80%
- Score < 0.05 ---- 0%

C. Questions (20%)

- **Output examples**

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

```
['bear', 'bears', 'bearing', 'bear']
['sit', 'sits', 'intervening', 'intervened']
['characterize', 'characterizes', 'characting', 'characterized']
['chide', 'chides', 'chiding', 'chided']
['cite', 'cites', 'citing', 'cited']
['explain', 'festoons', 'festoring', 'festooned']
['back', 'backs', 'backsliding', 'backslid']
['cide', 'cides', 'ciding', 'cided']
['survey', 'surrenders', 'surveying', 'surrendered']
['wet', 'wets', 'wetting', 'chew']
Gaussian score : 0.35
```

2. Gaussian noise with 4 tenses with Gaussian score

```
input:flared
target:flare
prediction:flare

input:functioning
target:function
prediction:furnish

input:functioning
target:functioned
prediction:functioned

input:healing
target:heals
prediction:heals

Average BLEU-4 score : 0.8319248477410198
```

- **Very Useful Hints**

1. While training, your input and output words should have the same tense. For example, if your input is 'accessing'+'progress', then your output should also be 'accessing'+'progress'.
2. 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 want to use.
3. The teacher forcing ratio and KL weight are very important for training this model and **significantly influence** the performance.
4. You should know how traditional VAE works before you start to build the model.

- **Reference**

1. VAE reference code: https://github.com/pytorch/examples/tree/master/vae
2. Seq2seq reference code:
   https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html
   https://github.com/pytorch/tutorials/blob/master/intermediate_source/seq2seq_translation_tutorial.py
3. Generating Sentences from a Continuous Space [Samuel R. Bowman et al. 2016]
4. Auto-Encoding Variational Bayes [Diederik P. Kingma et al. 2014]
5. Natural Language Toolkit: https://www.nltk.org/