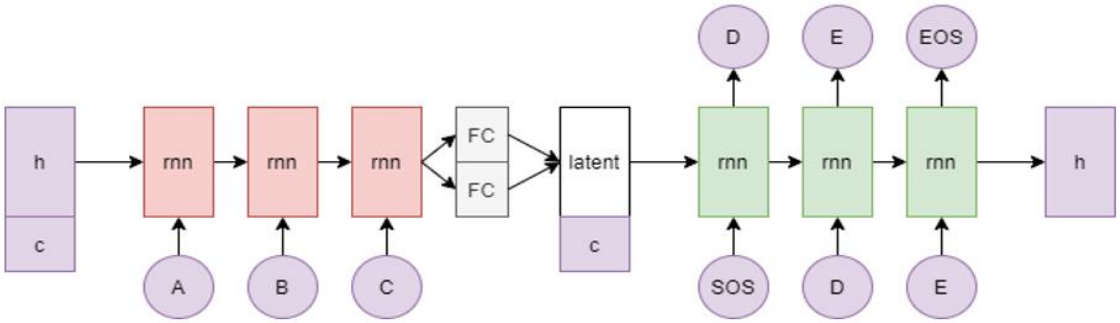
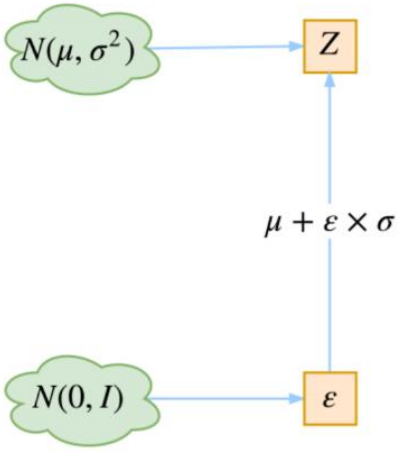
Lab4-Conditional sequence-to-sequence VAE

309552042 黃敏涓

I. Introduction (5%)



透過在 Encoder 和 Decoder 中輸入額外標籤(one-hot vector)再以非監督式學習的方式訓練出有條件的生成模型。模型中加入了 reparameterization trick(如下圖)，使 Encoder 得到的 mean 和 variance 所形成的 latent space q(z|x)能更接近 Gaussian Distribution。



並使用 KL_Divergence* KL_weight(KL cost annealing)和 Crossentropy loss 加總的 loss 做 backpropogation，以使得 latent space 的 distribution 更接近 Normal Distribution 並且防止噪聲為 0，保證模型具有生成能力。

II. Derivation of CVAE (5%)

$$\log p(x|c) = KL\left(q(z|c)\|p(z|x,c)\right) + \sum_z q(z|c)\log\frac{p(x,z|c)}{q(z|c)}$$

$$\mathcal{L}(\nu,\theta|c) = \mathbb{E}_{z\sim q}\left[\log p(x|z,\theta,c)\right] - KL(q(z;\nu|c)\|p(z))$$

c : condition

III. Implementation details (15%)

A. Encoder:

使用 lstm 來 implement Encoder, 並在最後加上一層全連接層，以計算 mean 和 log_variance。

B. Decoder:

使用 lstm 來 implement Decoder，並使用全連接層 output 出結果。

C. Reparameterization trick:

將 Encoder 產生的 mean 和 log_variance 去做 reparameterization，以此過程來優化均值方差的模型:

z=mean+ epsilon* var # z: global latent sentence representation, epsilon: 從 normal distribution 隨機抽出的一組變數

```
means, log_var = self.encoder(x, c1)

std = torch.exp(0.5 * log_var)
eps = torch.randn(self.linear_size)
z = eps * std + means
```

D. Dataloader:

Class 中包含__init__(), __len__(),__getitem__()函式

__init__(): 吃進.txt，並將.txt 中的 data 和 type(0,1,2,3 表示)以 list 形式儲存

__len__(): return data 長度

__getitem__(): return 出該 index 的 tensor 型態及 type 的 one hot encoding

E. Text Generation:

使用 Gaussian noise 採樣 100 個點出來，並由 tensor 轉成 word 後，再去計算 Gaussian score

```
for i in range(100):
    c = idx2onehot(list(range(eng.type_size)), eng.type_size) #char 2 one-hot enocding
    output = model.inference(c)
    gens = [tensor2word(get_output(o)) for o in output]
    print(gens)
    print("g_scores :", Gaussian_score(gens))
```

F. Hyper Parameter:

KLD_Weight =0 (KL annealing: step/epochs * KLW_max)

LR=0.05

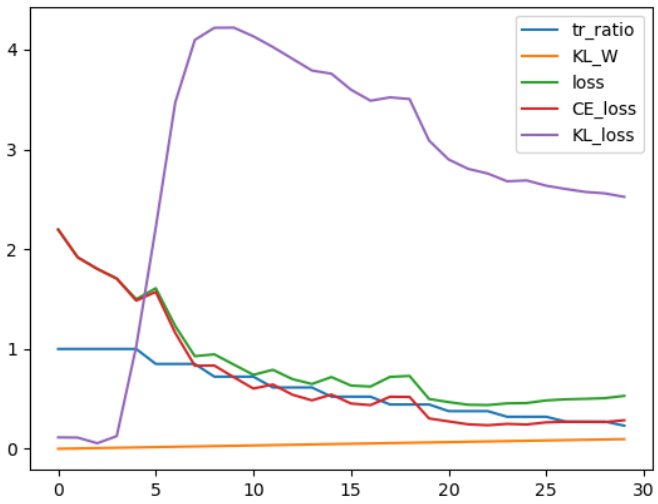
KLW_max= 0.1

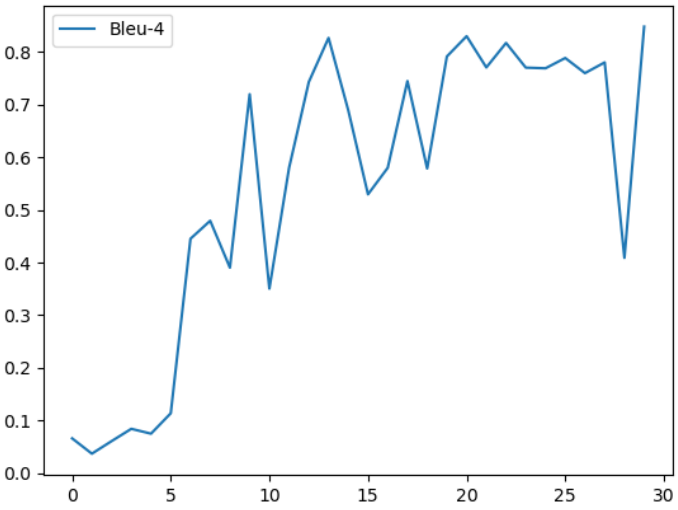
teacher_forcing_ratio=1(*=0.85)

Epoch= 30

IV. Results and discussion (25%) –

Show your results of tense conversion and generation





hidden_size	teacher_forcing_ratio	KLD_weight	LR	Result
256	*=0.85	0	0.05	Avgbleu : 0.8201 Gaussian score= 1.0
256	*=0.85	0	0.1	Avgbleu:0.7596 Gaussian score= 1.0
256	*=0.5	0.5	0.05	Avgbleu:0.7567 Gaussian score= 0.75

KL cost annealing:

Reconstruction loss KLD

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

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

引入一個權重 w 來控制 KL，並讓 w 從 0 開始隨著訓練逐漸增大。作者的意思是讓模型學會 encode 更多信息到 z 里，隨著 w 增大再 smooth encodings，如果一開始乘以很小的 w，模型就會選擇忽視 KL，選擇優先降低 reconstruction loss，當 w 慢慢增大，模型也慢慢開始關注降低 KL。若 KLD 很快地降為 0，則 lantent space 的 distribution 的 variance 還是很小，那又跟單個點差不多了。

Teacher forcing:

每 3 個 epoch 就將 teacer_forcing_ratio * 0.85