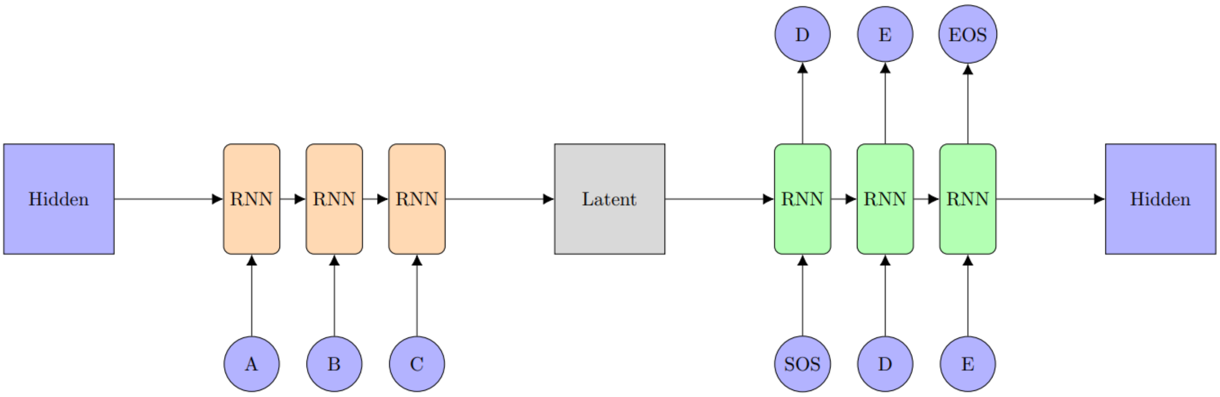
Deep Learning and Practice

Lab 5 309505018 郭俊廷

1. Introduction (5%)

用seq2seq Network檢查英文拼字錯誤並輸出正確的拼字, seq2seq有分為encoder和decoder, encoder負責壓縮資料成一個vector, decoder負責解析vector, 完成spelling correction.

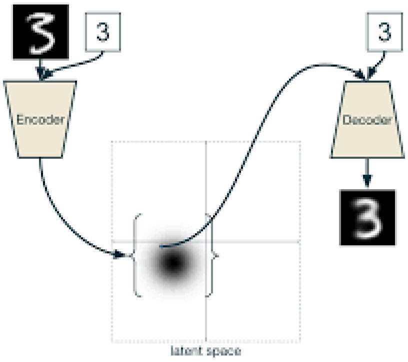


2. Derivation of CVAE (5%)

Conditional Variational Autoencoder 主要有兩個部分, 編碼器 (Encoder)和解碼器(Decoder) 首先編碼器中輸入的是真實圖像(或單字)，經過一系列卷積(Convolution)操作提取真實圖像(或單字)特徵的平均數和變異數，並將其轉化成一個特定的高斯分布，編碼器是選擇一個符合該高斯分布的一個隨機向量。 然後解碼器輸入的是編碼器中輸出的隨機向量：經過一系列的轉置卷積(Transposed Convolution)操作真實圖像(或單字), 跟一般的VAE相比因為在Encoder和Decoder都有加入控制某個變量來實現生成某一類圖像, 所以生成出來會比較穩定

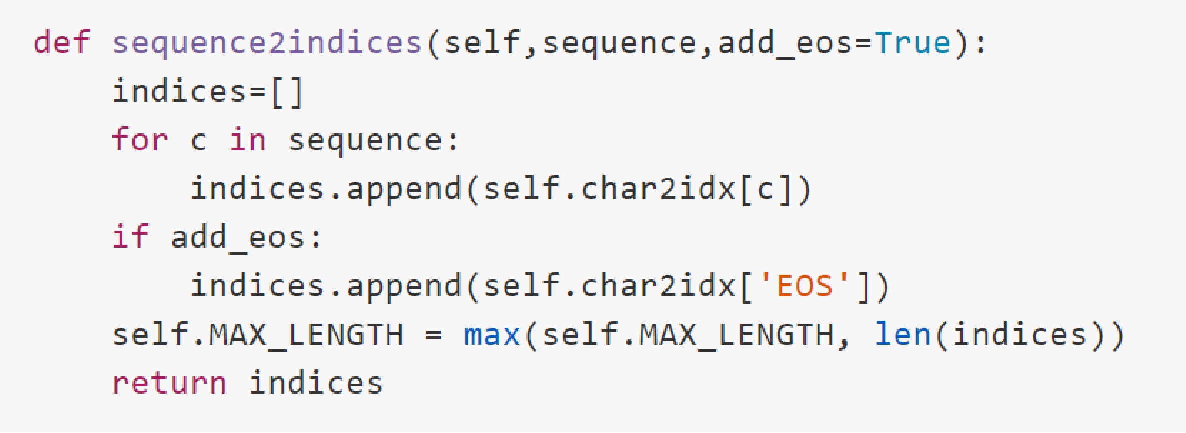
Loss Function of Conditional Variational Autoencoder :

$$E[logP(X|z,c)] - D_{KL}\ [\ Q(z|X,c)\ \|\ P(z|c)\ ]$$



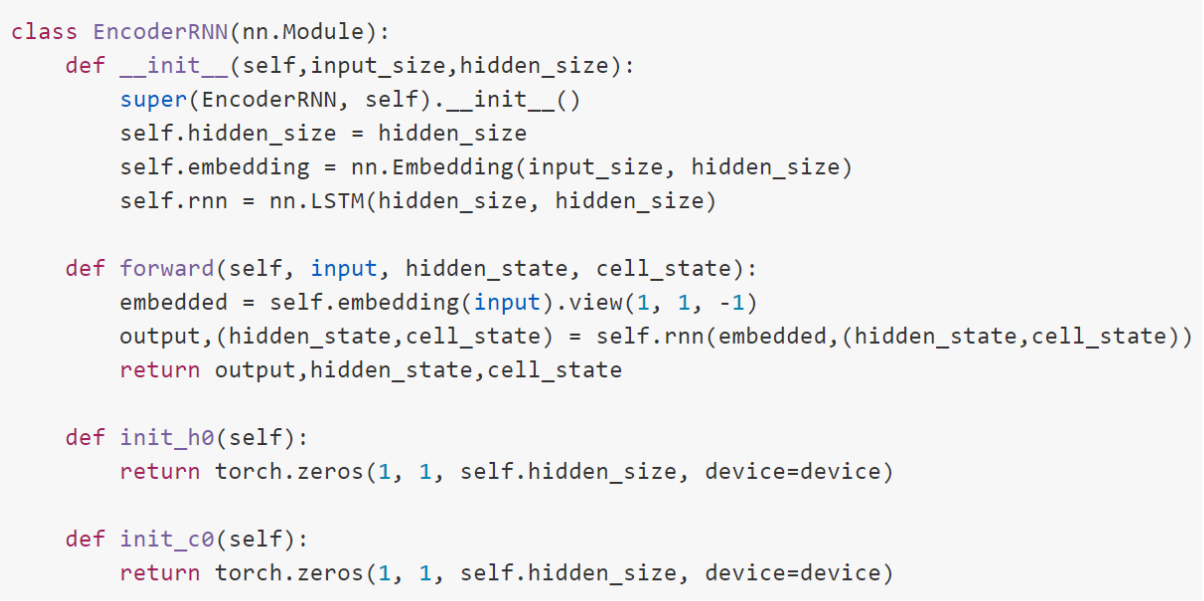
3. Implementation details (15%)

(1) 先建一個字母對應表: {'SOS':0, 'EOS’:1, ‘UNK’:2, ‘a’:3,’b’:4,‘c’:5,.......‘z’:28} SOS代表開始輸入字串, EOS代表字串結束, 如果沒有這個字母就用UNK代替, 所以如果要輸入journal -> ’journal’+'EOS' =[12,17,23,20,16,3,14,1]



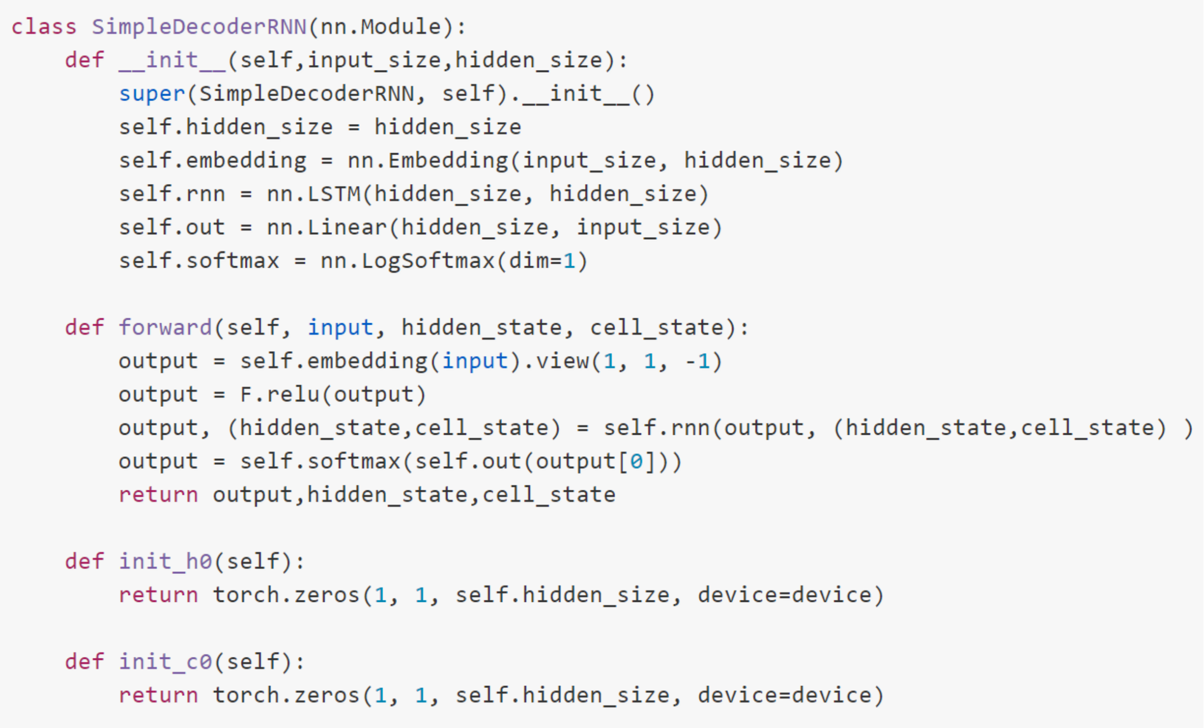
```
def sequence2indices(self,sequence,add_eos=True):
    indices=[]
    for c in sequence:
        indices.append(self.char2idx[c])
    if add_eos:
        indices.append(self.char2idx['EOS'])
    self.MAX_LENGTH = max(self.MAX_LENGTH, len(indices))
    return indices
```

(2)把一個字串變成一個vector後, 接下來把這個vector轉成tensor, 這裡可以透過torch的embedding API轉為tensor, 輸入到encoder後, encoder中的LSTM會進行多次time step forwarding, 便可以得到context vector.



```
class EncoderRNN(nn.Module):
    def __init__(self,input_size,hidden_size):
        super(EncoderRNN, self).__init__()
        self.hidden_size = hidden_size
        self.embedding = nn.Embedding(input_size, hidden_size)
        self.rnn = nn.LSTM(hidden_size, hidden_size)

    def forward(self, input, hidden_state, cell_state):
        embedded = self.embedding(input).view(1, 1, -1)
        output,(hidden_state,cell_state) = self.rnn(embedded,(hidden_state,cell_state))
        return output,hidden_state,cell_state

    def init_h0(self):
        return torch.zeros(1, 1, self.hidden_size, device=device)

    def init_c0(self):
        return torch.zeros(1, 1, self.hidden_size, device=device)
```

(3) encoder會輸出 hidden_state和cell_state, 然後再把他輸入到decoder, forward pass回傳的output就是predict結果了

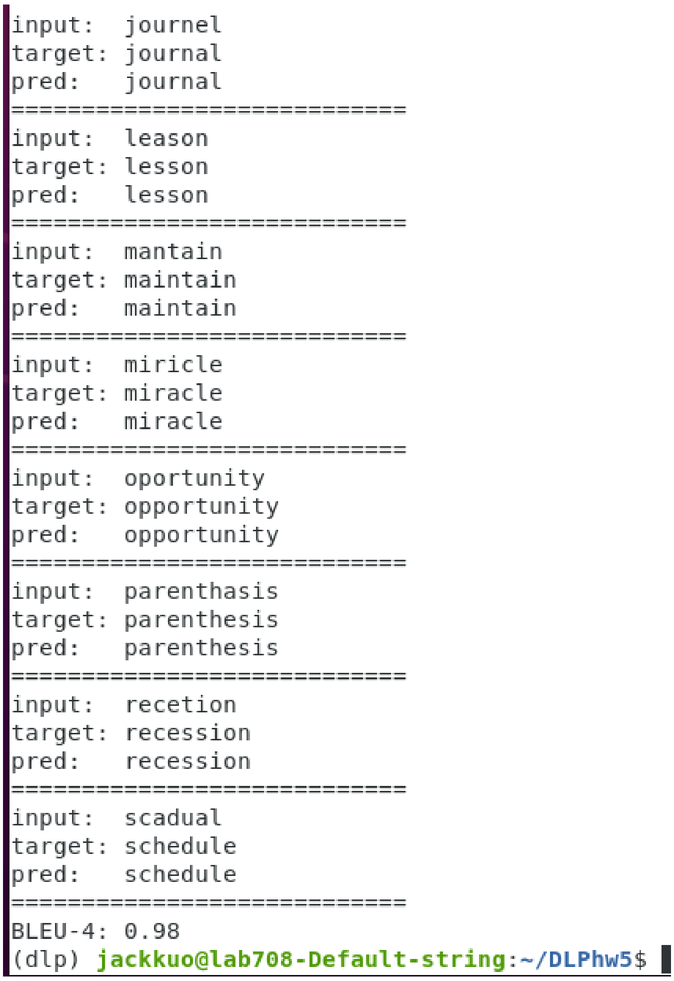


```python
class SimpleDecoderRNN(nn.Module):
    def __init__(self,input_size,hidden_size):
        super(SimpleDecoderRNN, self).__init__()
        self.hidden_size = hidden_size
        self.embedding = nn.Embedding(input_size, hidden_size)
        self.rnn = nn.LSTM(hidden_size, hidden_size)
        self.out = nn.Linear(hidden_size, input_size)
        self.softmax = nn.LogSoftmax(dim=1)

    def forward(self, input, hidden_state, cell_state):
        output = self.embedding(input).view(1, 1, -1)
        output = F.relu(output)
        output, (hidden_state,cell_state) = self.rnn(output, (hidden_state,cell_state) )
        output = self.softmax(self.out(output[0]))
        return output,hidden_state,cell_state

    def init_h0(self):
        return torch.zeros(1, 1, self.hidden_size, device=device)

    def init_c0(self):
        return torch.zeros(1, 1, self.hidden_size, device=device)
```

4. Results and discussion (25%)

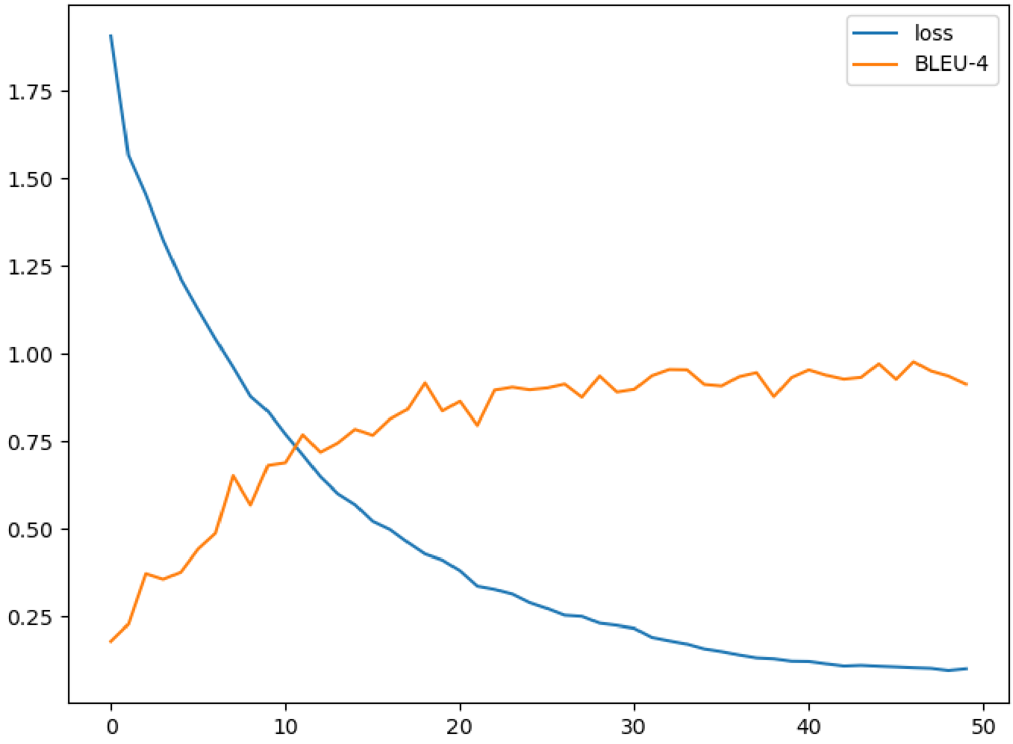
BLEU-4 score: 0.98
teacher_forcing_ratio: 0.5

我使用teacher_forcing_ratio為0.5訓練50 epoch最後BLEU-4可以達到0.98分, training loss也很平滑穩定的下降,是個很好的結果



```
input:  journel
target: journal
pred:   journal
============================
input:  leason
target: lesson
pred:   lesson
============================
input:  mantain
target: maintain
pred:   maintain
============================
input:  miricle
target: miracle
pred:   miracle
============================
input:  oportunity
target: opportunity
pred:   opportunity
============================
input:  parenthasis
target: parenthesis
pred:   parenthesis
============================
input:  recetion
target: recession
pred:   recession
============================
input:  scadual
target: schedule
pred:   schedule
============================
BLEU-4: 0.98
(dlp) jackkuo@lab708-Default-string:~/DLPhw5$
```

Training loss & BLEU-4 score


loss
BLEU-4