MLDS HW3 – Generative Adversarial Network

TensorJoe

r05922063 陳啓中 r05921032 陳昱維 r05943093 蔣君涵 d04921018 艾弗里

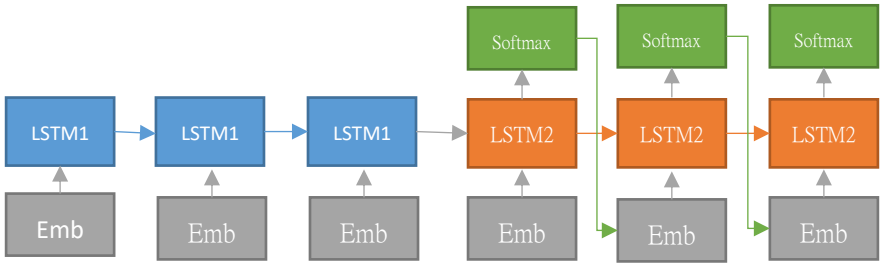
Environment

OS:	Ubuntu 14.04
CPU:	Intel Core i7-4790
GPU:	NVIDIA GTX970
Python Library:	Tensorflow r1.0
	Numpy 1.11.0
	Scipy 0.19.0
	Facebook FastText

Model description & reward functions

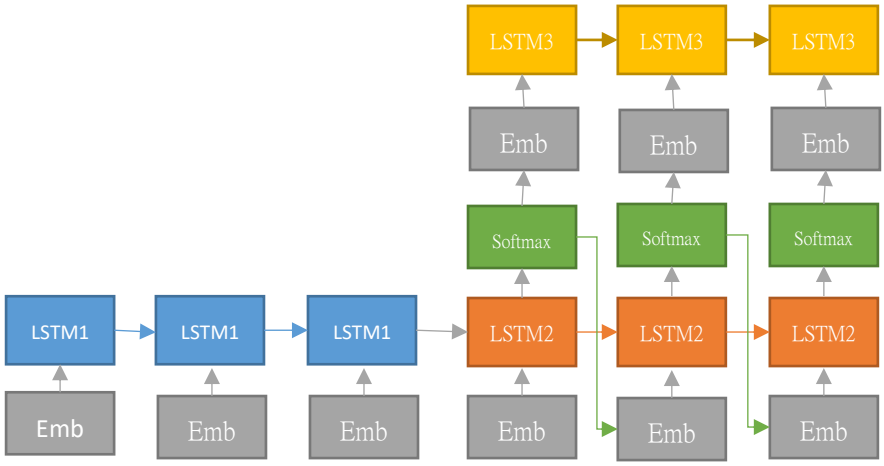
1. Sequence-to-sequence model

   我們的 S2S model 所採用的是 encoder 和 decoder 均為 LSTM，input 為 300-dimention word embedding，使用 FastText pretrained model 所產生。Decoder 中 output 為一層 softmax。我們使用最原始的 XENT 做 model 的訓練。



2. Reinforcement Learning on S2S

   我們的 reinforcement learning 是採用 actor-critic model，actor 首先先用上面的 model 做 pre-train，再用 policy gradient 做 fine-tuning，value 值來自於 critic，基本上 actor 部分的 network 架構都跟 S2S model 一模一樣。Critic 為另一個 LSTM，與 LSTM1 final state 及 LSTM2 所產生字詞的 embedding 做連接，output 為一個數字，即為 Q-value，以 Monte-Carlo Method 做訓練，reward function 使用的是 BLEU score。



How do you improve your performance

1. Training data 預處理

   Training data 我們所使用的是 cornell movie dialogs corpus，除了將對話擷取出來外，另外我們觀察到問答之中通常會夾雜無關的句子，而在問句最後一句及答句第一句才是關鍵，因此為了加速訓練速度我們將無關的句子通通丟掉。除了上述加速方法外，我們還有將僅出現 1 次的單字設為 oov，大致減少了一半的單字量；將過長(20 個單字)的句子移除，training data 大概只減少了 5%，但換來運算速度的大幅提升。

2. Baseline function

   Baseline 我們嘗試過 average 方法，也就是在每個 batch 都去取得一次 reward 的 average 當作 baseline，後來發現這會造成 value function 前後不一致的問題。後來使用 constant 的方法，效果較好。

Experiment settings and observation

Batch size:	30
LSTM1 size:	500
LSTM2 size:	500
LSTM3 size:	500
Baseline:	0.0
Learning rate:	0.0001 (Adam)
Epoch:	30 for S2S, 1 for RL

實驗結果發現 RL 方式雖然能將 dummy sentence 消除，但在訓練時間一拉長時則會將 network 破壞掉，推測是因為 RL 方法一次只會修正一條路徑，而這條路徑有著較高機率會選擇相對好的 actions，長久下來因為 critic 把這些選擇

當作 baseline，而讓 actor 對這些 action 的機率降低，造成效能一路下滑。因此在最後我們只讓 RL 跑一個 epoch，用以消除 I don’t know 這類句子。

Team Division

程式：陳啟中
報告撰寫：陳啟中