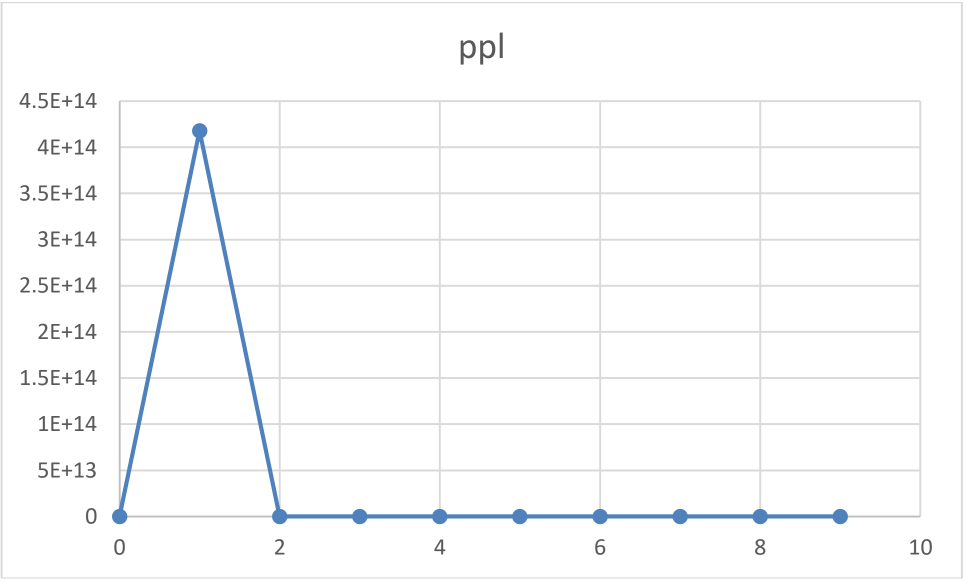
ADL HW2 report

生醫電資所 r07945029 王思敏

Q1: Describe your ELMo model.

1. Training corpus processing. (tokenization, vocabulary building)
2. Model architecture and implementation details.
3. Hyperparameters of your ELMo model. (number of layers, hidden dimension, output dimension, optimization algorithm, learning rate and batch size)
4. Plot the perplexity score on train/dev set while training.
5. Show the performance of the BCN model with and without ELMo on the public leaderboard.

After reading corpus, data will be broke to sentence first. Then the data will be used to build the vocab dictionary if the word count less than 3. Word and character dictionary will input the embedding layer padding with <pad> and filling word not existed with <oov>. The data will then be split to batches and convert to numeric sequence by word and character dictionary index. The model consists of CNN layers with kernel size 1, 2, 3, 4, 5, 6, and 7, input dim size 50*32 and output dim size 50*1024. Then the output will input to linear layer forward and backward with size 512. Adam is used for optimizer. Learning rate is 0.001, learning rate decay is 0.8, batch size is 32, maximum epoch is 10, maximum sentence length is 20, and word minimum count is 5. The performance of BCN model is only 0.42895 on public leaderboard and 0.44072 on private leaderboard while BCN model with ELMo reaches 0.46334 on public leaderboard and 0.48506 on private leaderboard.



Q2: Compare different settings for ELMo.

1. Different number of training steps.
   You can train one model for large number of training steps, then take the intermediate checkpoints for this problem.
2. Different hyperparameters.
   You can train a smaller ELMo model and compare it to your original model.

With 100 records as the training data:
When the max training epoch is too high, the ppl will begin to increase which means confusion. With 100 records as training data and 20 epoch, ppl begins to increase or oscillate after epoch=17 (In my case, oscillation around 7.45±0.2).
Decrease the max training epoch seems no influence on the training result while increate the word min count will cause a lower training speed and confusion i.e. higher perplexity.

min count 3	epoch	0	1	2	3	4	5	6	7	8	9
	ppl	26.22	5.05E+10	35.88	12.85	9.71	9.21	8.34	8.06	8	8

min count 8	epoch	0	1	2	3	4	5	6	7	8	9
	ppl	14.51	4.18E+14	1087604	157.58	18.98	14.66	10.08	8.69	8.03	8.03

min count 8	epoch	0	1	2	3	4	5
	ppl	14.51	4.18E+14	1087605	157.58	18.98	14.66

Q3: Describe your model that passes strong baseline.

1. Input of your model.
2. Model architecture.
3. Hyperparameters of your model.

Input of model is embedded by glove.840B.300d.txt. Character embedding dim is 50, and word embedding dim is 300. Model architectures are elmo, bert, and elmo stacked with bert from flair package. Result will be predicted from ensembling the results of the architectures above. Optimization algorithm is adam. Learning rate is 1.0e-3 Batch size is 32. With different kernel size, size equal to 3 will get better result (0.52) than 1 (0.50) on validation set, while the ensembling result included these two models at the same time will get better result popularly on test set (0.53~0.54). 1.

Q4: Describe your best model.

1. Describe your best model
   a. Input to your model
   b. Model architecture
   c. Hyperparameters (optimization algorithm, learning rate, batch size and other model-specific options)
2. Describe the reason you think why your best model performs better than other models.

The best model is the same as strong. Input of model is embedded by glove.840B.300d.txt. Character embedding dim is 50, and word embedding dim is 300. Model architectures are elmo, bert, and elmo stacked with bert from flair package. The score of prediction of best model with best.sh is about 0.54. It’s the outcome of ensembling 7 prediction results. While the score of best model is over 0.55 and is the ensembling result of 9 prediction results which requires more than 20 mins to make a prediction. The models of 9 predictions included elmo, cased bert, uncased bert, and elmo stacked with uncased bert. Also, kernel size shows effect on prediction result.

Q5: Compare different input embeddings.

Word embedding such as the word2vec algorithm can get a numeric representation of a single term. However, the term vector model for training a set of data can be very expensive and using the pre-trained term vector model, such as the Glove vectors may run into a memory deficit when loading on a less powerful computer.

Fewer vectors need to be calculated with character embedding, and these vectors can be loaded into a log and derive unit / sentences / paragraphs / documents vectors in turn. However, training characters are more expensive in calculation during embedding because the number of characters is about 5 times more than that of tokens. Byte-pair encoding is intended primarily for data compression. The advantage of BPE is that it can balance the size of the word list and number of tokens needed to be used in sentences. The disadvantage is that it does not provide multiple probability of segmentation.

Q6: BERT

1. Please describe the tasks that BERT used for pre-training. What benefits do they have comparing to language model pretraining used in ELMo?

2. Please describe in detail how you would formulate the problem in HW1 and apply BERT on it.
   BERT works similar to ELMo while being able to consider both the upstream and downstream contexts with covering part of words by mask model. ELMo can only consider the upsrean contexts. To imply BERT in HW1, we can replace the embedding matrix with an embedder of BERT model.