CSE 517: Homework 4 - Recurrent Neural Networks, Attention, and Reading Comprehension

Lilly Kumari

Problem 1. CBOW Model

Solution The CBOW model doesn't seem to work.

In CBOW (common bag-of-words) model, the word vectors for a particular sentence are just averaged to get an embedding (representation) for the sentence. So, the final embedding is not able to capture the word order, i.e. XYZ will have the the same embedding as YXZ. There's no concept of memory in Vanilla CBOW model, so it fails at capturing important contextual information. Also, since it just averages out the word embeddings for a particular sequence (sentence), it's incapable of identifying which words matter/weigh more given the context.

Figure 1 and 2 show the loss on training data and validation set respectively. Figure 3 and 4 show the EM score and F1 score over the validation dataset as training progresses. The hyperparameters chosen for the training are:

- num_steps = 20000
- batch_size = 64
- word_count_th = 10
- hidden_size = 50
- init_lr = 0.5
- max_context_size = 150
- max_ques_size = 15

Table 1 shows the performance of CBOW model on the test dataset.

Loss	**acc1**	**acc2**	**EM**	**F1**
8.5103	5.432%	5.484%	3.389%	10.101%

Table 1: CBOW Model: Performance on Test dataset

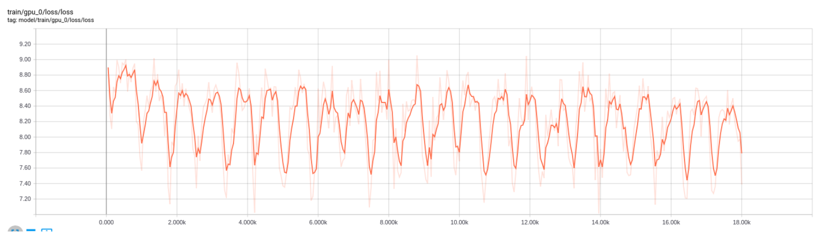

Figure 1: CBOW: Loss on Training Dataset

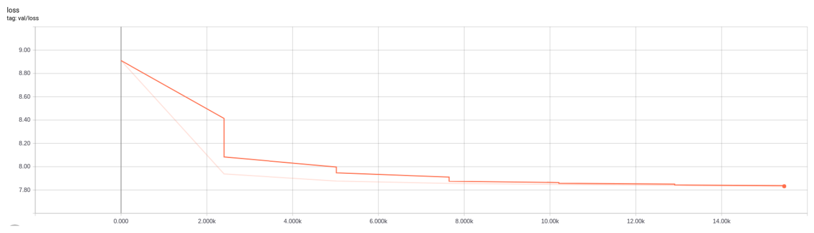

Figure 2: CBOW: Loss on Validation Dataset

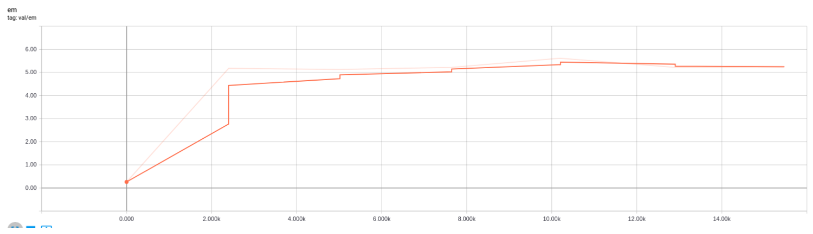

Figure 3: CBOW: Validation Dataset - EM score

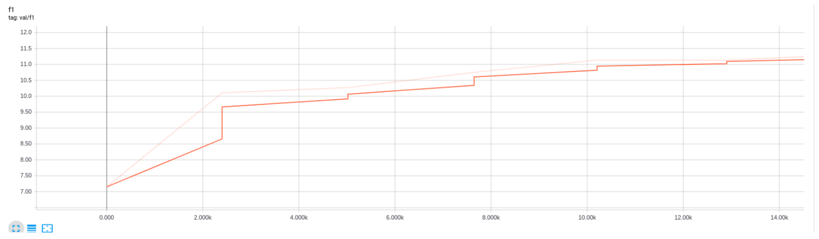

Figure 4: CBOW: Validation Dataset - F1 score

Problem 2. RNN Model (with GRU)

Solution The hyperparameters chosen for the training of the RNN Model are:

- num_steps = 20000

- batch_size = 64
- word_count_th = 10
- hidden_size = 50
- init_lr = 0.5
- max_context_size = 150
- max_ques_size = 15
- Dropout Rate = 0.1, 0.3, 0.5 (so, keep_prob = 0.9, 0.7, 0.5)
- Separate GRUcells used to derive hidden state representations of questions and paragraphs.

Plots for dropout rate of 0.1 Figure 5 and 6 show the loss on training data and validation set respectively. Figure 7 and 8 show the EM score and F1 score over the validation dataset as training progresses.

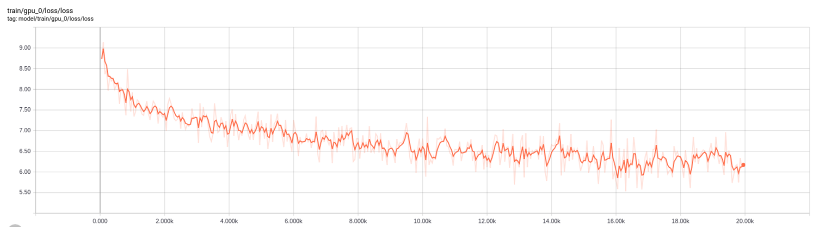

Figure 5: RNN (dropout 0.1): Loss on Training Dataset

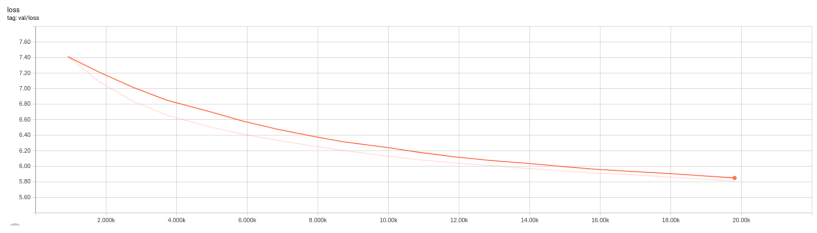

Figure 6: RNN (dropout 0.1): Loss on Validation Dataset

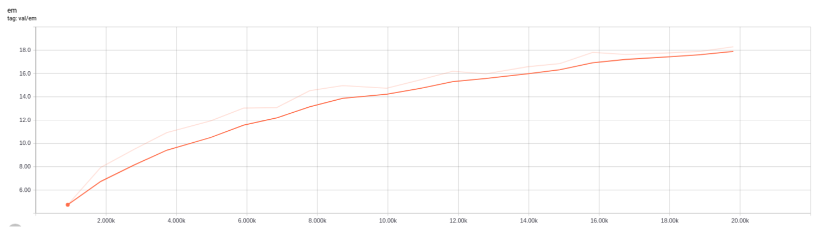

Figure 7: RNN (dropout 0.1): Validation Dataset - EM score

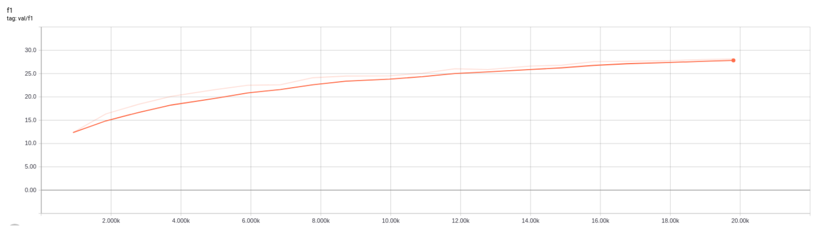

Figure 8: RNN (dropout 0.1): Validation Dataset - F1 score

Plots for dropout rate of 0.3 Figure 9 and 10 show the loss on training data and validation set respectively. Figure 11 and 12 show the EM score and F1 score over the validation dataset as training progresses.

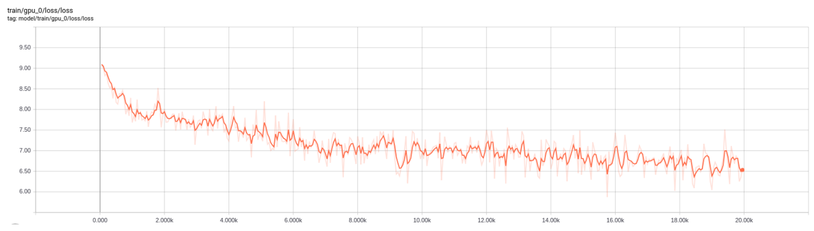

Figure 9: RNN (dropout 0.3): Loss on Training Dataset

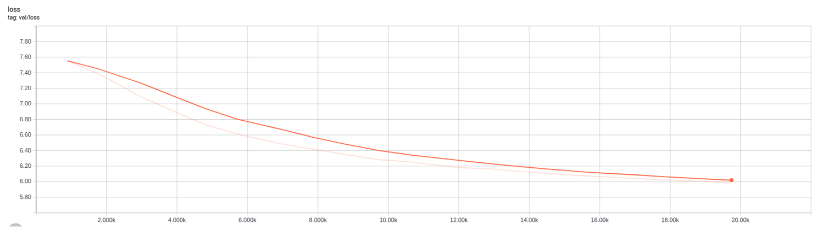

Figure 10: RNN (dropout 0.3): Loss on Validation Dataset

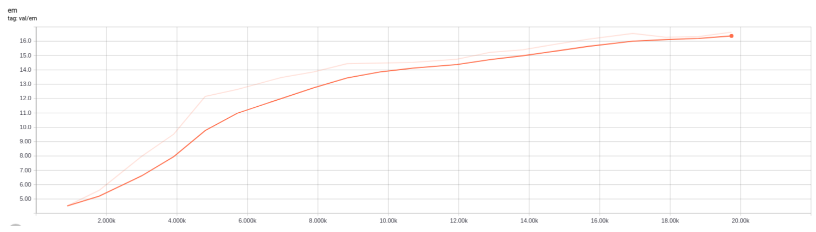

Figure 11: RNN (dropout 0.3): Validation Dataset - EM score

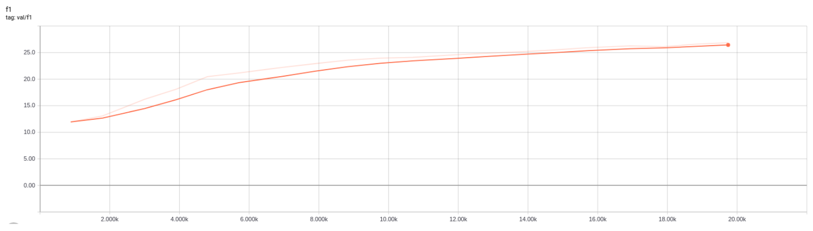

Figure 12: RNN (dropout 0.3): Validation Dataset - F1 score

Plots for dropout rate of 0.5 Figure 13 and 14 show the loss on training data and validation set respectively. Figure 15 and 16 show the EM score and F1 score over the validation dataset as training progresses.

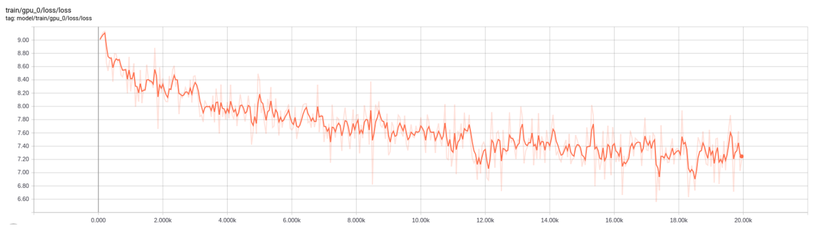

Figure 13: RNN (dropout 0.5): Loss on Training Dataset

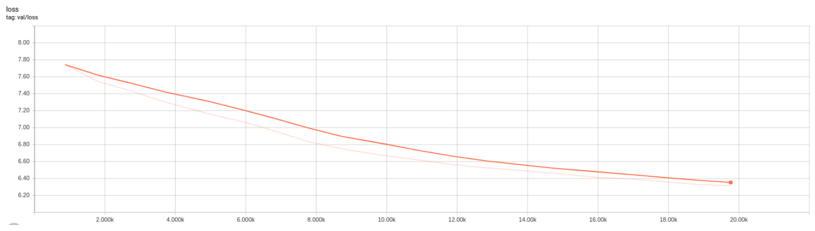

Figure 14: RNN (dropout 0.5): Loss on Validation Dataset

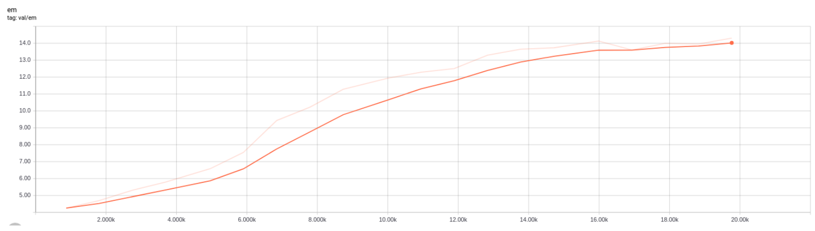

Figure 15: RNN (dropout 0.5): Validation Dataset - EM score

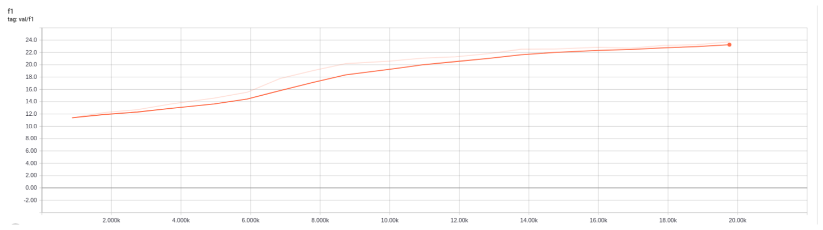

Figure 16: RNN (dropout 0.5): Validation Dataset - F1 score

Table 2 shows the performance of RNN model (with GRU) on the test dataset against different dropout rates.

dropout rate	Loss	acc1	acc2	EM	F1
0.1	6.4714	17.719%	21.935%	16.167%	26.724%
0.3	6.6701	15.856%	20.150%	13.890%	24.202%
0.5	7.0140	13.632%	16.787%	11.511%	20.413%

Table 2: RNN model: Performance on Test dataset

As the dropout rate increases (keep_prob = 1 - dropout; keep_prob decreases), the performance of the RNN model goes down. This is because as the network is small with less number of weights to be learnt, the model performs well when the dropout is in the range 0.1 - 0.3. As further neurons get dropped, the model underfits.

Problem 3. Attention Model

Solution The hyperparameters chosen for the training of the Attention Model are:

- num_steps = 20000
- batch_size = 64
- word_count_th = 10
- hidden_size = 50
- init_lr = 0.5
- max_context_size = 150
- max_ques_size = 15
- Dropout Rate = 0.1 (so, keep_prob = 0.9)
- For learning attention weights, weight matrix $\mathcal{W}^{\mathcal{P}}$ initialized with random values from a normal distribution with mean 0.0 and std deviation of 0.1
- Equation 10 given in the homework was used derive the attention weights

- Separate GRUcells used to derive hidden state representations of questions and paragraphs.

Figure 17 and 18 show the loss on training data and validation set respectively. Figure 19 and 20 show the EM score and F1 score over the validation dataset as training progresses.

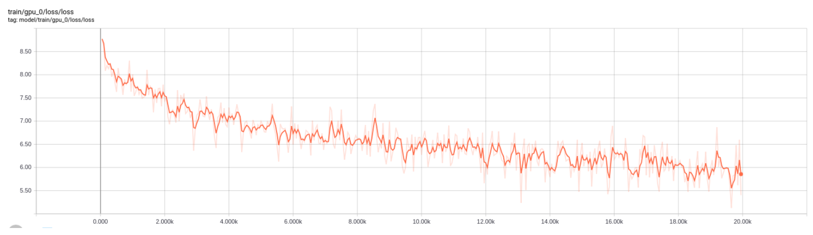

Figure 17: Attention: Loss on Training Dataset

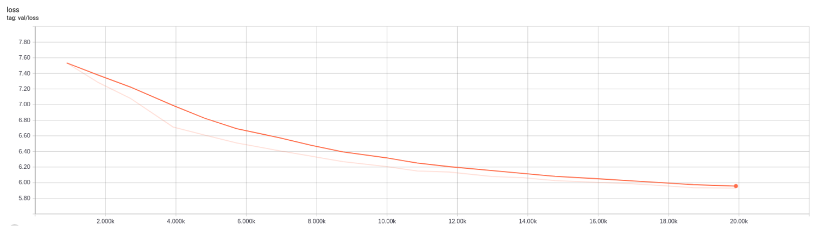

Figure 18: Attention: Loss on Validation Dataset

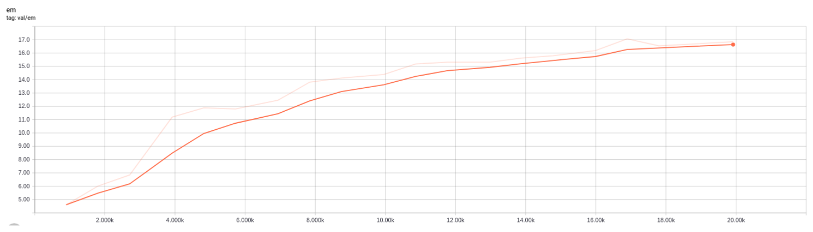

Figure 19: Attention: Validation Dataset - EM score

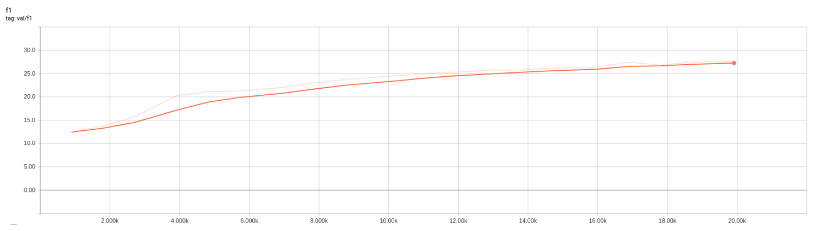

Figure 20: Attention: Validation Dataset - F1 score

Table 3 shows the performance of Attention model (with GRU) on the test dataset.

Loss	**acc1**	**acc2**	**EM**	**F1**
6.3682	19.271%	22.452%	17.098%	28.097%

Table 3: Attention Model: Performance on Test dataset

Problem 4. Bonus Problem

Solution Tried an attention model with LSTM cell instead of GRU cell and the following hyper-paramters:

- max_context_size = **200**
- Dropout Rate = **0.1** (keep_prob = 0.9)
- num_steps = 20000
- batch_size = 64
- word_count_th = 10
- hidden_size = 50
- init_lr = 0.5
- max_ques_size = 15

Figure 21 and 22 show the loss on training data and validation set respectively. Figure 23 and 24 show the EM score and F1 score over the validation dataset as training progresses.

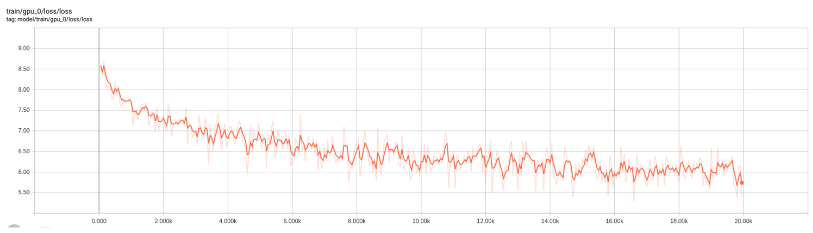

Figure 21: Attention Model using LSTM : Loss on Training Dataset

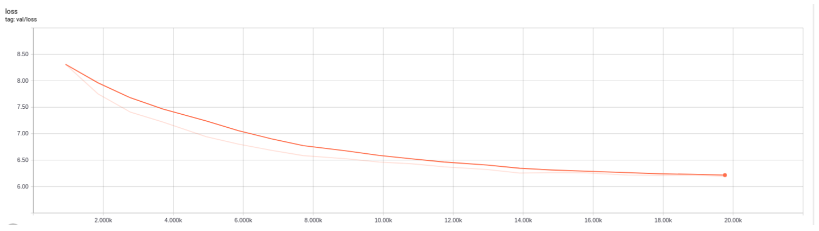

Figure 22: Attention Model using LSTM: Loss on Validation Dataset

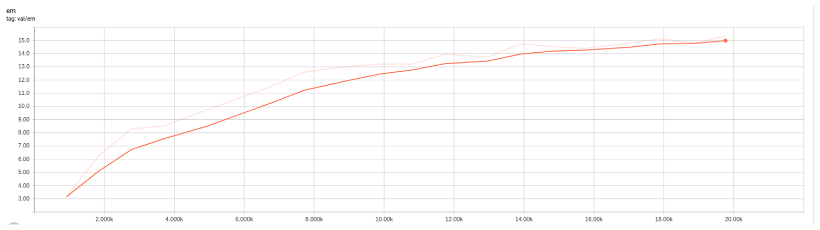

Figure 23: Attention Model using LSTM: Validation Dataset - EM score

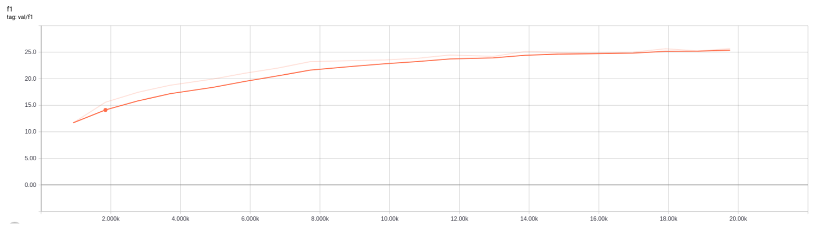

Figure 24: Attention Model using LSTM: Validation Dataset - F1 score

Performance Improvement on test dataset as compared to Ques 2 and 3 is 5% from F1 score (best attained till Q3)of 28.097% to F1 score of 29.529%

Table 4 shows the performance of Attention model (with LSTM) on the test dataset.

Loss	**acc1**	**acc2**	**EM**	**F1**
6.2808	19.400%	23.823%	17.719%	29.529%

Table 4: Attention Model with LSTM (max context size = 200 and dropout = 0.1): Performance on Test dataset