NLP ASSIGNMENT 2

SECTION 1: MODEL IMPLEMENTATION

- DEEP AVERAGING NETWORK:

As mentioned in the given paper,
The deep averaging network (DAN), works in three simple steps:
1.take the vector average of the embeddings associated with an input sequence of tokens
2.pass that average through one or more feedforward layers
3.perform (linear) classification on the final layer's representation

I first initialized the layers of the DAN Network in the init part 'num_layers' number of times using 'tf.keras.layers.Dense'. Then I computed a probability matrix of size 'sequence_size' using 'tf.random.uniform' and then I chose the instances where probability is greater than the dropout value i.e. 0.2. I got a mask from that and then I masked 'vector_sequence' with that mask.
In order to compute the average that needs to be passed to the first layer of the network, I took a sum of the resultant above to be used as the numerator. Then I counted the number of 1's in the 'sequence_mask' to get the denominator. Then I computed the average and stored it a list and passed it to the first layer of my network. Then I ran a loop and passed the output of the first layer to the second layer and so on as is typically done in a DAN Network. Then I returned the last layer of the final output as 'combined_vector' and the entire stack of the final output as the 'layer_representations'.

- GATED RECURRENT UNIT:

As taught during class, we use GRUs as an improvement over LSTM as LSTM has a large number of gates that need to be computed and GRU reduces the number of gates to just 2- update gate and the reset gate. These gates basically control the amount of information going to the output and since there are only two gates or vectors that need to be computed it is an improvement over LSTM. In my implementation, I first initialized the GRU layers in the init 'num_layers' number of times using 'tf.keras.layers.GRU'. Then in the call part, I first passed the 'vector_sequence' and mask 'sequence_mask' to the first layer and then I passed that output to the above layers in a loop. I then extracted the first and the third dimensions from the output as I wanted the resultant size to be [64,50] and then appended it to a list and returned 'combined_vector' and 'layer_representations'.

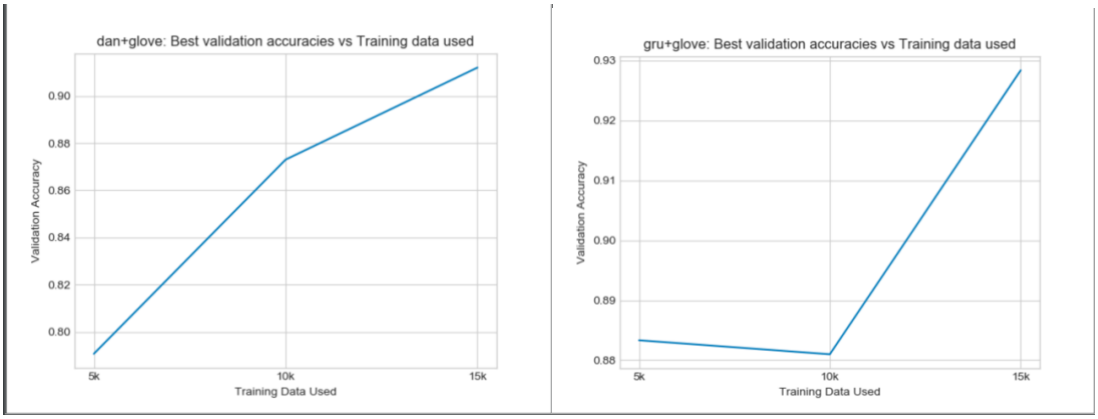
- PROBING MODEL:

The probing model basically helps us analyze the performance of the model layer by layer and I implemented that by initializing the layer using 'tf.keras.layers.Dense' and using 'softmax' as my activation function. I then called the pretrained model by giving 'inputs' as the input and then I extracted the 'layer_num' layer as passed in the function and applied it to my probing layer which I had initialized earlier and stored it in logits.

SECTION 2: ANALYSIS

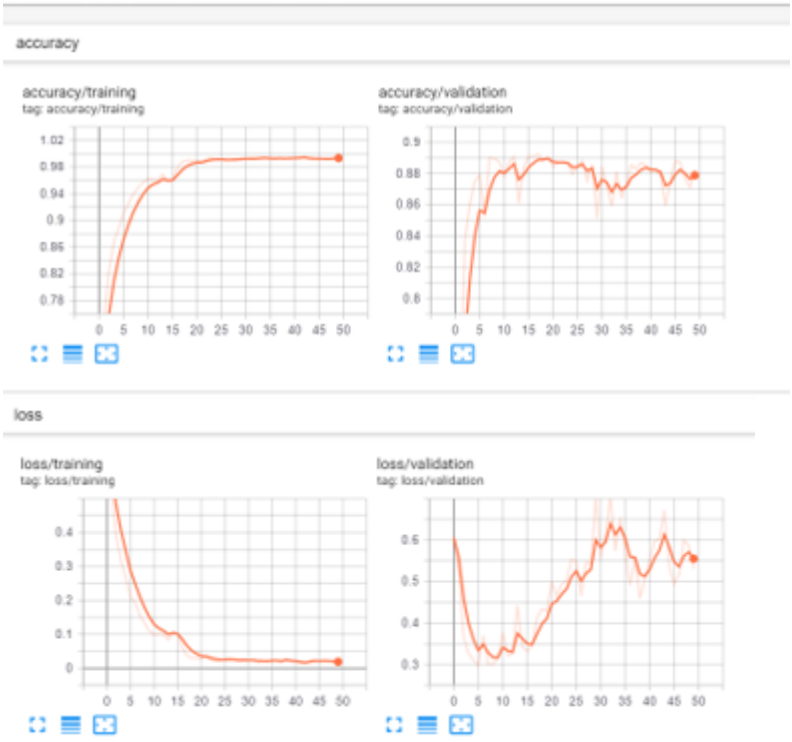
Learning Curves:

- Increasing the Training data:



As can be seen from the plots, DAN's performance increases steadily as we increase the data size whereas GRU starts good but then its performance dips a little with the 10k imdb file and then increases suddenly.

- Increasing the Training Time:



Increasing the number of epochs for DAN has the following effects on:

1. Accuracy: For training it increases till some point and then becomes constant after a while but for validation it starts increasing but then there is a lot of variation in the region where training loss was constant.
2. Loss: For training loss, the graph actually looks like the reverse of the training accuracy graph and it decreases steadily. Validation loss on the other hand fluctuates a lot after decreasing at the start.

Error Analysis:

- Advantage of DAN over GRU:

  DAN takes lesser training time than GRU which can also be seen during the implementation of the assignment and it also has better sentiment accuracy than GRU. Also, from the above plot on increasing the training data we can understand that it handles large training data much better than GRU and doesn't let large size affect its performance.

- Advantage of GRU over DAN:

  Since GRU doesn't compute average at any point preserves the important information from the input which may get lost in the case of DAN. GRU also doesn't suffer from the vanishing/exploding gradient problem. It also performs better in bigram ordering task than DAN. GRU also considers sequential information of the input which is ignored in DAN.

- Case where DAN fails:

  In sentences which involve double negatives, DAN fails to perform well and in most cases doesn't predict the correct sentiment that was intended.
  Example: 'I didn't do nothing'
  Here, didn't and nothing cancel each other's effect but DAN is not able to understand this.

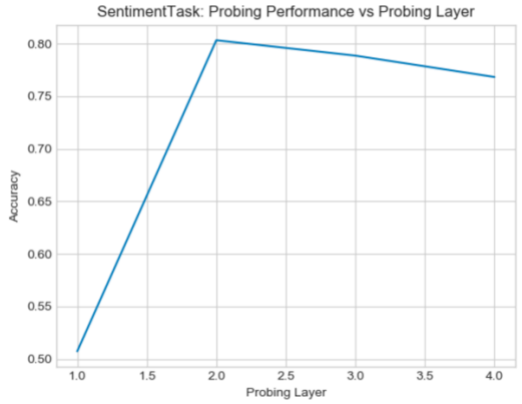
- Case where GRU fails:

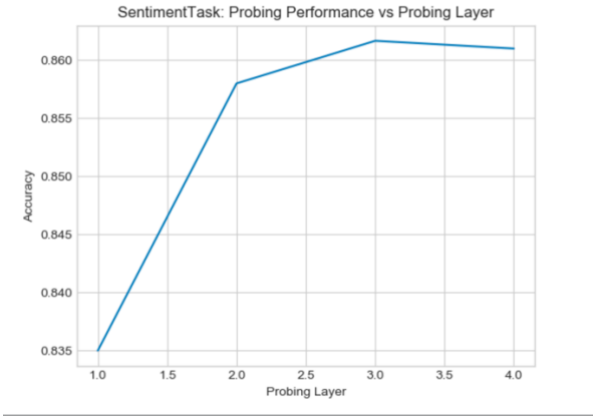
  GRU tends to make mistakes as sequences get longer.
  Example: 'The movie that Shireen told me to watch the other day at her place when we were planning to go camping turned out to be awesome.'

  This sentence is really long and the most important sentiment is mentioned at the end and the subject is mentioned at the very beginning of the sentence.

SECTION 3: PROBING TASKS:

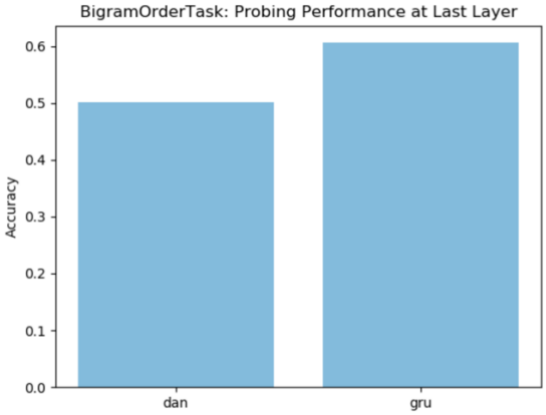
- Probing Sentence Representation for Sentiment Task





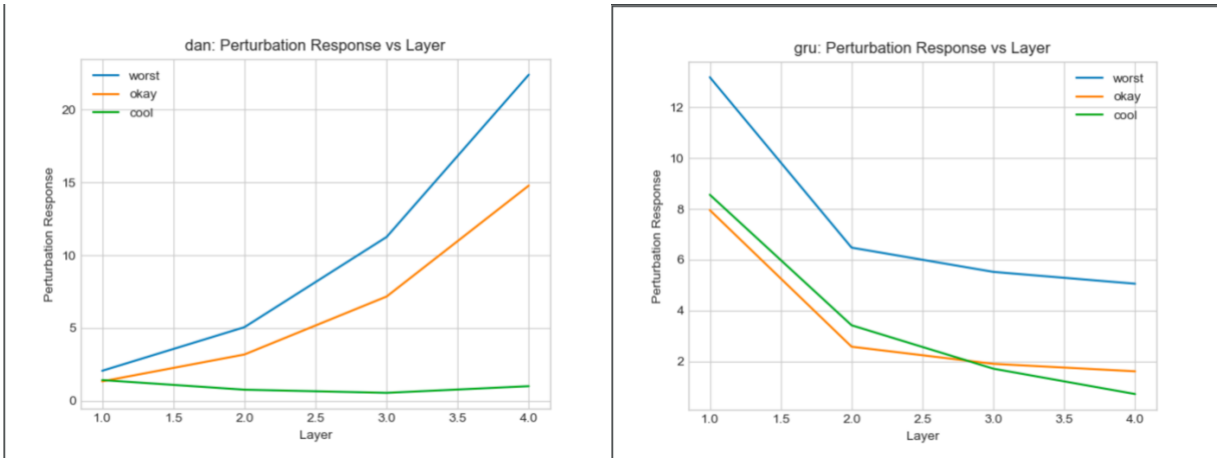
The first plot is for DAN and the second plot is for GRU. For DAN we can see that the accuracy of the model increases linearly till layer 2 and then starts decreasing gradually. GRU on the other hand has a much steadier performance and accuracy increases from one layer to the next and becomes constant around layer 4.

- Probing Sentence Representations for Bigram Order Task



From the above plot we can observe that for the Bigram Order task, probing performance at the last layer is much better for GRU than DAN and has 60% accuracy in comparison to 40% accuracy performance of DAN.

- Analyzing Perturbation Response of Representations:



Perturbation response helps us in understanding how small changes get magnified as the input is passed between different layers. In the case of DAN, the response keeps on gradually increasing as the input is passed to the layers. We also see that the distance between the green and orange line is large as there is a difference in sentiment there. Same is the case with the other two lines. In the case of GRU however, the performance is opposite, and the response goes on decreasing. Also, there is little to no distance difference between the orange and green lines as it is not able to distinguish that much between positive and neutral sentiment. But there is still a significant difference between the blue line and the other two lines, so it detects negative words much better.