REPORT- SENTENCE REPRESENTATION

SBU ID: 112673842

1. MODEL IMPLEMENTATION

1.1. DAN

Under init method, layers of DAN model are initiated using tf.keras.Layers.Dense with activation method relu. Then a matrix is created with random values from uniform distribution. Now, after comparison of all the elements in the matrix with dropout value, a new matrix is obtained with True's and False's based on whether they are greater than dropout value or not. Now convert this into a Boolean matrix (0's and 1's). In order to retain the zeros in the earlier matrix, we need to multiply the Boolean matrix with the sequence_mask matrix to obtain a new sequence_mask matrix. This sequence_mask obtained should be reshaped to get the same shape as vector_sequence using reshape and transpose methods in tensorflow. Vector _sequence is multiplied with the transpose of the 3d matrix obtained. Reduce sum of vector_sequence is done column wise to obtain a 1d matrix. Now a vector_sequence sum is obtained by using reduce_sum which is further used to get vector_sequence average. This vector_sequence average is given as the input to the first layer of the dense network. A list is created and all the outputs of each layer are appended to this. Combined vector will have the output from the last layer. Layer representation is the transpose of the stack of the list containing all the outputs.

1.2. GRU

Under init method, layers of GRU model are initiated using tf.keras.Layers.GRU with return_sequence and return_state as True. Now a list of outputs and states are created. The first layer of GRU is vector_sequence with sequence_mask as the mask. The outputs from the previous layer are passed to the next layer as input. Combined vector will have the state from the last layer. Layer representation is the transpose of the stack of the list containing all the states.

1.3. PROBING MODEL

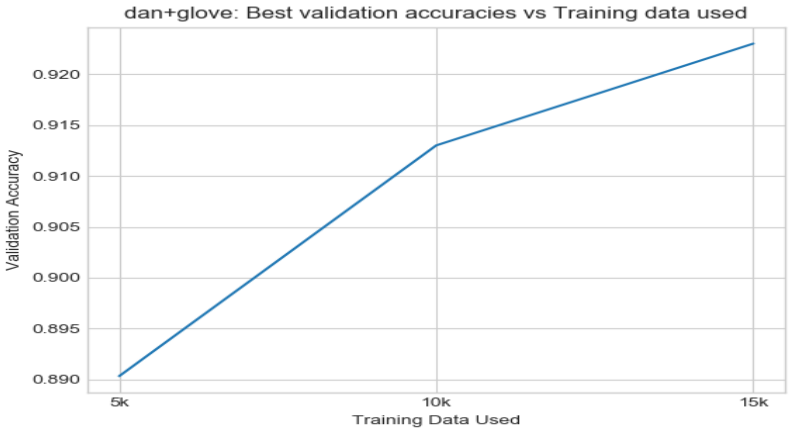
Under init method, a layer of DAN model is initiated using tf.keras.Layers.Dense with activation method softmax. Fetch the value of layer_representations from the dictionary consisting on combined_vector and layer_representations using the pretrained model. Later we need to obtain the logits for a particular layer number(in this case, 3)

2. ANALYSIS

2.1. LEARNING CURVES

2.1.1. Increasing the training data

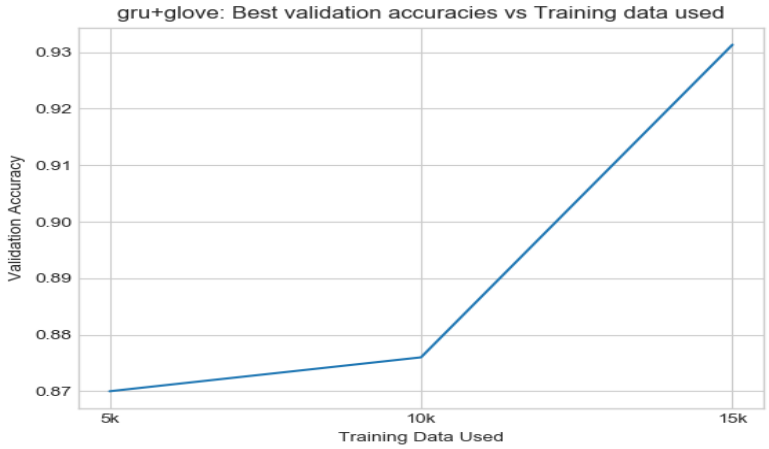
- ***DAN:***



Training Data Used	Validation Accuracy
5k	0.890
10k	0.912
15k	0.923

In DAN model, as we increase the training data, validation accuracy also increases. Here, each layer learns a more abstract representation of the given input than the previous layer, thus improving the accuracy in every representation. The depth allows it to capture subtle variations in the input to get a better accuracy.

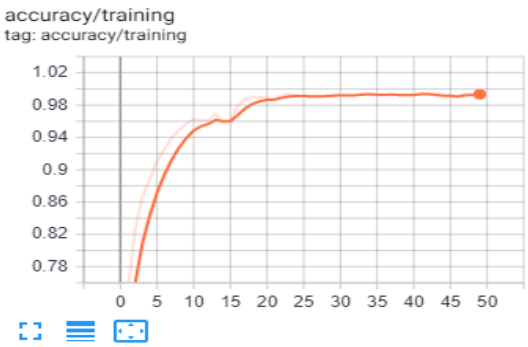
- ***GRU:***

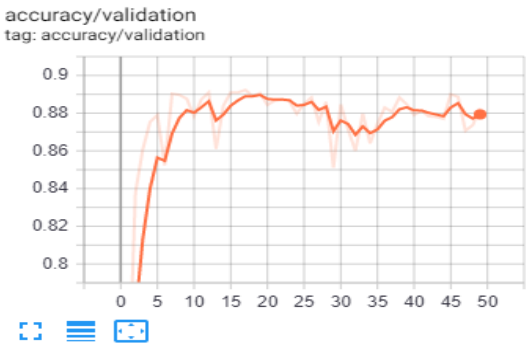


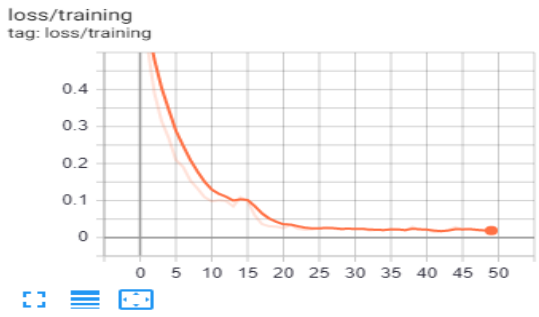
Training Data Used	Validation Accuracy
5k	0.87
10k	0.876
15k	0.932

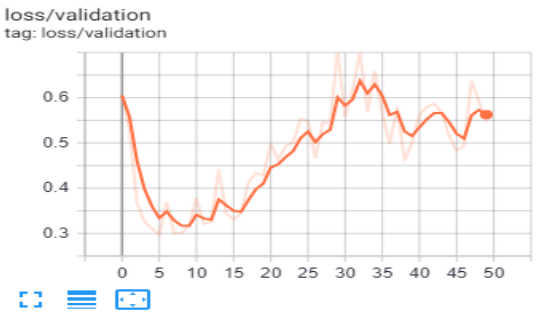
In GRU model, as we increase the training data, validation accuracy also increases. It's also evident that at every training data level, GRU has a better accuracy than DAN. This is because convergence gives better solutions.

2.1.2. Increasing training time (number of epochs)









As the number of epochs increased in DAN model during training, the loss value has decreased almost steadily. The loss in validation has fluctuated during the validation The accuracy of both training and validation model almost remained the same.

2.2. ERROR ANALYSIS

2.2.1. Advantage of DAN over GRU and Advantage of GRU over DAN

Advantage of DAN over GRU	Advantage of GRU over DAN
As the layers increase in DAN, the smaller differences between sentences are magnified.	DAN doesn't consider the order of the words in a sentence but GRU does.

2.2.2. Failure cases of GRU that DAN can get right and vice-versa

Failure cases of GRU that DAN can get right:

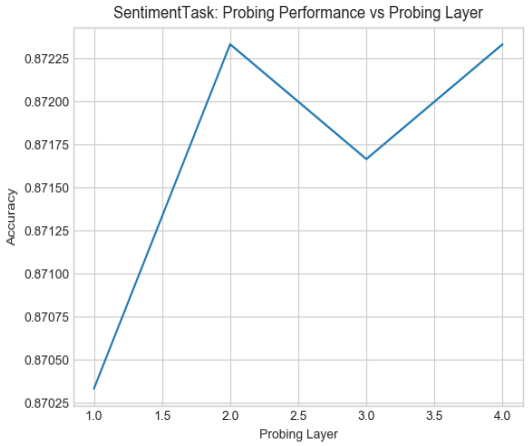
From the IMDB reviews, DAN would predict the below line as negative but GRU wouldn't:
"I wonder if the surrounding politics had something to do with trying to make a movie for all tastes but ending up with something that pleases no one."

Failure cases of DAN that GRU can get right:
*"I would read the book but would **not** suggest it"*
*"I would **not** read this book but would suggest it"*
The only difference between these two sentences is the placement of the word "not" in the sentence. Since, DAN would not consider the ordering it would not capture the actual difference between these two sentences whereas GRU would.

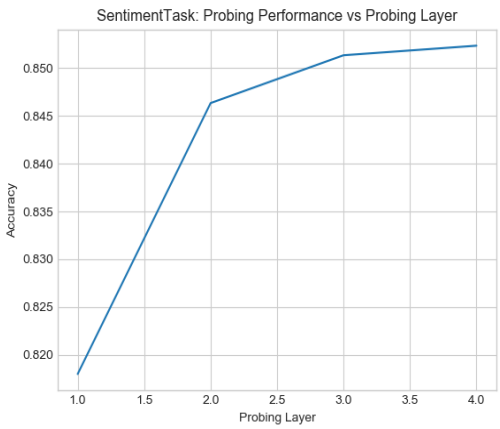
From the IMDB reviews, GRU would predict the below line as negative but DAN wouldn't:
"*The only reason i am bothering to comment on this movie is to save you all 97 minutes of your life and maybe your money.I bought it ex-rental for 33.00, it looked interesting, so i took a chance. Within minutes of turning it on i realised i'd made a mistake."*

3. PROBING TASKS

3.1. PROBING SENTENCE REPRESENTATION FOR SENTIMENT TASK



DAN

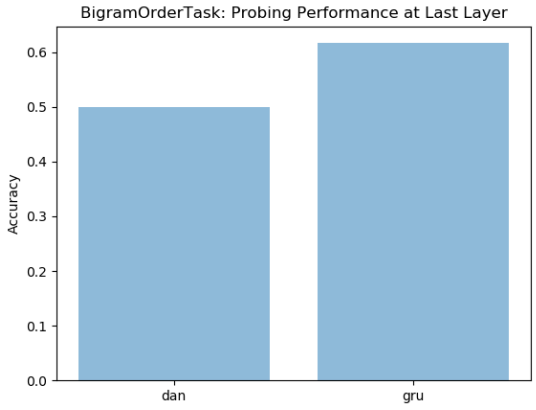


GRU

DAN		GRU	
Probing Layer	Accuracy	Probing Layer	Accuracy
1	0.87025	1	<0.820
1.5	0.87127	1.5	0.832
2	0.87227	2	0.847
2.5	0.87200	2.5	0.849
3	0.8715	3	0.852
3.5	0.87200	3.5	0.853
4	0.87227	4	0.854

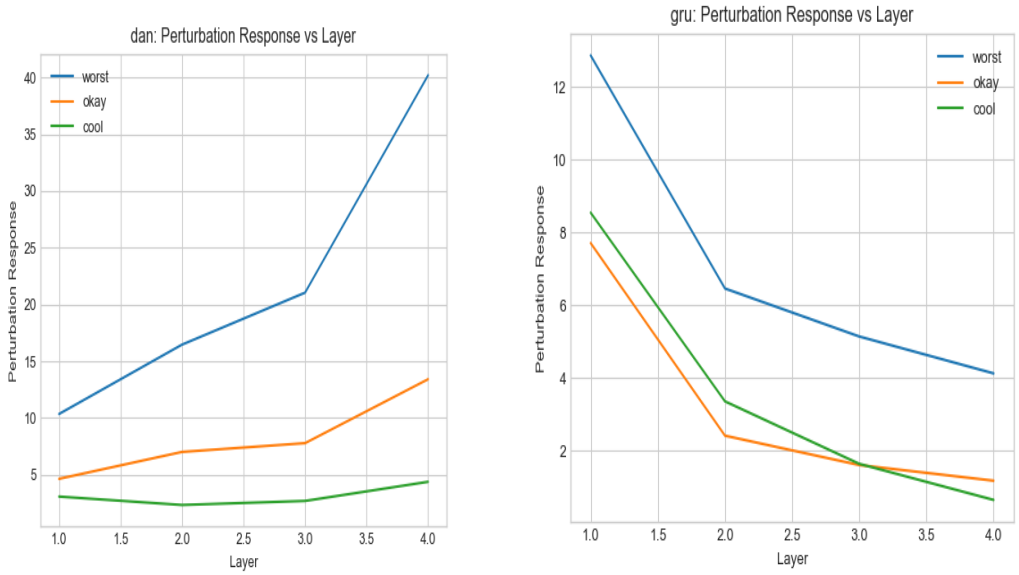
Overall, DAN has better accuracy than GRU. But GRU's accuracy is increasing from one probing layer to another. Whereas, DAN's accuracy steadily increased and then decreased and again started increasing again.

3.2. PROBING SENTENCE REPRESENTATIONS FOR BIGRAM TASK



DAN Accuracy is 0.5
GRU Accuracy is 0.6

3.3. ANALYSING PERTURBATION RESPONSE OF REPRESENTATIONS



Here, we see that in DAN, for all the three words worst, okay and cool in the sentence “the film performances were awesome” the perturbation response increases as the number of layers increases. Hence small changes get magnified as layers get added. Whereas in GRU, the difference between all the 3 sentences almost remained the same till 2nd layer. At layer 3 there is an intersection between the sentence with ‘okay’ and ‘cool’ although ‘worst’ is still dissimilar from them. This can be possible because ‘okay’ and ‘cool’ in that sentence almost mean the same.