NLP Report

1. Describe how you build your model ? How did you do to preprocess your data from dataset ? The distribution of the concern is imbalanced, what did you do to improve the macro F1 score on those concern which are in small scale ?

My Model architecture:

In this homework, I use LSTM for my basic architecture. In order to enhance my network performance and make it more robust, I use Special Dropout after word embedding layer, as well as dropout after LSTM layer and the first FC layer.

Data Preprocess:

1.Data Cleaning:

Since the data from tweets contains a lot of noise, such as URLs, Hashtags, Mentioned Username, Emoticons and Punctuations, I use regular expression and the Python package NLTK which including stopwords and WordNetLemmatizer function to obtain a clean dataset. The following table displays the performance with and without data cleaning.

Model	Glove Coverage Rate	Marco F1 Score
LSTM w/ data cleaning	0.9619	0.623
LSTM w/o data cleaning	0.2820	0.489

2.Word Encoding:

After data cleaning, I use tokenization to divide a string into a list of tokens and convert the string into numeric values.

3.Word Embedding

After word encoding, I use word embedding with pretrain model “Glove” to transform words into the continuous vectors. This approach enables capture semantic similarities between words, making it easier for the LSTM model to understand the text."

Handle Imbalance label:

Since the distribution of the concern is imbalanced, I tried to use the weighted loss to improve the macro F1 score.

At the first, I set the weights is set to 1/counts. However, I observe that the performance is lower, I think the potential reason that the weight of small-scale concern is dominating the loss. Thus, I set the lower bound counts to 300, resulting in improved performance compared to other models

Model	Handle Dominated label	Marco F1 Score
LSTM		0.623
LSTM w/ weighted loss	X	0.601
LSTM w/ weighted loss	O	0.635

2. Have you tried pretrained word embedding ? (e.g. Glove or Word2vec) What was their influence on the result after using them ?

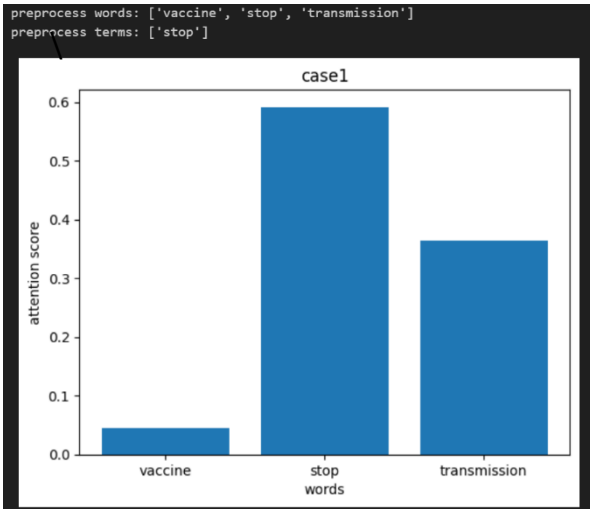
Yes, in my LSTM model, I experimented with three different Glove pretrained models, and those of them all significantly improve the F1 score from vanilla LSTM model, However, those three pretrain model doesn't having significant difference in the F1 score between those models, the possible reason is that those of them have sufficient coverage of the words in dataset.

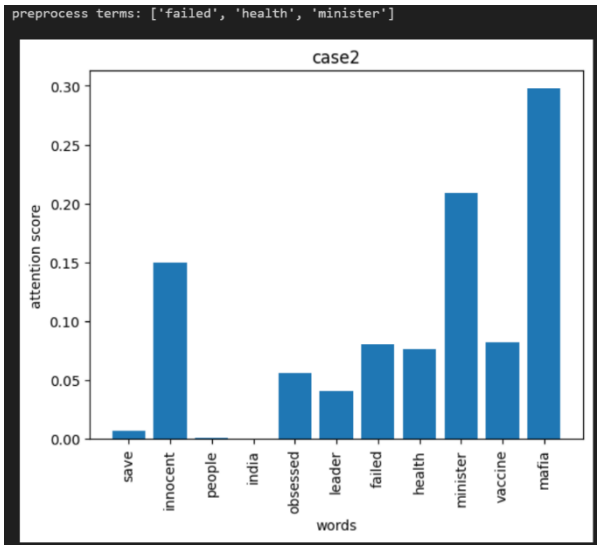
Embedding Model	Coverage Rate	Marco F1 Score
None		0.553
glove.6B.300d.txt	0.9152	0.624
glove.42B.300d.txt	0.9619	0.623
glove.840B.300d.txt	0.9431	0.625

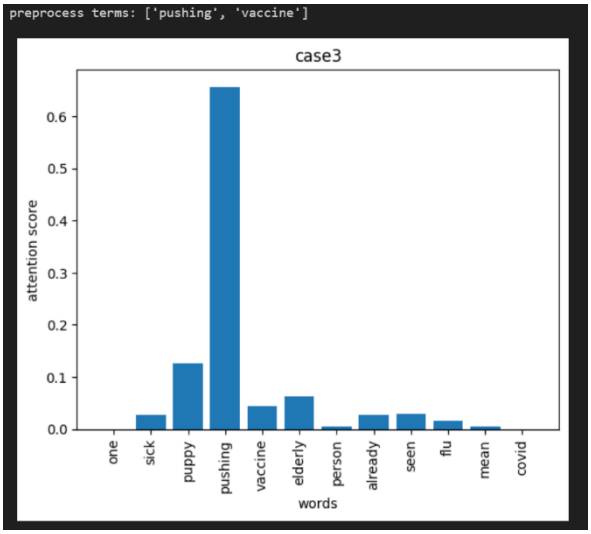
3. Have you tried attention on your model ? What was its influence on the result? When your model predict the concern, what was it focusing on ? Do some case studies.

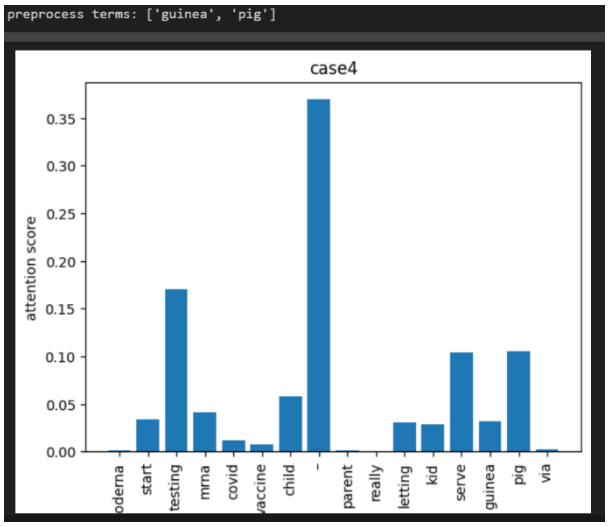
Yes, I have tried the attention on my model, but in my model, it don't have the significant difference from vanilla LSTM model, the following table is the comparison between w/ and w/o attention layer, and the next page is some of case studies.

Model	Marco F1 Score
LSTM w/ attention	0.623
LSTM w/o attention	0.618









In the four cases mentioned above, not all of them have the best attenion scores on the concern term,I believe this may be why the LSTM model with attention layer performance is not better than original LSTM model.