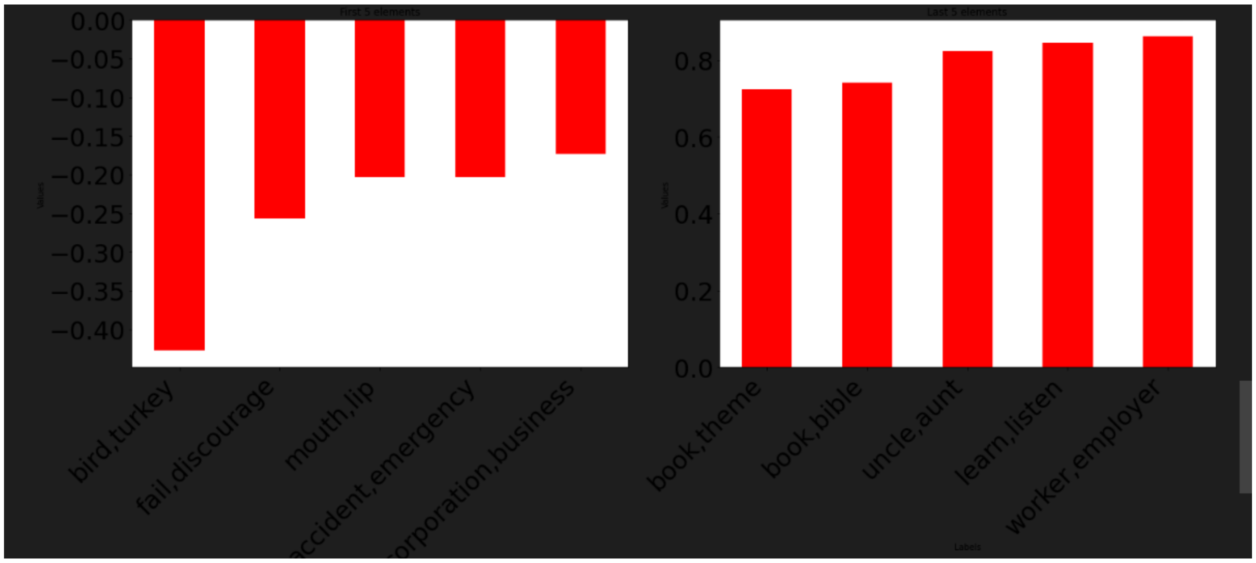
Precog Report

Task1

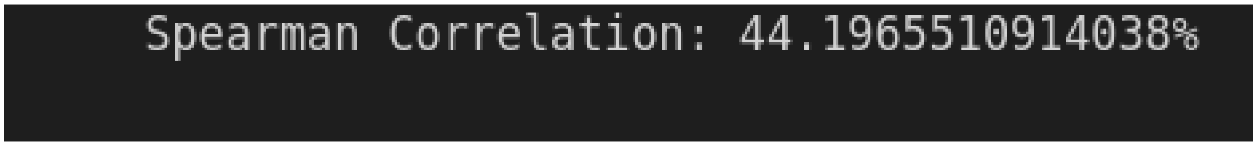
The constrained dataset exhibited a diminished Spearman correlation compared to the unconstrained dataset, attributable to the training being conducted on a restricted quantity of words. As the volume of words utilized for training is augmented, a broader coverage of the vocabulary within our dataset is achieved. This, in turn, leads to an enhancement in Spearman correlation, consequently elevating the overall accuracy of our model.

Various techniques were employed in the constrained segment, yet it was observed that its Spearman correlation lagged behind that of the unconstrained counterpart. The expansion of the word corpus facilitates the generation of additional context-word pairs for Word2Vec, and more words for similarity evaluation in the unique meaning methodology. This augmentation in the dataset's comprehensiveness serves to heighten the model's accuracy.



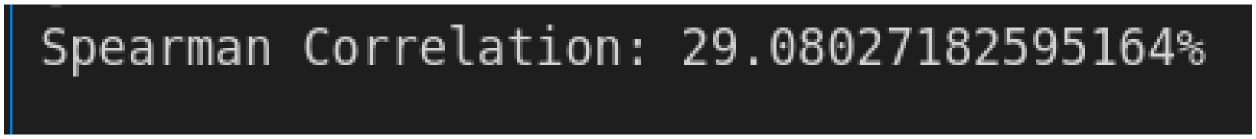
sometimes it gives false values due to less amount of epoch and also can be due to less maximum tokens given to the data set

The spearman correlation for the unconstrained part is the following which is quite higher than the



```
    Spearman Correlation: 44.1965510914038%
```

that of the constrained part (note this a second method):



```
Spearman Correlation: 29.08027182595164%
```

For this task I attempted all the parts which were given

Task 1 a)

In this I used my own CBOW model which I trained on 50000 words of brown dataset. The model used was Sequential. The window size was taken to be 5 .

In other method I used wordnet as a curated knowledge base where I first used it to get distinct meaning words from brown then used that to create vector where each vector element is the similarity score of wordnet with the distinct words then I used cosine similarity and calculated the the spearman correlation of the data with the actual values.

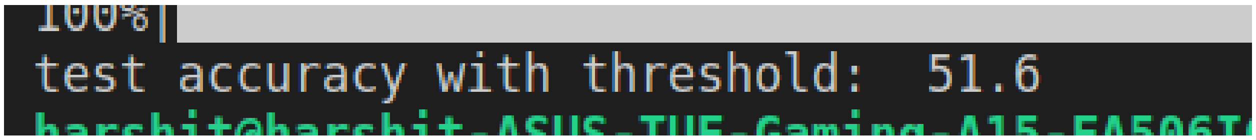
Task 1 b)

In this I used the pretrained google word2vec to get the similarity and applied cosine similarity and calculated the spearman correlation of the data with the actual values.

Task 2 a)

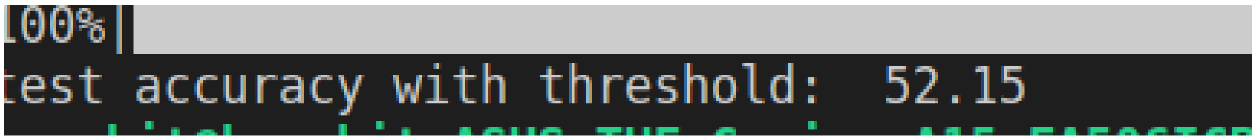
In this, I used word2vec as the model to obtain word embeddings and have done it with two methods. These methods involve obtaining tf-idf weighted vectors and the average of all word2vec vector embeddings of the words.

For tf-idf it is



```
100%|
test accuracy with threshold:  51.6
harshit@harshit-ASUS-TUF-Gaming-A15-FA506I
```

For average it is



```
100%|
test accuracy with threshold:  52.15
```

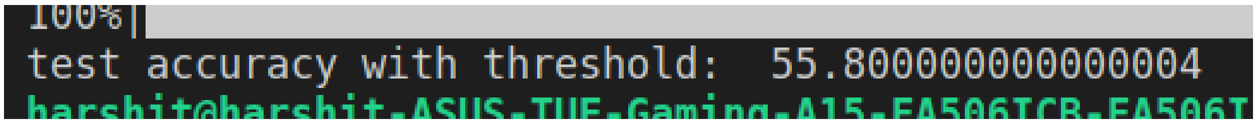
We would think that average is giving the better accuracy so it would be the better method but it is not always the case as the corpus used for tf idf is just the example sentence if it would have been a corpus with multiple instances where the sentence may have been used it would have performed better than the average to be more accurate.

Conversely, if a phrase is common across many documents and lacks discriminative power, the average of Word2Vec might give it equal importance with other phrases. TF-IDF can downweight such common phrases, focusing more on the unique and informative ones.

If a phrase is rare but highly informative within a document collection, TF-IDF can appropriately weigh the phrase and highlight its importance in the weighted Word2Vec representation. The average of Word2Vec may treat all phrases equally, potentially diluting the impact of rare and informative phrases

so there can be cases where one is useful or cases where the other maybe useful for example where TF-IDF may assign non-negligible weights to common stop words if they are infrequent across the entire document collection but average would take the average off all the words regardless

Task 2 b)



```
100%|
test accuracy with threshold:  55.800000000000004
harshit@harshit-ASUS-TUF-Gaming-A15-FA506ICB-FA506I
```

In this stop words are removed from the sentence then the average of word2vec over all words is calculated.

Another method which I used was using the TF-IDF values which as the corpus was not too high as I used on the sentence itself is not a good way it would have been only if a huge corpus would have been given or may have used all the sentences have not tried that but could be used .

If a sentence contains rare words with high TF-IDF values, these words will dominate the TF-IDF weighted Word2Vec representation. However, these rare words may not have well-learned embeddings in the Word2Vec model due to limited occurrences in the training data. On the other hand, the average of Word2Vec considers all words equally, smoothing out the impact of rare words.

If a sentence contains common words with low discriminative power, such as frequent stop words or generic terms, the average of Word2Vec may give these words equal importance. In contrast, TF-IDF would downweight such common terms, potentially providing a more focused representation by emphasizing less frequent but more informative words.

Bonus Task

1) Transformers

In this method sentence-transformer was used for calculating the embedding of a sentence .

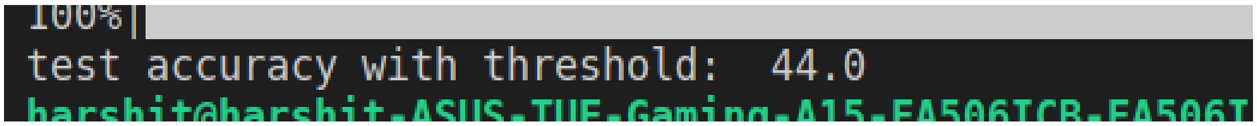
Xgb classifier was used as model for training and testing the data . This gave high accuracy

because XGB and sentence transformer are more effecient .

Accuracy - 61.3%

2) LLM

I used Gemini From Google and Use LLama for this



```
100%|
test accuracy with threshold:  44.0
harshit@harshit-ASUS-TUF-Gaming-A15-FA506ICB-FA506I
```

So we can say that the Transformers perform the best on the given data rather than LLM and the technique which I used.