Paraphrase detection model and Label noise analysis

Prakhar Pandey 200101081
Pranjal Baranwal 200101083
Pratham Pekamwar 200101087
Dhruv Shah 200101124

Abstract

In this project, we analyze the impact of label noise on the performance of the paraphrase detection model. i.e. We will analyse the sensitivity to label noise (robustness) in our paraphrase detection model.

1 Introduction

One of the numerous real-world uses for natural language processing is paraphrasing. Paraphrasing is the expression of meaning using different words. We have created a paraphrase detection model. A paraphrase detection model is helpful in real-life scenarios as the use and integration of paraphrasing (algorithms in general) in our lives is increasing a lot and it is helpful to detect paraphrasing.

General Concepts

- Training
  Training in our model is used for finding threshold.

$$\begin{aligned} \frac{\partial}{\partial w_1} g(\mathbf{v}) &= 0 \\ \frac{\partial}{\partial w_2} g(\mathbf{v}) &= 0 \\ &\vdots \\ \frac{\partial}{\partial w_N} g(\mathbf{v}) &= 0 \end{aligned} \tag{1}$$

  Transfer learning occurs when we use the knowledge that was gained from solving one problem and apply it to a new but related problem. Specifically, this is a process that takes a model that has already been programmed for paraphrasing and then tunes or tweaks the model to make it perform a tasks that are similar to the Initial Dataset. Assuming the original dataset is similar to the new task(test dataset) for paraphrase detection.
  This model can be trained to the input dataset which will set the threshold value for the testing dataset or actual use. This is done because paraphrasing depends on the context which can be captured by the training dataset. Hence, the training dataset should be similar to the type of data on which the model is expected to do the paraphrase detection.

- Label noise
  The term "Noise" in our analysis refers to the noise in the training dataset of the models. For our analysis, more noise means more sentence pairs with an incorrect prediction of paraphrase. This type of noise which impacts on the labels of training dataset is known as *label noise*.

- Input to paraphrase detection model
  There are two inputs to the model.
  1. Training dataset.
  It is used to fine-tune the model, which is important because it helps in determining the context by giving paraphrases in similar contexts. Our paraphrase detection model will take input as sentence pairs that are 1 or 0 according to if they are or are not the paraphrases of each other respectively.
  2. Test dataset (or two sentences if needed).
  The test dataset is similar to training dataset as we would need to know if the sentences are paraphrases of each other. But the model is actually designed to take two sentences for paraphrase detection.

- Analysis and Output of the experiment
  The output of the model can be 1 or 0 to denote paraphrase or not but the output of the experiment would be the accuracy of the model on test dataset for various settings of label noise. Label noise is induced externally and only this noise is considered for the analysis. We have two settings for the label noise.

1. Random noise.
2. Realistic noise.
Higher noise is likely to reduce the performance of the models. The analysis shows that the random noise won't affect the model unless the noise is too much more than the actual data. But the realistic noise does affect the optimal threshold and in turn the accuracy of the model.

2 Methods

Paraphrase detection model
The model detects if two sentences are paraphrases of each other in following steps.

1. It creates vectors from sentences using TF-IDF.
2. It calculates the cosine similarity of the two vectors.
3. It compares the cosine similarity with a threshold determined by the training dataset.

TF–IDF, short for term frequency-inverse document frequency, reflects how important a word is to a document in a collection or corpus. It is often used as a weighting factor. We use this on every word and get the importance of that word to the sentence(document) concerning the corpus(just the two sentences). This works because we don't want to give more weight to the common words in both sentences and more weight to other words. Then the vector for each sentence is the

$$tf - idf = tf * idf$$

We calculate tf as
tf(t,d) = count of word t in document d / total number of words in document d

But scikit-learn uses

$$tf(t) = No.\,of\,times\,term\,t\,occurs\,in\,document$$

In scikit-learn, default i.e. smooth_idf = True

$$idf = log((1 + samples)/(1 + documents)) + 1$$

We calculate the cosine similarity of the vectors of the two sentences, which are to be detected for paraphrasing. We fine-tune the model on the training dataset to find the optimal value of the threshold.

$$cos\Theta = (\mathbf{A}.\mathbf{B})/\left|\mathbf{A}\right|\left|\mathbf{B}\right|$$

The model classifies the two sentences as paraphrases of each other if the cosine similarity of the two sentences is above the optimal threshold, or else it classifies them as not paraphrases.

Dataset

The dataset used is Microsoft Research Paraphrase Corpus(MSRPC)(in the references section). It is extracted from news sources on the web, along with human annotations.

Format of the dataset
Quality ID1 ID2 String1 String2
Quality is 1(paraphrase) or 0(not paraphrase)
IDs are irrelevant to the experiment.
Strings are the two sentences.

The dataset is already divided into two parts.
1. Training dataset - 4,076 sentence pairs (2,753 positive: 67.52. Test dataset - 1,725 sentence pairs (1,147 positive: 66.5

3 Experiment

Control flow of experiment

1.Training
Import training dataset -> tokenization -> remove stop words -> stemming -> create vocabulary
make other words as unknown

2.Noise
Our objective in designing this experiment was to find out the impact of label noise in the training dataset of our model. Our paraphrase detection model was trained using training set with three types of noise

1. Random noise
(multiple levels (ranging from 0 to 100) and was tested on an independent test set)
2. Realistic noise (case-1)
(The noise added decreases linearly with respect to cosine similarity from 90% to 0%. The noise is added only to the labels which are 1(i.e. those that represent similar sentences))

3. Realistic noise (case-2)
(we also add noise to the 0 labels. On top of

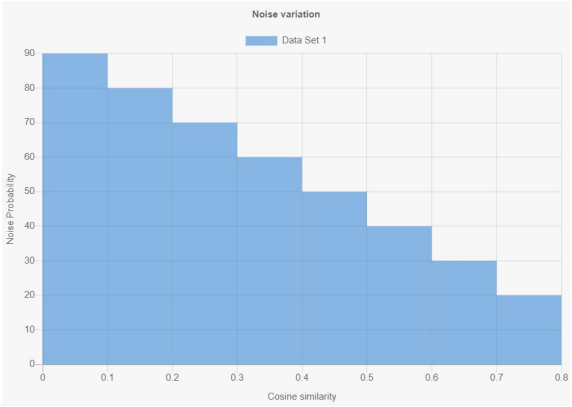

Figure 1: Graph for noise percentage

changes made in case 1 we also linearly increase the noise added in 0 labels with respect to cosine similarity from 0% to 90%.)
Here we refer to realistic noise as the type wherein the amount of noise added is dependent on cosine similarity as 0.7 similarity is more likely to be wrongly labelled as paraphrase than 0.9 similarity.

3.Optimal Threshold

For each noise setting
Looped over threshold values from 0.50 to 1.00 with 0.001 increase in value for each loop. This was done because it's not realistic to use threshold values below 0.5 for TF-IDF model to use. It just helps the model to not get impacted too much by noise.
Accuracy is calculated for all cases
Optimal threshold(optimal for training set) is the one for which accuracy is maximum

4.Testing preprocessing
Import test dataset -> tokenization -> remove stop words -> stemming -> adding unk by comparing with vocabulary

5.Testing
The dataset is checked for paraphrase according to optimal threshold value. With each combination, the performance of the model is generated. We used accuracy and F1-score as the performance metric for this study. The calculated values of accuracy and F1-score are reported in the results section.

Accuracy is the percentage of pairs of sentences in the dataset for which the paraphrase detection model gave correct output.

F1-score is a combination of two metrics- precision and recall. It is the harmonic mean of these two metrics. It takes both of these into account. The F1-Score value ranges between 0 and 1, the higher the F1-score, the better the model.

$$F1 - Score = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

Precision: Of all positive predictions, how many are really positive?
Recall: Of all real positive cases, how many are predicted positive?

$$Precision = \frac{True\ Positive}{True\ positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ positive + False\ Negative}$$

Graphs are created for better analysis.

6. Analysis

The data was analyzed to get the impact of both types of noise on the accuracy of the model

4 Results

Graph minima = Cosine similarity at which [cumulative of P(paraphrase) - cumulative of NP(not paraphrase)] is minimum

Output table 2
Graphs

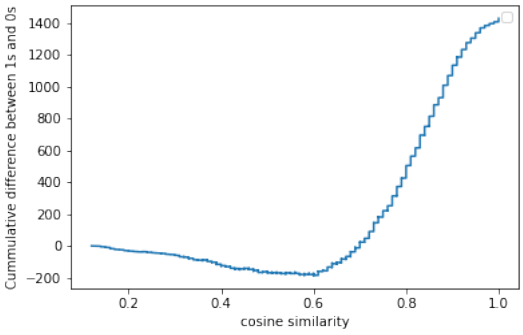

Figure 2: Graph for no noise

Noise percent	Optimum Threshold	graph minima	Accuracy	F1-Score
0	0.601	0.60	0.710	0.788
5	0.531	0.53	0.725	0.814
10	0.601	0.60	0.710	0.788
15	0.571	0.57	0.718	0.801
20	0.531	0.53	0.725	0.814
25	0.571	0.57	0.718	0.801
30	0.571	0.57	0.718	0.801
35	0.571	0.6	0.718	0.801
40	0.501	0.56	0.721	0.816
45	0.511	0.53	0.719	0.814
50	0.601	0.60	0.710	0.788
55	0.501	0.5	0.721	0.816
60	0.501	0.5	0.721	0.816
65	0.601	0.60	0.710	0.788
70	0.521	0.52	0.721	0.814
75	0.851	0.85	0.485	0.388
80	0.811	0.88	0.540	0.501
85	0.991	1	0.341	0.017
90	0.991	0.99	0.341	0.017
95	0.991	1	0.341	0.017
Realistic-1	0.691	0.69	0.674	0.728
Realistic-2	0.601	0.60	0.710	0.788

Table 1: Table to show the effect of noise in the Microsoft dataset.

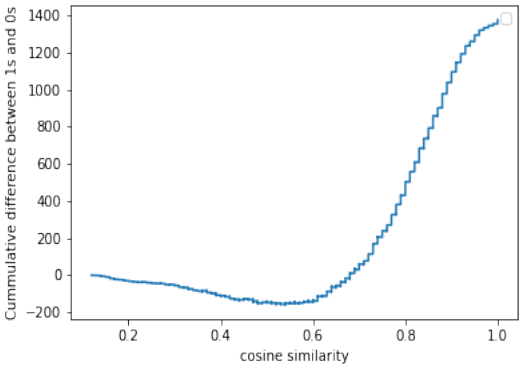

Figure 3: Graph for five percent noise

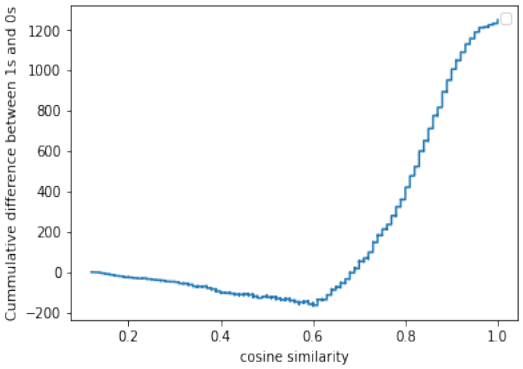

Figure 4: Graph for ten percent noise

5 Analysis

1. Analysis for random noise
Random noise decreases P(paraphrases) ev-

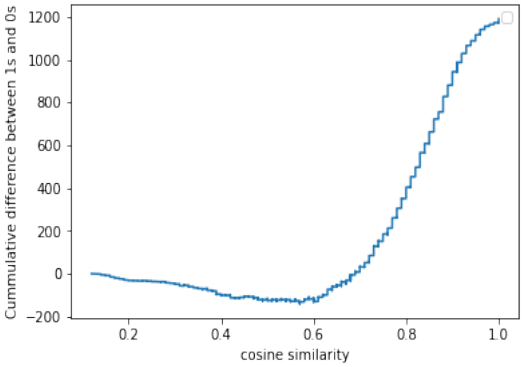

Figure 5: Graph for fifteen percent noise

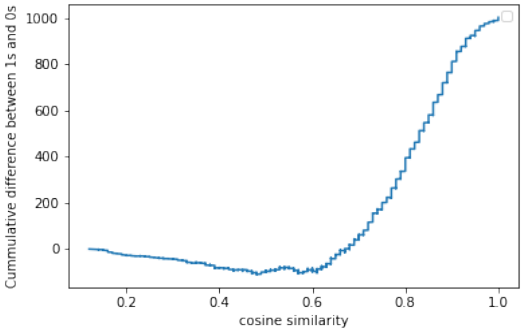

Figure 6: Graph for twenty five percent noise

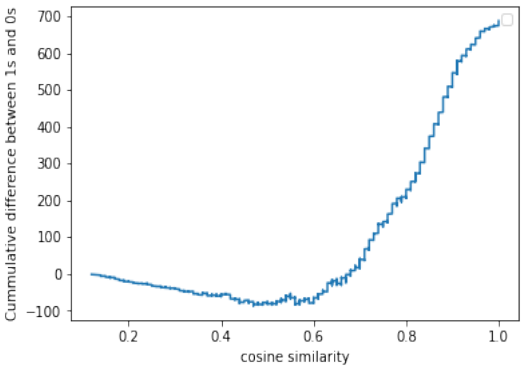

Figure 7: Graph for forty percent noise

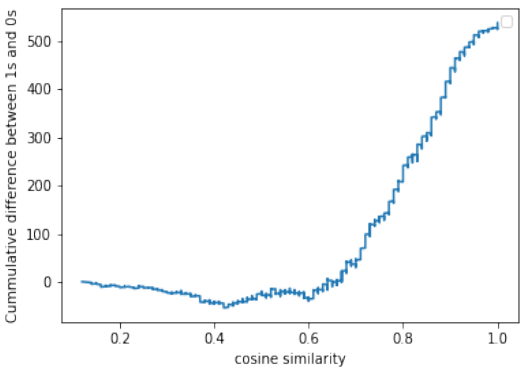

Figure 8: Graph for fifty percent noise

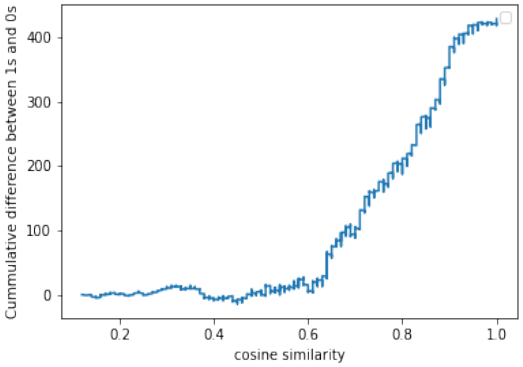

Figure 9: Graph for sixty percent noise

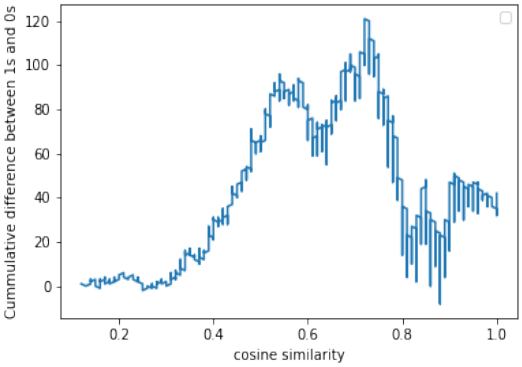

Figure 11: Graph for eighty percent noise

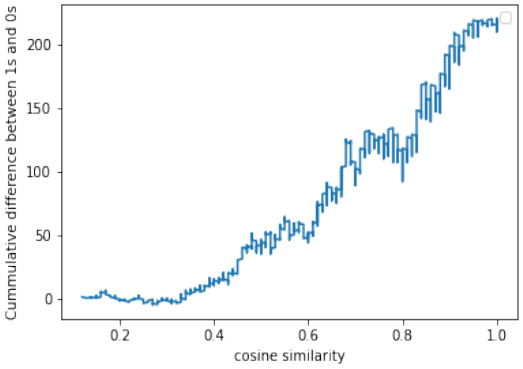

Figure 10: Graph for seventy percent noise

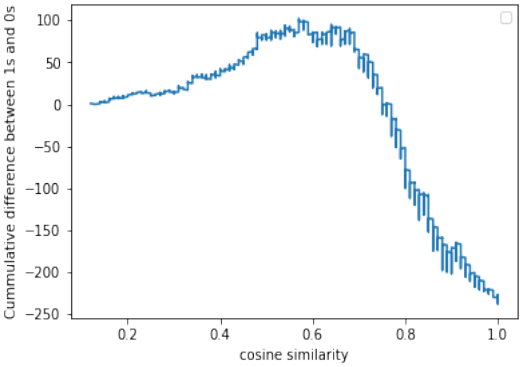

Figure 12: Graph for ninety percent noise

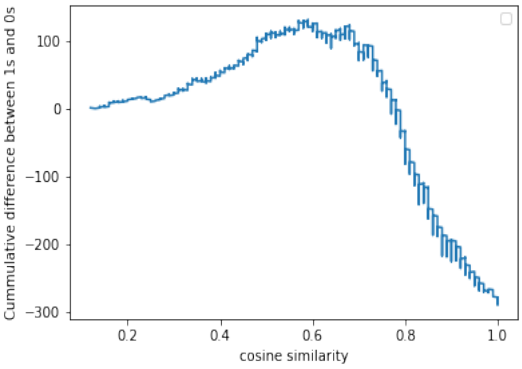

Figure 13: Graph for ninety five percent noise

erywhere and increases NP(non paraphrases) everywhere randomly. The optimal value changes(increases) when P(paraphrases) are less than NP(non paraphrase) on the values above the optimal value. We observe that the noise distribution does not affect the model's performance until the noise level is very high, on which the model breaks. This is mostly because the noise is distributed randomly which distributes quite uniformly so that it doesn't affect the threshold because the P(paraphrases) having cosine similarity above the optimal threshold are much more than NP(non-paraphrases) so that the only high noise would make the quantity of NP more than P on cosine similarity higher than the optimal threshold which is when optimal threshold increases. Also there are a few more things to observe in this implementation. The first is that there is logically a lower bound on the threshold. In this implementation, the threshold is not allowed to be lower than 0.5. This is because if the model classifies sentences that have a cosine similarity score lesser than 0.5 as paraphrases of each other then the model is not usable in practical situations. The graphs of cumulative difference between 1s and 0s with respect to the cosine similarities show that the optimum threshold value corresponds to some minima of the graph. The reason is that if the threshold lies on the right side of this minima, then on shifting of threshold to left, the model will have more correct predictions, hence that threshold wouldn't be the optimal threshold. This is because for each one encountered on cosine values less than optimal threshold, the model would have made a wrong prediction, and for each zero encountered on cosine values less than optimal threshold, the model would have made a correct prediction. Thus, mini-

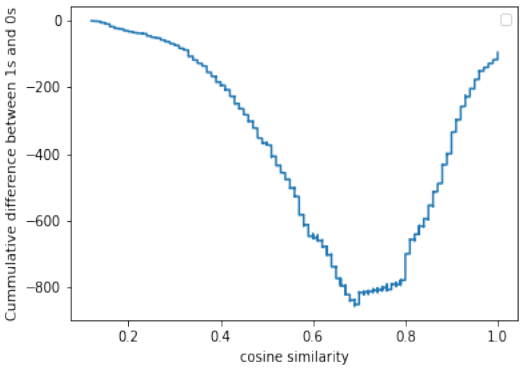

Figure 14: Graph for realistic case 1 noise

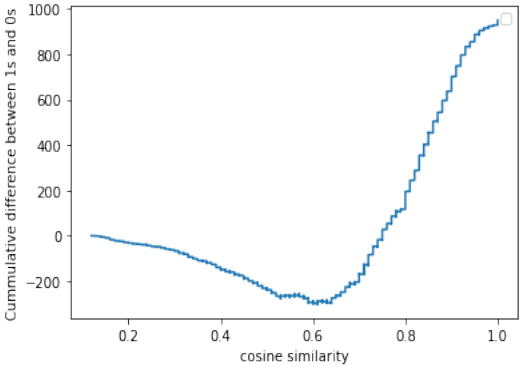

Figure 15: Graph for realistic case 2 noise

mizing the difference between these two leads to the most amount of correct predictions. Further, on increasing the noise level to a very high level (say 85%) the optimum threshold now becomes close to 1. The reason is because the nature of noise induced here is label 1s being converted to label 0s. For a sufficiently high noise level, the sample contains so many 0 labels that the max accuracy of predictions is obtained if all the sentences are labeled not paraphrase or 0 by the model. //

2. Analysis for Realistic Case 1 The noise is deemed realistic because in real world situations, the sentences that are clearly very similar to each other would have lower chances of being labeled wrong. Thus, for sentence pairs having cosine similarity near 1, there is a low chance of noise error. The noise level would then subsequently increase as the cosine similarity declines. This trend would break when the sentences become quite dissimilar. We have used a linear approximation to depict this trend. Since the statement having very low cosine similarities are not present in the corpus, thus, that case needs no special measures in our noise implementation. Therefore, the noise is implemented in slab wise fashion. The slab of 0-5% has 90% of its label 1 flipped into 0. The next slab has 80% noise and so on. Figure 18 shows that the intersection gives the optimum threshold. The optimal threshold can neither lie on the right of this intersection nor on the left. This can be seen by the table 2. Consider the range of 0.65-0.70. On a right shift of the optimum threshold from 0.65 to 0.70, the increment in the correct predictions would be 179, since these 'not paraphrases' are now correct predictions. However, the 220 'paraphrases' are now incorrect predictions since they lie below the threshold now. Thus the overall accuracy of the model will decline. Note that the threshold increases in this case as compared to the no noise case. This is due to the flipping of 1 labels to 0 near the optimum threshold region. There is not a major shift since as threshold increases, the chances of further noise in this range are lower.

3. Analysis for Realistic Case 2 In this case, the increase in label noise in label 1 and label 0 balance each other out (for example: for high cosine similarity, there are more P(paraphrases) but the noise percent is less, which flips it to NP(non-paraphrase). Also, there are fewer NP with high cosine similarities. Still, the noise percent is more which flips it to P), thus resulting in minimal change in optimal threshold and thus keeping the performance or accuracy unaffected.

Cosine Similarity Range	Paraphrases	Not Paraphrases
0-0.05	0	0
0.05-0.1	0	0
0.1-0.15	0	10
0.15-0.2	1	21
0.2-0.25	0	16
0.25-0.3	2	22
0.3-0.35	5	56
0.35-0.4	7	53
0.3-0.35	5	56
0.4-0.45	25	80
0.5-0.55	66	140
0.55-0.6	83	222
0.6-0.65	154	175
0.65-0.7	220	179
0.7-0.75	186	144
0.75-0.8	241	122
0.8-0.85	347	122
0.85-0.9	286	61
0.9-0.95	229	27
0.95-1	91	6
1	22	0

Table 2: Table to show the effect of noise in the Microsoft dataset.

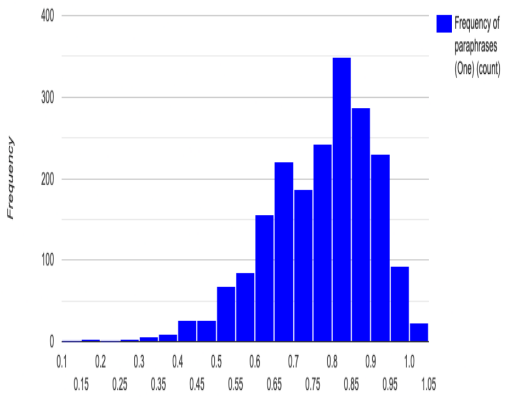

Figure 16: Analysis for realistic case 1-paraphrase

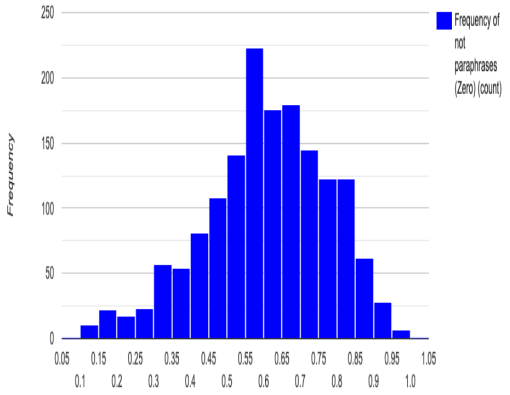

Figure 17: Analysis for realistic case 1-not paraphrase

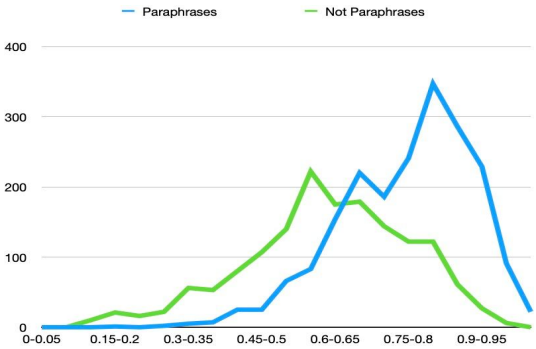

Figure 18: Graph for realistic noise

6 Conclusion

For Random Distribution

We observe that the noise distribution does not affect the model's performance until the noise level is very high, on which the model breaks. This is mostly because the noise is distributed randomly, which distributes quite uniformly so that it doesn't affect the threshold because the P(paraphrases) having cosine similarity above the optimal threshold are much more than NP(non-paraphrases) so that the only high noise would make the quantity of NP more than P on cosine similarity higher than the optimal threshold which is when optimal threshold increases.

For Realistic Distribution (case1)

The performance of the model declines definitively in this case. This is because the optimal threshold increases because the P(paraphrase) cases whose cosine similarity is more than the optimal threshold decrease, which allows the optimal threshold value to increase since it will enable it to improve the accuracy by correctly guessing some NP(non-paraphrase) cases whose cosine similarity values are now less than the new optimal threshold.

For Realistic Distribution (case2)

In this case, the increase in label noise in label 1 and label 0 balance each other out (for example: for high cosine similarity, there are more P(paraphrases) but the noise percent is less, which flips it to NP(non-paraphrase). Also, there are fewer NP with high cosine similarities. Still, the noise percent is more which flips it to P), thus resulting in minimal change in optimal threshold and thus keeping the performance or accuracy unaffected.

Advantage of using TF-IDF
Robustness against noise

The TF-IDF model is quite robust when measured for noise sensitivity. However, this comes in tradeoff to the accuracy, which is lower than neural network models which are great in accuracy in NLP tasks. This is because the dependence of our model on the training set is lower than the neural models which get all their weights and biases from the training dataset and overfit to the noise.

7 Google colab link

Link

8 References

1. Microsoft Research Paraphrase Corpus
MicrosoftOfficialLink