Paraphrase Identification and Generation - Team 4

Github Link - https://github.com/manasvi-26/Paraphrase-Identification-and-Generation

Introduction

The report discusses the work done by Team 4 as part of the Information Retrieval and Extraction course, Monsoon 2021. We discuss the importance of the paraphrasing task in the area of research in Natural Language Processing (NLP). We also discuss popular work that has been done in the area of paraphrasing. The report also has a detailed account of the experimentation that was done including the results obtained with pre-trained models used off the shelf as well as the model fine-tuned by us for the task of paraphrase identification and generation.

Much focus is given to Google Research's Paraphrase Adversaries from Word Scrambling (PAWS) dataset through the task. The final model that we used for the paraphrase identification task was the BERT transformer fine-tuned on the PAWS dataset.

We use the Text-to-text Transfer Transformer (T5) model to get the results for the task of paraphrase generation. The newly recognized role of transfer learning in bettering the performance of deep learning models on Natural Language Processing (NLP) tasks is the reason why we decided to work on the same.

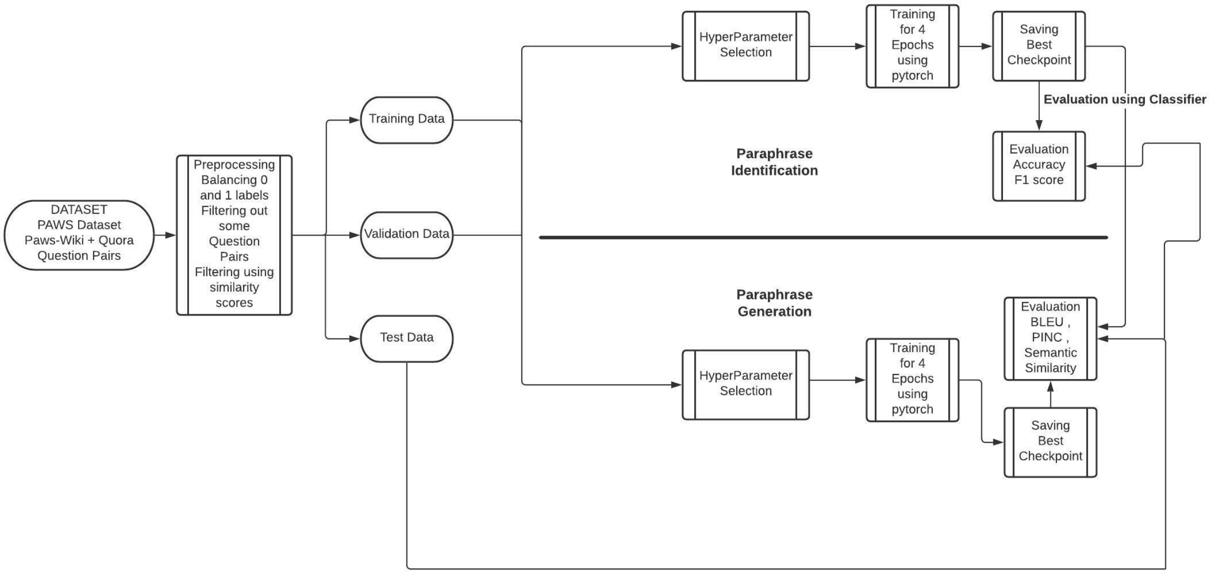
Problem Statement

Our task for the major project was to build a system that would detect and generate paraphrases of sentences. Paraphrase generation is one of the most important and challenging NLP tasks. Paraphrasing is nothing but generating sentences that are similar in meaning but are different in terms of sentence structure. The paraphrasing task as a whole can be divided into two major tasks:

1. Paraphrase Detection/Identification
2. Paraphrase Generation

The Paraphrase identification task is treated as a supervised machine learning task of binary classification. We trained a model on the PAWS dataset [1] that contains pairs of sentences and a label identifying them as paraphrases or not.

The paraphrase identification task is different from another popular NLP task, Semantic Textual Similarity (STS), in that the former is concerned with identifying whether two texts have a similar meaning, whereas the latter is concerned with the degree of that similarity.



Workflow Diagram

Meanwhile, the Paraphrase generation problem is a sequence generation problem where a given input would result in the model giving out *n* sentences that are paraphrases of the given input sentence.

Importance of this problem

Paraphrasing has a wide range of applications across various fields. It frequently finds use in the fields of information retrieval, question answering and text summarization.

One of the best examples of applications of the paraphrase identification task is Plagiarism detection. Paraphrasing can also be useful in evaluating machine translation tasks.

The task of generating paraphrases is useful for creating new samples that can help expand existing corpora, and as a result, increase the amount of data available.

Related Work

As mentioned earlier, the Paraphrasing task is divided into two major tasks, Paraphrase identification and Paraphrase generation. The traditional baseline systems treated the two tasks independently. Palivela et al [2] talk about the integration of the separate tasks of Paraphrase identification and Paraphrase Generation to get a single, unified model. They used a T5 model [3] to exploit the benefits of transfer learning and fine-tuned it keeping the paraphrase generation task in mind. The model was also capable of performing the paraphrase identification task. The model was then evaluated against popular metrics of BLEU [4], ROUGE [5], METEOR [6], WER [7], and GLEU [8].

One of the major takeaways from this work is the effective use of transfer learning in solving both the subtasks of the paraphrasing problem. Raffel et al. explore the use of transfer learning in Natural Language Processing tasks and discusses why it is a good idea to make use of the same in the context of NLP. The overview is to train a model in a data-rich task and then fine-tune it for downstream tasks. NLP tasks require the model to develop general-purpose knowledge. The reason is that besides identifying patterns, NLP tasks require the model to "understand" text. When a model develops this general knowledge, we are enabling the model to understand the text better. The Text-To-Text Transfer Transformer (T5) is introduced that is pre-trained on a large unlabelled dataset as required in transfer learning problems in order to enable the model to develop general-purpose knowledge. Identifying and training this model was the next challenge. Wikipedia dataset is of immense quality. But it lacked the volume to pre-train a transfer learning model. The Common Crawl project, on the other hand, is enormous in size. But it lacks quality. So to balance out and satisfy both the requirements, the Colossal Clean Crawled Corpus (C4) was introduced. This is over two orders of magnitude larger than the Wikipedia dataset and is also cleaned. The cleaning included deduplication, discarding incomplete sentences, removing noise and removing offensive content.

We also look at several papers that deal with different paraphrase generation techniques. Hegde and Patil [9] discuss an unsupervised approach to paraphrase generation that involves using pretrained language models (particularly GPT-2 [10]). The solution they discuss involves corrupting source sentences and then creating paraphrases by reconstructing the corrupted source sentences using GPT-2.

Datasets

Paraphrase Adversaries from Word Scrambling **(PAWS)** is a dataset containing human-labeled pairs of paraphrases. It features the importance of modeling structure, context, and word order information for the problem of paraphrase identification and generation.

Since the original dataset has two subsets, one based on Wikipedia and the other one based on the Quora Question Pairs (QQP) dataset, we reconstruct the original using these two subsets. Since the QQP dataset is based on question forums that Quora marked as a duplicate, it did not particularly focus on purely paraphrased sentences. However, we included this to add a very slight noise to our dataset and increase the generality of our model. (questions and statements)

The columns of the dataset are - **sentence1, sentence2, and label(paraphrases or not)**

In our current dataset, the sentence pairs are only in the English language.

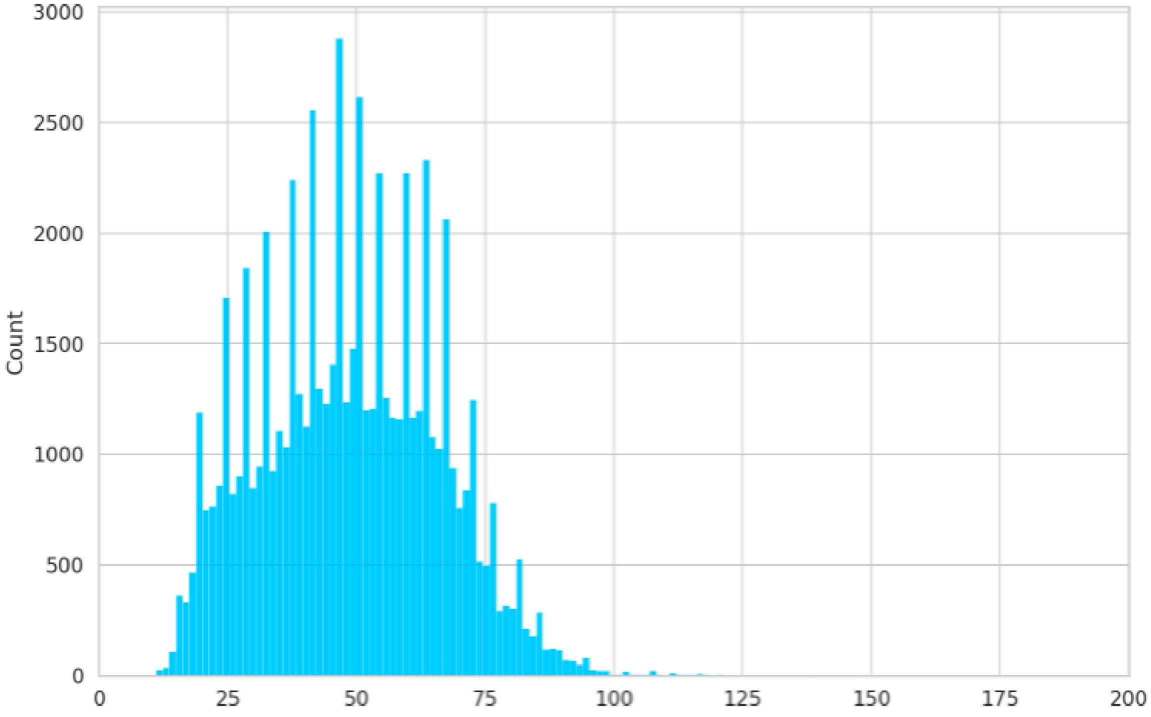
We remove the sentence pairs which are labeled as 1 and have very little semantic similarity (less than 0.2) by using Sentence-BERT [11]. This prevents the model from generating vastly different sentences in terms of semantic similarity.

To avoid class imbalance and the problems that arise with it, we made sure to maintain a balanced dataset in terms of the ratio of 1-labeled sentences and 0-labelled sentences.

For the Generation Model, only sentence pairs labeled as 1 were given as input.

PAWS dataset	**Train**	**Validation**	**Test**
Total Number of Rows	89,401	15,000	15,000
Number of Rows Labeled as 1	41,829	7,039	7,036
Number of Rows Labeled as 0	47,572	7,961	7,964

To identify the max padding length for the tokenization in our models, we plotted the counts of lengths of sentence pairs in the dataset. Sentence1 and Sentence2 are fed into the tokenizer without any padding and then these sequence lengths are used. The plot below shows the distribution of sequence lengths. Looking at the plot below a max pad length of **120** was decided for identification.



Y-axis - Count , X-axis - Sequence Lengths in the dataset

Identification

Baselines tried

We tried several techniques for Paraphrase Identification, listed below:

Attempt 1:

The first attempt was to use a pre-trained T5 model from the Huggingface library, encode the sentences in the sentence pair and check for the cosine similarity. This gave very poor results and the accuracy was as low as 57%. This model was trained on the English sentence pairs of the PAWS-X dataset [12].

Attempt 2:

In this attempt, we tried to use a pre-trained Sentence Transformer from the Huggingface library, encode the sentences in the sentence pair and check for the cosine similarity. Since this was equivalent to a sentence-similarity task, the mean cosine similarity values for pairs of sentences that were paraphrases and those that weren't were almost similar, as a result of which we didn't get good accuracy. This model was trained on the English sentence pairs of the PAWS-X dataset. We tried three different pre-trained Sentence Transformer models and obtained the following accuracies.

- 57.49% (bert-base-nli-mean-tokens)
- 55.7% (paraphrase-albert-small-v2)
- 56.59 % (paraphrase-multilingual-mpnet-base-v2)

Previous attempts (Phase 1):

After the above attempts, we decided to incorporate **Fine-Tuning** for our models. This was done using a framework called **Fast AI,** which basically facilitates Transfer Learning on pre-trained models. **Transfer learning** is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. It basically means fine-tuning and retraining an already pre-trained generic model for specific tasks to obtain much better results. The models and datasets used here were:

- PAWS-X (English) and MRPC with BERT-base-uncased - (93.1% accuracy with PAWS-X and 82.84% accuracy with MRPC)
- PAWS-X (English) with XLNET [13] - (90% accuracy)

(Note: This was on PAWS-X, which is a smaller machine labelled subset of PAWS)

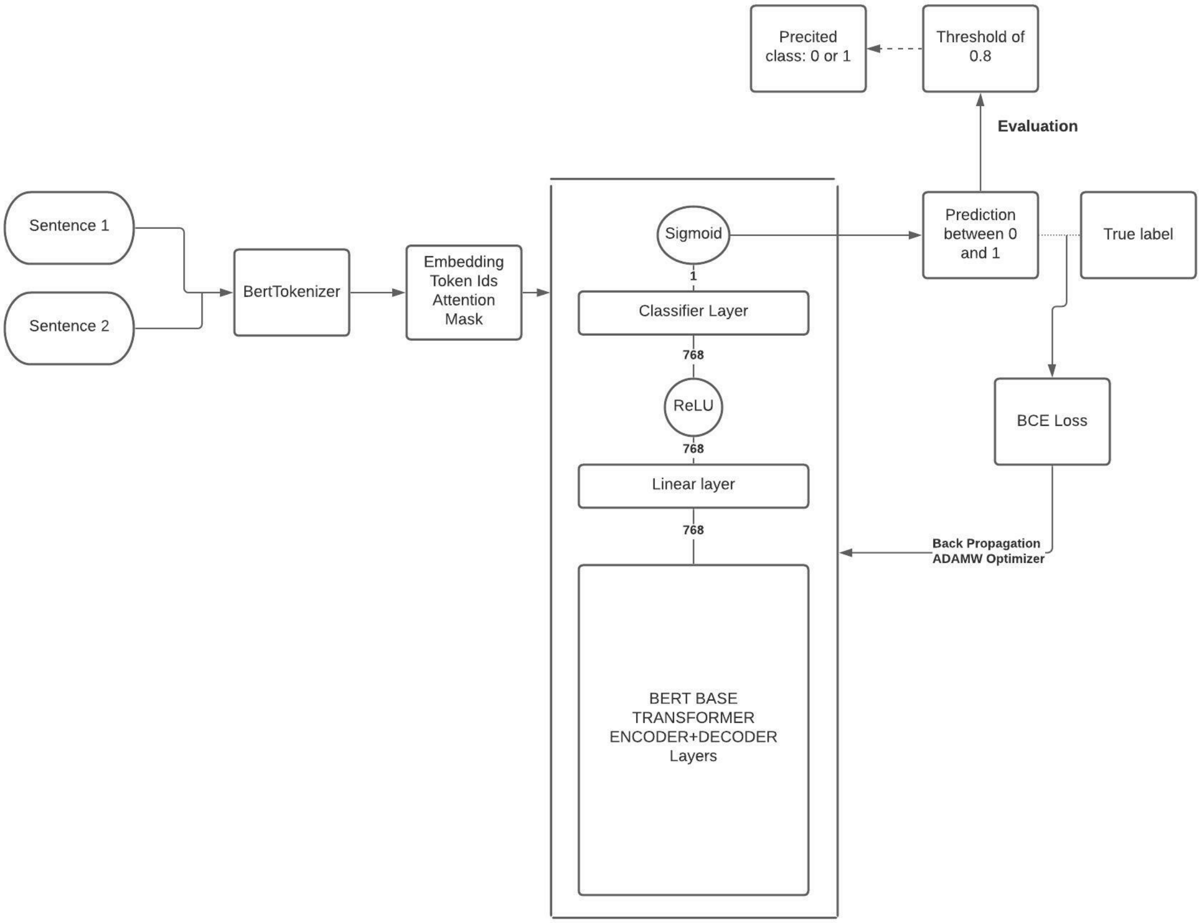
Final model

The BERT transformer [14] was fine-tuned for the paraphrase identification task. Note that we used the PAWS dataset here (and not the PAWS-X English dataset). The objective of this task is to label a pair of sentences as paraphrases/non-paraphrases of each other. Two extra layers were added on top of the BERT model along with a Rectified Linear Unit layer in between them. The last layer converted the output to a single dimension which was then put in a sigmoid function to give a probability between 0 and 1. The loss criterion used was Binary Cross-Entropy loss.

The Pre-Trained model

The BERT - Bidirectional Encoder Representations from Transformers by Google was used as the base pre-trained model for this classification task. Specifically, the pretrained 'bert-base-cased' model from the Bert Base architecture was used. It is a language model which is bi-directionally pre-trained on unlabeled text using masked language modelling (MLM) and so can have a deeper sense of language context and flow than single-direction language models. As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. This characteristic allows the model to learn the context of a word based on all of its surroundings. Due to this nature of pertaining performed on BERT, it can be fine-tuned easily by adding just a single layer to convert it into a classifier and give state-of-the-art results.

Architecture



Tokenization

The tokenizer was taken from the BertTokenizer module from the transformers library.

BERT requires the following tokenization:

[CLS] A [SEP] B [SEP]

[CLS] - Start of sequence

A, B - Sentence and Paraphrase respectively

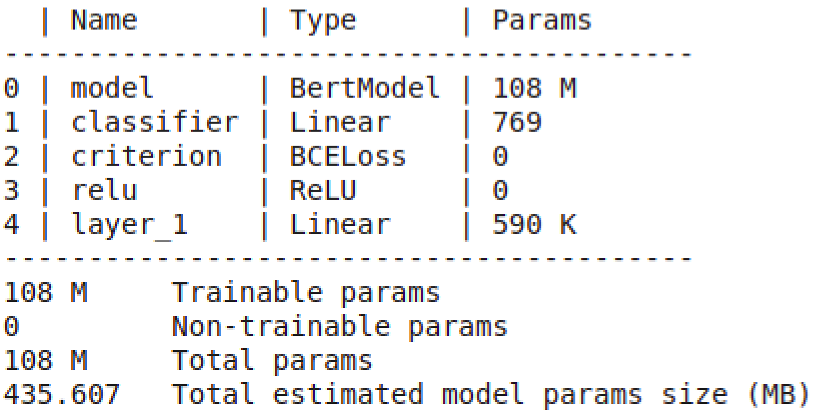
[SEP] - Separator token

For the model to understand and digest the inputs easily, each such sequence (sentence-paraphrase pair after the above tokenization) is then padded to a fixed length at the right using [PAD] tokens. On analysis of the database, we observed that each such pair reached no more than a length of 100 -120 tokens and so we chose 120 as the max pad length. This padded sequence is finally converted to a tensor of token ids as embeddings and fed to the forward layer of the model.

The BERT Base model uses 12 layers of transformers block with a hidden size of 768 and number of self-attention heads as 12 and has around 110M trainable parameters. On top of this, we added the following:

- A Linear layer - Input 768, Output 768
- ReLU layer - Rectified Linear Unit
- Classifier Layer - Input 768 , Output 1
- A sigmoid function on the last node

After the forward layers of the BERT model, the pooled output (with the shape (batch size, hidden layer size)) was fed to the first Linear Layer. Then the ReLu activation function was applied to this layer. This output was then converted into a one-dimensional number through the last Classifier Layer which was finally put in a Sigmoid activation function. $\frac{1}{1+e^{-x}}$.



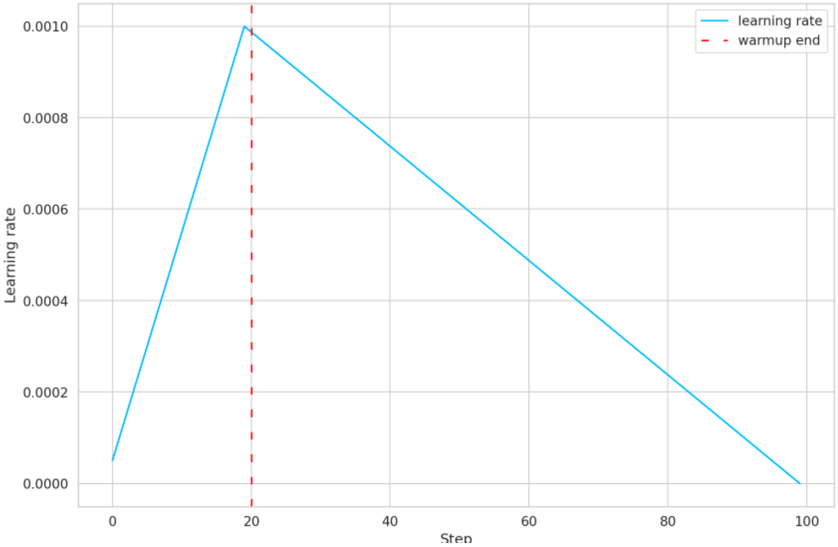
```
  | Name       | Type      | Params
-----------------------------------------
0 | model      | BertModel | 108 M
1 | classifier | Linear    | 769
2 | criterion  | BCELoss   | 0
3 | relu       | ReLU      | 0
4 | layer_1    | Linear    | 590 K
-----------------------------------------
108 M     Trainable params
0         Non-trainable params
108 M     Total params
435.607   Total estimated model params size (MB)
```

Loss BackPropagation and Optimizer

The loss criterion used here was the Binary Cross Entropy Loss (BCELoss())

$$\ell(x,y) = L = \{l_1, \dots, l_N\}^\top, \quad l_n = -w_n \left[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n) \right],$$

Above is the equation for BCEloss/log loss. Here *y* is the true label and x_n is the predicted probability. The backpropagation for this loss is carried out and the optimizer used is the *AdamW* optimizer. A linear warmup scheduler is also used which basically increases the learning rate up to a number of warmup steps and then linearly reduces it to 0 for the rest of the training steps.



Hyperparameters

The following hyperparameters were used for the entire process. These include model-specific as well as training specific hyperparameters

Hyperparameter	Value	Function
Max Learning Rate	4e-5	The maximum rate of change the training weights can reach during the training
Batch size	6	Number of data points in a single batch of training

Adam Epsilon	1e-8	Used for avoiding divide by zero error in the AdamW optimizer equations
Number of epochs	4	Number of iterations of training
Gradient Accumulation Steps	16	The number of steps over which the gradient is accumulated over for each step
Warmup Steps	The number of training steps divided by 6. Here the number of training steps is equal to ((Number of training data points) divided by (Batch size * Number of gradient accumulation steps)) * (Number of epochs)	Number of steps till which the learning rate increases to the max before decay starts

Output and Prediction

Our model outputs a single 1-dimensional tensor as its output which is a number between 0 to 1. For a prediction of 0, its number will be close to 0 and vice versa for 1. A threshold of **0.8** was used to predict the label as 1.

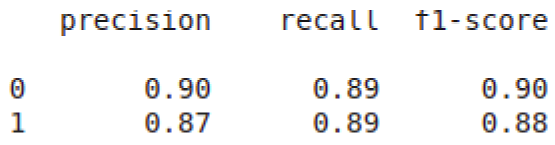
Results

After 4 epochs, the validation loss decreased from 0.69 to 0.3. The largest decrease was after the first epoch where the BCE Loss decreased from 0.69 to 0.39. After that, as the learning rate decreased linearly, loss decreased to 0.31.

For evaluation, the best checkpoint during training was saved and used.

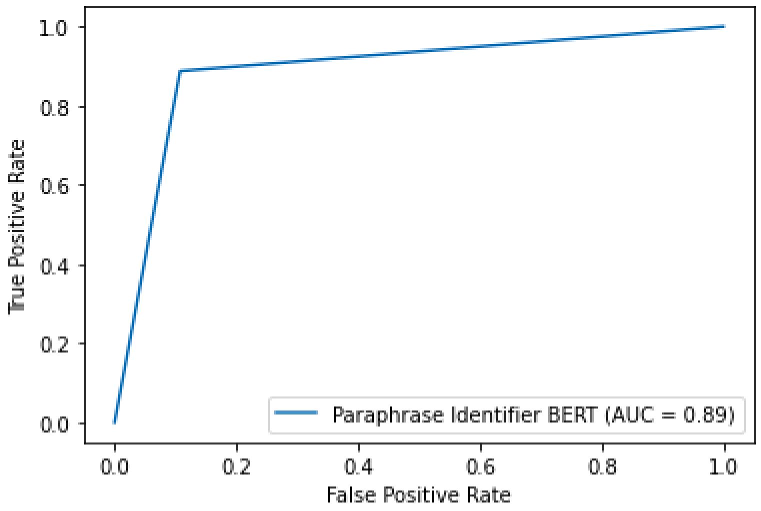
Accuracy - Over 11,000 sentence pairs in the test data, our model gave an accuracy of **89%.** We considered this number to be good considering the slight noisiness of our dataset due to the Quora question pairs

F1-Score - 88.1%



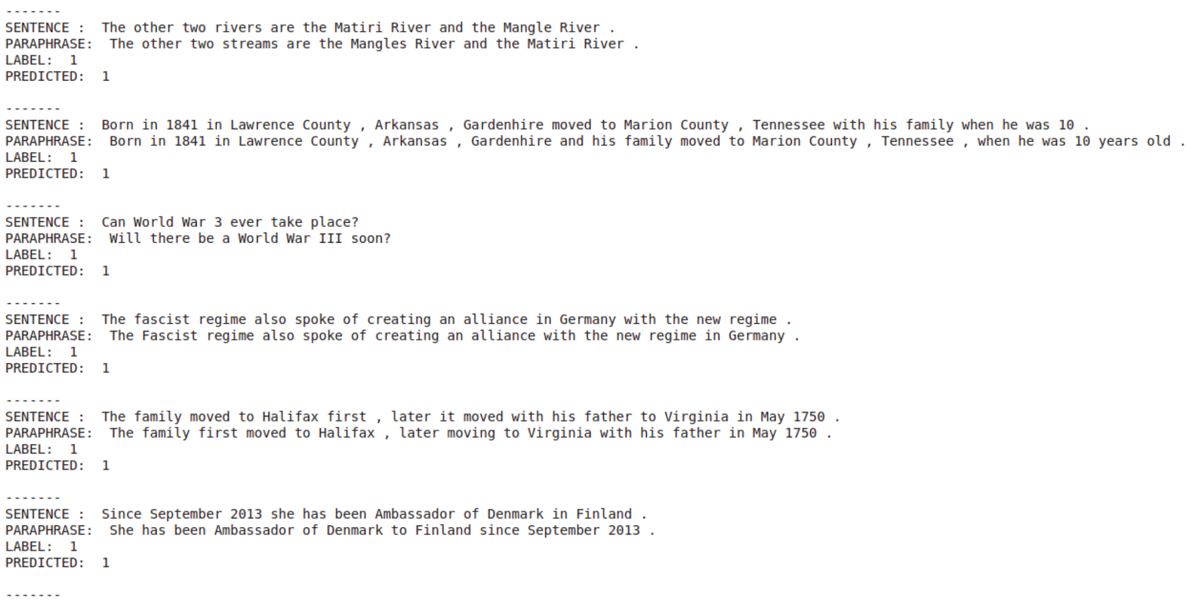
```
     precision    recall  f1-score

0         0.90      0.89      0.90
1         0.87      0.89      0.88
```

ROC-AUC

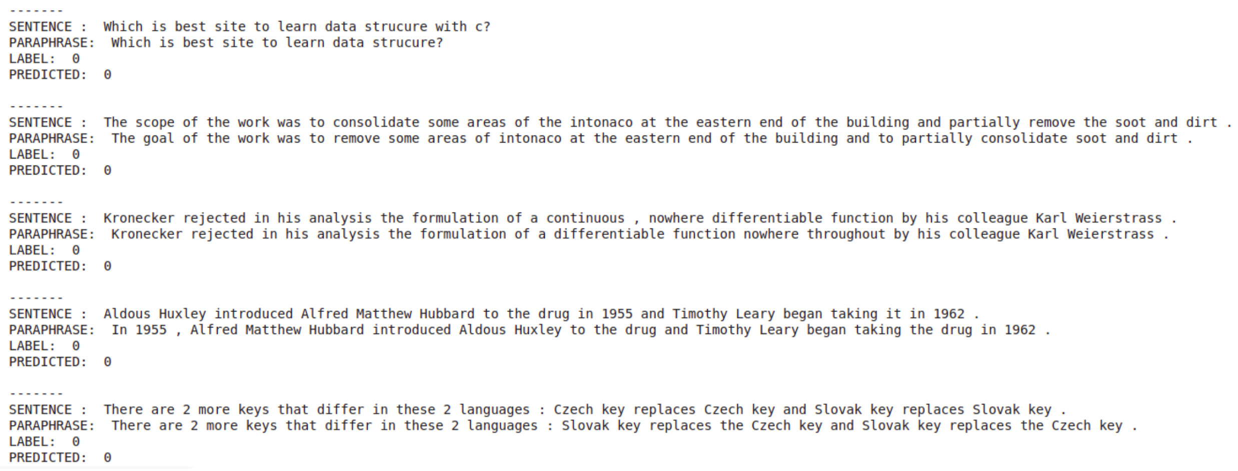


Sample Predictions

Correctly Labeled

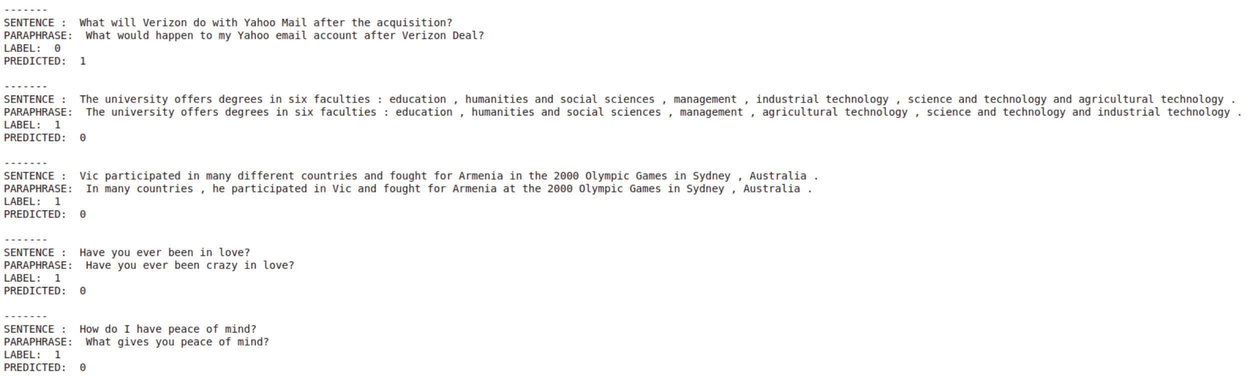


```
-------
SENTENCE :  The other two rivers are the Matiri River and the Mangle River .
PARAPHRASE:  The other two streams are the Mangles River and the Matiri River .
LABEL:  1
PREDICTED:  1

-------
SENTENCE :  Born in 1841 in Lawrence County , Arkansas , Gardenhire moved to Marion County , Tennessee with his family when he was 10 .
PARAPHRASE:  Born in 1841 in Lawrence County , Arkansas , Gardenhire and his family moved to Marion County , Tennessee , when he was 10 years old .
LABEL:  1
PREDICTED:  1

-------
SENTENCE :  Can World War 3 ever take place?
PARAPHRASE:  Will there be a World War III soon?
LABEL:  1
PREDICTED:  1

-------
SENTENCE :  The fascist regime also spoke of creating an alliance in Germany with the new regime .
PARAPHRASE:  The Fascist regime also spoke of creating an alliance with the new regime in Germany .
LABEL:  1
PREDICTED:  1

-------
SENTENCE :  The family moved to Halifax first , later it moved with his father to Virginia in May 1750 .
PARAPHRASE:  The family first moved to Halifax , later moving to Virginia with his father in May 1750 .
LABEL:  1
PREDICTED:  1

-------
SENTENCE :  Since September 2013 she has been Ambassador of Denmark in Finland .
PARAPHRASE:  She has been Ambassador of Denmark to Finland since September 2013 .
LABEL:  1
PREDICTED:  1

-------
```



```
-------
SENTENCE :  Which is best site to learn data strucure with c?
PARAPHRASE:  Which is best site to learn data strucure?
LABEL:  0
PREDICTED:  0

-------
SENTENCE :  The scope of the work was to consolidate some areas of the intonaco at the eastern end of the building and partially remove the soot and dirt .
PARAPHRASE:  The goal of the work was to remove some areas of intonaco at the eastern end of the building and to partially consolidate soot and dirt .
LABEL:  0
PREDICTED:  0

-------
SENTENCE :  Kronecker rejected in his analysis the formulation of a continuous , nowhere differentiable function by his colleague Karl Weierstrass .
PARAPHRASE:  Kronecker rejected in his analysis the formulation of a differentiable function nowhere throughout by his colleague Karl Weierstrass .
LABEL:  0
PREDICTED:  0

-------
SENTENCE :  Aldous Huxley introduced Alfred Matthew Hubbard to the drug in 1955 and Timothy Leary began taking it in 1962 .
PARAPHRASE:  In 1955 , Alfred Matthew Hubbard introduced Aldous Huxley to the drug and Timothy Leary began taking the drug in 1962 .
LABEL:  0
PREDICTED:  0

-------
SENTENCE :  There are 2 more keys that differ in these 2 languages : Czech key replaces Czech key and Slovak key replaces Slovak key .
PARAPHRASE:  There are 2 more keys that differ in these 2 languages : Slovak key replaces the Czech key and Slovak key replaces the Czech key .
LABEL:  0
PREDICTED:  0
```

Incorrectly Labeled



```
-------
SENTENCE :  What will Verizon do with Yahoo Mail after the acquisition?
PARAPHRASE:  What would happen to my Yahoo email account after Verizon Deal?
LABEL:  0
PREDICTED:  1

-------
SENTENCE :  The university offers degrees in six faculties : education , humanities and social sciences , management , industrial technology , science and technology and agricultural technology .
PARAPHRASE:  The university offers degrees in six faculties : education , humanities and social sciences , management , agricultural technology , science and technology and industrial technology .
LABEL:  1
PREDICTED:  0

-------
SENTENCE :  Vic participated in many different countries and fought for Armenia in the 2000 Olympic Games in Sydney , Australia .
PARAPHRASE:  In many countries , he participated in Vic and fought for Armenia at the 2000 Olympic Games in Sydney , Australia .
LABEL:  1
PREDICTED:  0

-------
SENTENCE :  Have you ever been in love?
PARAPHRASE:  Have you ever been crazy in love?
LABEL:  1
PREDICTED:  0

-------
SENTENCE :  How do I have peace of mind?
PARAPHRASE:  What gives you peace of mind?
LABEL:  1
PREDICTED:  0
```

Mostly the incorrectly labelled were questions from the QQP dataset.

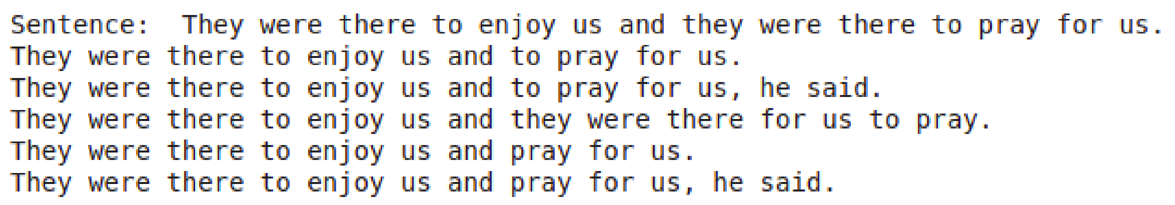
Generation

To achieve the paraphrase generation task the text-to-text transformer algorithm is used. A T5-base pre-trained model from hugging face is used which is then fine-tuned on the task of paraphrase generation. While fine-tuning various combinations of hyperparameters were tried. The trained model was then used to generate paraphrases by using a combination of Top-P and Top-K nucleus sampling. Analysis was done on the generated paraphrases using two classes of scores - Ngram similarity/dissimilarity(BLEU, PINC) and Semantic Similarity(using pre-trained SentenceBERT embeddings). Finally, we used the model trained for Paraphrase Identification as the evaluation metric.

Baseline Tried

The BART model was tried initially as the baseline. Specifically BartForConditionalGeneration from the transformers library

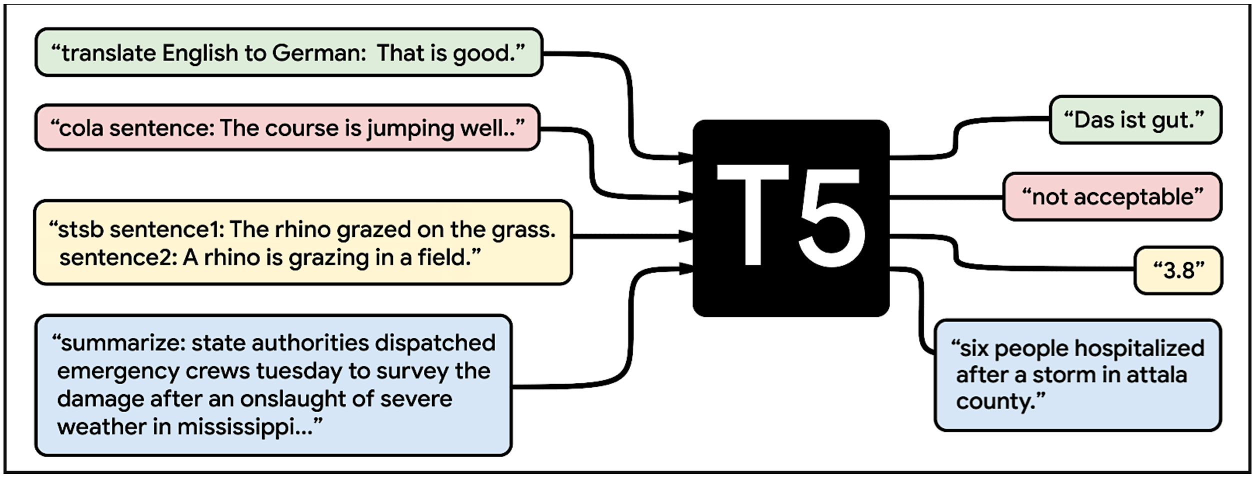
Here is a sample output from the above:



```
Sentence:  They were there to enjoy us and they were there to pray for us.
They were there to enjoy us and to pray for us.
They were there to enjoy us and to pray for us, he said.
They were there to enjoy us and they were there for us to pray.
They were there to enjoy us and pray for us.
They were there to enjoy us and pray for us, he said.
```

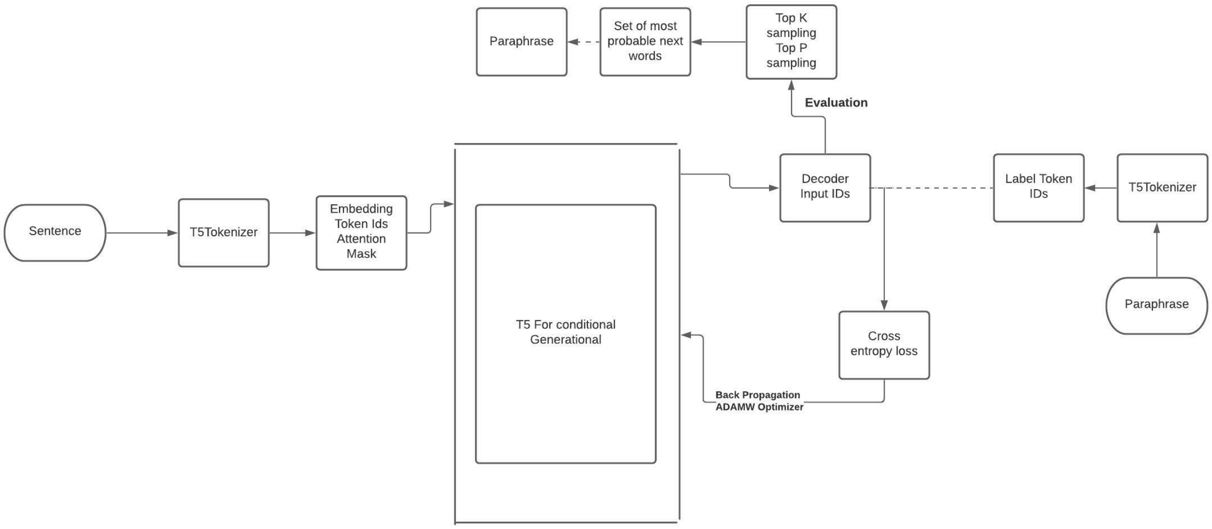
Why T5?

In transfer learning, an algorithm is first trained on a data-rich task (general/open or closed domain data) and then the trained model is finetuned on another downstream task. The Text-To-Text Transfer Transformer (T5) algorithm aims to convert every language problem into a text-to-text format. T5 is trained on a mix of labeled (Colossal Clean Crawled Corpus) and unlabelled data.


"translate English to German: That is good."
"cola sentence: The course is jumping well.."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi..."
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

The T5 model gives state-of-the-art results on more than 20 NLP tasks. Hardly any technique performs consistently as T5 while preserving flexibility to train on any downstream task. It is quite different from the BERT-style models that can only output either a class label or a span of the input.

Architecture And Training Process

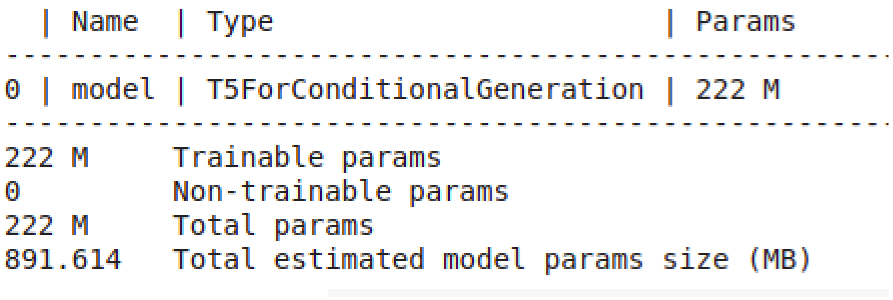

Paraphrase
Set of most probable next words
Top K sampling Top P sampling
Evaluation
Sentence
T5Tokenizer
Embedding Token Ids Attention Mask
T5 For conditional Generational
Decoder Input IDs
Label Token IDs
T5Tokenizer
Paraphrase
Cross entropy loss
Back Propagation ADAMW Optimizer

Tokenizing

The T5-Tokenizer was used. Here the pair of paraphrases was given as input to the tokenizer which then generated the tokens and attention masks. The prefix "Paraphrase: " was added at the start of each sentence to have a specifying token for this particular task

Model Layers

The pretrained T5-base model imported from the hugging face library contains approximately 220M parameters having 12 layers, 3072 feed forwards hidden states, 768-hidden layers, and 12- heads. These layers were used to feed forward the Input token Ids, Input attention mask and decoder attention mask. Along with this, the label token ids (token ids of the label paraphrase sentence) were also fed in the layers. The forward function then outputs a loss using the above.



```
  | Name  | Type                       | Params
-----------------------------------------------------
0 | model | T5ForConditionalGeneration | 222 M
-----------------------------------------------------
222 M     Trainable params
0         Non-trainable params
222 M     Total params
891.614   Total estimated model params size (MB)
```

The model was fine-tuned using the help of the PyTorch lightning module. During the training process, the mini-batch gradient descent algorithm was used. At the end of each epoch, the average training and validation loss was logged. The loss function used was the standard Cross entropy loss.

For **Back Propagation and Optimizers,** AdamW was used along with a linear warmup scheduler as explained above in the classifier section

Hyper Parameters

Final hyperparameters list used to finetune T5 Model for Paraphrase Generation Task :

Hyperparameter	**Value**
max sequence length (number of tokens)	512
learning rate	2e-5
adam epsilon (for the adam optimizer)	1e-8
training batch size	6
accumulate grad batches (to accumulate the gradients of multiple batches)	16
warmup steps (for the learning rate scheduler)	4
gradient clip value	0.5
number of epochs	4

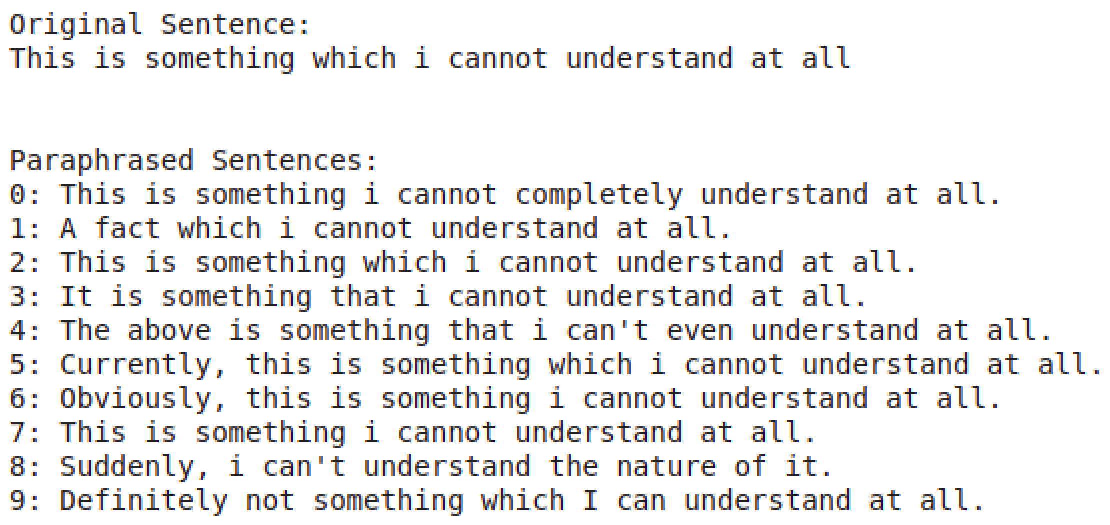
Outputs

For the actual generation task, the model uses **Top-k and Top-p nucleus sampling.**

In Top-k sampling, the *K* most likely next words are filtered and the probability mass is redistributed among only those *K* next words

Top-p sampling chooses from the smallest possible set of words whose cumulative probability exceeds the probability p.

A combination of these methods is used to generate the best probable set of next words given an input sentence. And so a paraphrase is generated using these words.



```
Original Sentence:
This is something which i cannot understand at all


Paraphrased Sentences:
0: This is something i cannot completely understand at all.
1: A fact which i cannot understand at all.
2: This is something which i cannot understand at all.
3: It is something that i cannot understand at all.
4: The above is something that i can't even understand at all.
5: Currently, this is something which i cannot understand at all.
6: Obviously, this is something i cannot understand at all.
7: This is something i cannot understand at all.
8: Suddenly, i can't understand the nature of it.
9: Definitely not something which I can understand at all.
```

Evaluation:

One of the limitations to the development of the paraphrasing task is the lack of standard metrics
such as BLEU, which has played a crucial role in driving progress in Machine Translation.

It must be noted that in order for a paraphrase to be considered "good", it should be lexically dissimilar to the source sentence while preserving the meaning.

We evaluate our Paraphrase Generation task using the following metrics:

BLEU Score:

The Bi-Lingual Evaluation Understudy score, or BLEU for short, proposed by Papineni et al (2002), is a metric for evaluating a generated sentence to a reference sentence.

The primary idea in BLEU is to compare n-grams of the candidate with the n-grams of the reference translation and count the number of matches (position independent).

A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0.

For this task, BLEU helps us measure semantic adequacy and fluency.

However, BLEU alone would not be a good metric to consider; an optimal paraphrasing engine would simply return the input if BLEU were the sole evaluation metric.

PINC Score:

Chen and Dolan [15] proposed a new evaluation metric, PINC (Paraphrase In N-gram Changes), that relies on simple BLEU-like n-gram comparisons to measure the degree of novelty of automatically generated paraphrases. PINC score helps us measure lexical dissimilarity since we essentially want to minimize the number of n-gram overlaps between the two sentences. PINC score for a source sentence s and a candidate sentence c can be mathematically written as follows:

$$PINC(s,c) = \frac{1}{N}\sum_{n=1}^{N} 1 - \frac{|ngram_s \cap ngram_c|}{|ngram_c|}$$

N is the maximum n-gram considered and $ngram_s$ and $ngram_c$ are the lists of n-grams in the
source and candidate sentences, respectively.

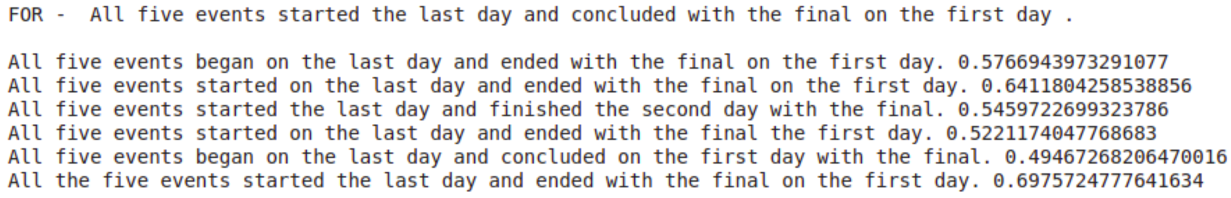
For our experiments, we choose N=4, while computing the PINC score.

N-gram similarity score:

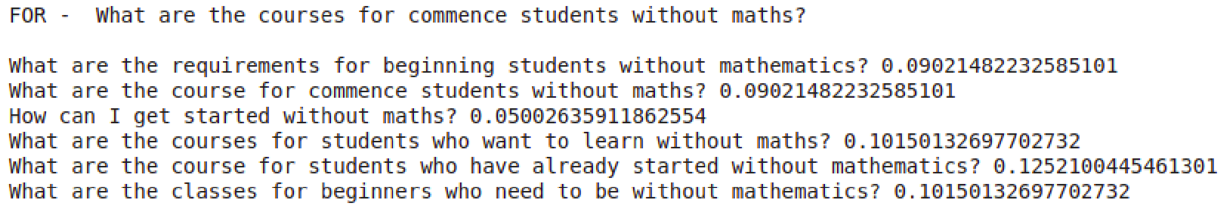
Since both the PINC and BLUE scores are n-gram based, we combine them in order to get a single N-gram based score. We use this score to evaluate our generation model outputs. The key idea behind using such a score was to balance the two aspects to a good paraphrase: preserved meaning and lexical dissimilarity. We know that a high PINC score would imply that the source and candidate sentences are highly different, hence we consider (1 - PINC score) for the n-gram similarity score.

$$NGramScore = (BLEUscore + (1 - PINCscore))/2$$

Here are some examples of how the n-gram scores perform on paraphrases



```
FOR -  All five events started the last day and concluded with the final on the first day .

All five events began on the last day and ended with the final on the first day. 0.5766943973291077
All five events started on the last day and ended with the final on the first day. 0.6411804258538856
All five events started the last day and finished the second day with the final. 0.5459722699323786
All five events started on the last day and ended with the final the first day. 0.5221174047768683
All five events began on the last day and concluded on the first day with the final. 0.49467268206470016
All the five events started the last day and ended with the final on the first day. 0.6975724777641634
```



```
FOR -  What are the courses for commence students without maths?

What are the requirements for beginning students without mathematics? 0.09021482232585101
What are the course for commence students without maths? 0.09021482232585101
How can I get started without maths? 0.05002635911862554
What are the courses for students who want to learn without maths? 0.10150132697702732
What are the course for students who have already started without mathematics? 0.1252100445461301
What are the classes for beginners who need to be without mathematics? 0.10150132697702732
```

Semantic Similarity Score:

This metric was used to help us identify if the paraphrases are semantically similar.

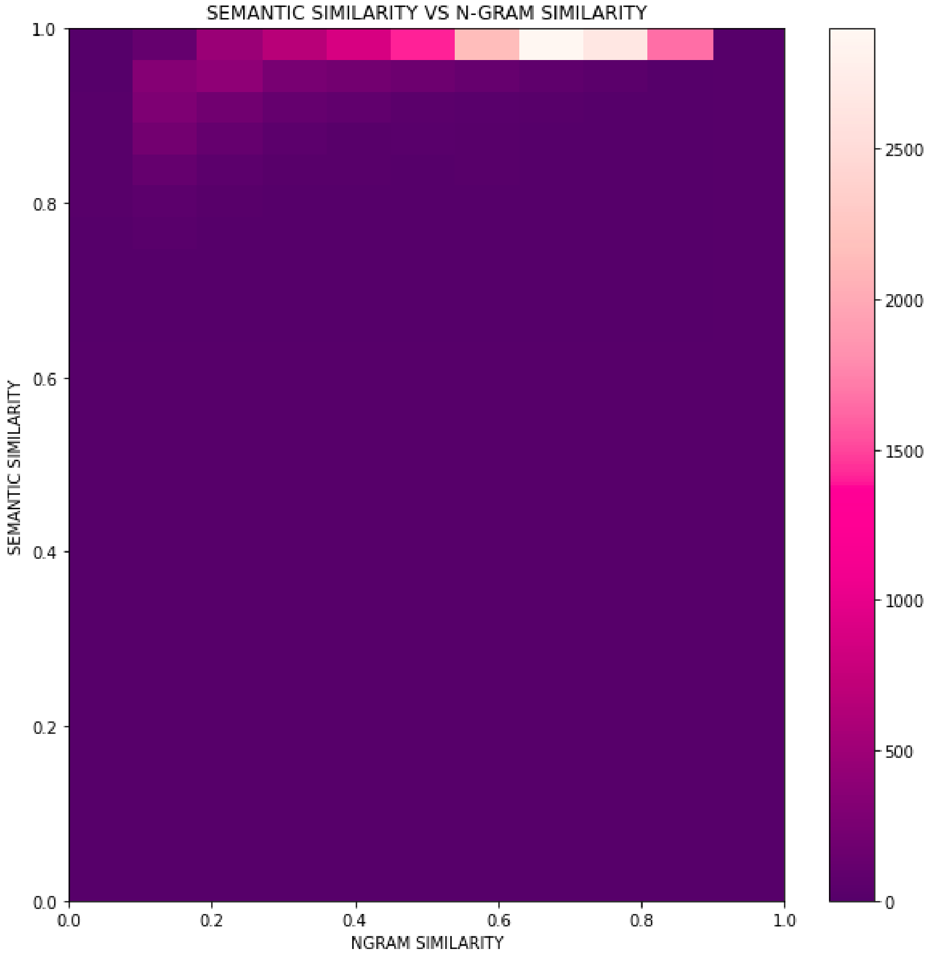
To further evaluate our generated paraphrases we used an already pre-trained Sentence BERT Model from hugging face. Sentence-BERT (SBERT), is a modification of the pretrained BERT network that uses Siamese and Triplet-network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity.

Using the model we first get the vector embeddings of the generated pair of paraphrases. Then we calculate the cosine similarity of the two sentences, thereby getting the semantic score.

Analysis of the scores

We decided to use these two dimensions - semantic score and n-gram similarity score to analyze the quality of our paraphrases. We took a sample of sentences from our test dataset and generated around 5-20 paraphrases for each sentence. Then we calculated the semantic as well as n-gram similarity scores for all these pairs. We then plotted a histogram with **semantic similarity as the y-axis and n-gram similarity as the x-axis.**

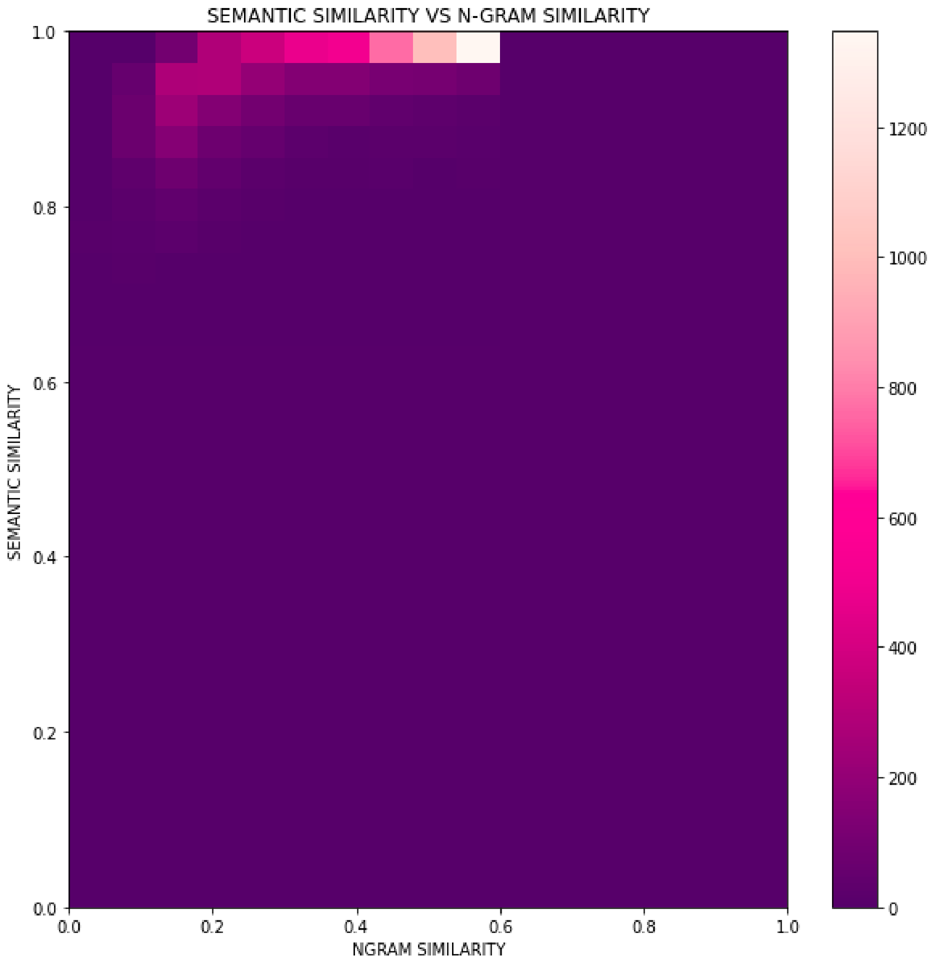
Given below are the plots



Semantic similarity vs Ngram similarity 2D histogram

We saw that almost all the scores are accumulated at the top of the plot and slightly towards the middle. This tells us that even though sentence paraphrase pairs have moderate n-gram similarity, they still maintain their semantic sense.

Here is a plot with the pairs with an n-gram score of more than 0.6 removed



Again it is clearly visible that a decrease in n-gram similarity does not decrease semantic scores and only if the n-gram score is very low (0.2-0.3) then the semantic score drops very slightly (the slight shading on the top left of the plot)

Using our fine-tuned BERT Classifier for evaluation

In order to evaluate our paraphrase generator's performance, we decided to use the paraphrase identification model that we built.

We first took 1000 randomly sampled sentences from our dataset to use for the evaluation. These sentences were chosen from the entire dataset with equal probability. The 1000 sentences were then fed into the paraphrase generation model, which generated at most 20 paraphrases for each input sentence. The number of paraphrases was different for each input sentence, but we decided to keep an upper limit of 20.

After having at most 20 sentences generated for each input sentence, we created sentence pairs which were tested using the paraphrase identification model. Each pair consisted of the input sentence and one of its corresponding generated sentences. The identifier then outputted a score of 0 or 1 for each pair.

Score Calculation

For each of the 1000 input sentences, we calculated the average number of pairs that returned positive from the identification model and assigned that average as the score for that particular sentence. We then calculated the average score of all the 1000 input sentences and returned that as the accuracy of our model.

This can be mathematically represented as follows:

Let s_i be the i^{th} input sentence, and s_i^j be the j^{th} paraphrase generated from s_i. Then, the score of s_i is:

$$score_i = \frac{1}{K}\sum_{j=1}^{K} f(s_i, s_i^j)$$

where K is the number of paraphrases generated for s_i (max 20) and,

$$f(x,y) = \begin{cases} 1 & if\ x\ and\ y\ are\ paraphrases \\ 0 & if\ x\ and\ y\ are\ not\ paraphrases \end{cases}$$

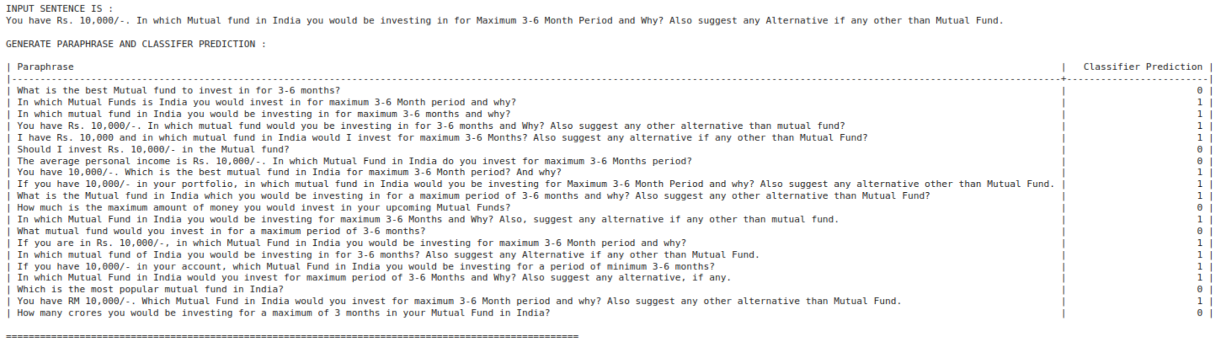
Now, the overall classifier-score can be calculated as:

$$Classifier\ Score = \frac{1}{N}\sum_{i=1}^{N} score_i$$

Where N is the total number of input sentences, which we took as 1000

The final number that we got on the randomly selected 1000 sentences was 0.90597.

Sample Generated Paraphrases and their corresponding predictions :



```
INPUT SENTENCE IS :
You have Rs. 10,000/-. In which Mutual fund in India you would be investing in for Maximum 3-6 Month Period and Why? Also suggest any Alternative if any other than Mutual Fund.

GENERATE PARAPHRASE AND CLASSIFER PREDICTION :

| Paraphrase                                                                                                                                                                  |   Classifier Prediction |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------------------|
| What is the best Mutual fund to invest in for 3-6 months?                                                                                                                   |                       0 |
| In which Mutual Funds is India you would invest in for maximum 3-6 Month period and why?                                                                                    |                       1 |
| In which mutual fund in India you would be investing in for maximum 3-6 months and why?                                                                                     |                       1 |
| You have Rs. 10,000/-. In which mutual fund would you be investing in for 3-6 months and Why? Also suggest any other alternative than mutual fund?                          |                       1 |
| I have Rs. 10,000 and in which mutual fund in India would I invest for maximum 3-6 Months? Also suggest any alternative if any other than Mutual Fund?                      |                       1 |
| Should I invest Rs. 10,000/- in the Mutual fund?                                                                                                                            |                       0 |
| The average personal income is Rs. 10,000/-. In which Mutual Fund in India do you invest for maximum 3-6 Months period?                                                     |                       0 |
| You have 10,000/-. Which is the best mutual fund in India for maximum 3-6 Month period? And why?                                                                            |                       1 |
| If you have 10,000/- in your portfolio, in which mutual fund in India would you be investing for Maximum 3-6 Month Period and why? Also suggest any alternative other than Mutual Fund. |                       1 |
| What is the Mutual fund in India which you would be investing in for a maximum period of 3-6 months and why? Also suggest any other alternative than Mutual Fund?          |                       1 |
| How much is the maximum amount of money you would invest in your upcoming Mutual Funds?                                                                                     |                       0 |
| In which Mutual Fund in India you would be investing for maximum 3-6 Months and Why? Also, suggest any alternative if any other than mutual fund.                           |                       1 |
| What mutual fund would you invest in for a maximum period of 3-6 months?                                                                                                    |                       0 |
| If you are in Rs. 10,000/-, in which Mutual Fund in India you would be investing for maximum 3-6 Month period and why?                                                      |                       1 |
| In which mutual fund of India you would be investing in for 3-6 months? Also suggest any Alternative if any other than Mutual Fund.                                         |                       1 |
| If you have 10,000/- in your account, which Mutual Fund in India you would be investing for a period of minimum 3-6 months?                                                 |                       1 |
| In which Mutual Fund in India would you invest for maximum period of 3-6 Months and Why? Also suggest any alternative, if any.                                              |                       1 |
| Which is the most popular mutual fund in India?                                                                                                                             |                       0 |
| You have RM 10,000/-. Which Mutual Fund in India would you invest for maximum 3-6 Month period and why? Also suggest any other alternative than Mutual Fund.                |                       1 |
| How many crores you would be investing for a maximum of 3 months in your Mutual Fund in India?                                                                              |                       0 |

=================================================================================================
```

We took two 0 predictions and two 1 predictions and analyzed their BLEU and PINC scores (with reference to the source sentence):

What is the best Mutual fund to invest in for 3-6 months? BLEU: 0.04269512076670416 PINC: 0.7878787878787878

What mutual fund would you invest in for a maximum period of 3-6 months? BLEU: 0.06592872322968105 PINC: 0.7293956043956044

In which Mutual Fund in India would you invest for maximum period of 3-6 Months and Why? Also suggest any alternative, if any. BLEU: 0.3222383297853335 PINC: 0.5013998682476943

If you have 10,000/- in your portfolio, in which mutual fund in India would you be investing for Maximum 3-6 Month Period and why? Also suggest any alternative other than Mutual Fund. BLEU: 0.5872543877934288 PINC: 0.3758864015572859

A very low BLEU score (< 0.1) and a high PINC score (> 0.7) result in our classifier predicting 0. However, even for moderate to low BLEU scores of 0.3 to 0.5, it successfully recognizes paraphrases.

References

1. [Zhang et al., 2019] Zhang, Y., Baldridge, J., and He, L. (2019). PAWS: Paraphrase Adversaries from Word Scrambling. InProc. of NAACL
2. [Palivela, 2021] Palivela, H. (2021). Optimization of paraphrase generation and identification using language models in natural language processing.International Journal of Information Management Data Insights, 1:100025.
3. [Raffel et al., 2020] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y.,Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer.
4. [Papineni et al., 2002] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. InProceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL ’02, page 311–318, USA. Association for Computational Linguistics.
5. [Lin, 2004] Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
6. [Banerjee and Lavie, 2005] Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. InProceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.
7. [Szymański et al., 2020] Piotr Szymański and Piotr Żelasko and Mikolaj Morzy and Adrian Szymczak and Marzena Żyła-Hoppe and Joanna Banaszczak and Lukasz Augustyniak and Jan Mizgajski and Yishay Carmiel. WER we are and WER we think we are
8. [Mutton et al., 2007] Mutton, A., Dras, M., Wan, S., and Dale, R. (2007). Gleu: Automatic evaluation of sentence-level fluency.
9. [Hegde and Patil, 2020] Hegde, C. and Patil, S. (2020). Unsupervised paraphrase generation using pre-trained language models.
10. [Radford et al., 2019] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

11. [Reimers and Gurevych, 2019] Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence em-beddings using siamese bert-networks.
12. [Yang et al., 2019] Yang, Y., Zhang, Y., Tar, C., and Baldridge, J. (2019). Paws-x: A cross-lingual adversarial dataset for paraphrase identification.
13. [Yang et al., 2020] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V.(2020). Xlnet: Generalized autoregressive pretraining for language understanding.
14. [Devlin et al., 2018] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805.
15. [Chen and Dolan, 2011] Chen, D. L. and Dolan, W. B. (2011). Collecting highly parallel data for paraphrase evaluation. InProceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT ’11, page 190–200, USA.Association for Computational Linguistics.