Clickbait Intensity and Style Analysis

A project report submitted as part of the requirements for the course titled Information Retrieval and Extraction - Monsoon 2022

By
Team Number : 1
Team Name : The Retrieval
Gopichand Kanumolu (2021701039), gopichand.kanumolu@research.iiit.ac.in
Lokesh Madasu (2021701042), lokesh.madasu@research.iiit.ac.in

International Institute of Information Technology
Hyderabad

18, November, 2022

Contents

1 Introduction

1.1 Task Description

A clickbait can be defined as a link with a headline that lures the users to click on it to see more, without explicitly mentioning what they will see when they click on the link. It creates curiosity among the users by creating information gap. Online social media agencies use clickbaits to attract the users and attempts to increase page visits and shares to increase revenue by showing advertisements.

1.2 Problem Statement

Detecting whether a headline is clickbait or not is not a trivial task, since they use catchy words and typically shorter length sentences, it is difficult for the models to understand the way in which they are written. It requires higher order meaning understanding. However, just detecting and classifying the clickbaity headlines does not fully serve the purpose and it is important to predict how intensive the clickbait is. Since, not all the clickbaity headlines are equally intensive, it is important to figure out intensity of the clickbaity headline. In addition to that, identifying the words/phrases in the headline that are actually making it clickbaity and coming up with a way to increase/decrease the clickbait intensity by substituting these words in a novel way is a challenging task.

1.2.1 Why it is important?

Clickbaits often wastes readers time by not living up to the expectations in terms of information quality and relevance to the headline and loses users trust. This can create negative impact on the website by ranking algorithms. It is important for the organization/media to control the intensity of the clickbait and have a balance between the curiosity they create and the amount of information they provide. From the societal benefit point of view, to prevent the spread of misleading information and content from uncertified sources it is very important to detect and control the clickbaits.

2 Dataset

2.1 Dataset Description

We use the Webis Clickbait Corpus, which contains tweets and their clickbait intensities that are manually annotated by the annotators. Along with the tweets (post text) and intensity scores it also contains other attributes like timestamp, target paragraph, target keywords, class(clickbait/non clickbait), mean, median and mode values of the intensity scores given by the annotators.

2.2 Dataset Statistics

Field	Train	Test
Number of samples in the data	17506	4341
Number of click bait samples in the data	4281	1104
Maximum sequence length	267	476
Minimum sequence length	1	1
Average sequence length	12	11
Total number of words	206521	*
Total number of unique words	39231	*

Table 1: dataset statistics

2.3 Distribution Plots

2.3.1 Number of Tokens in Text Vs Frequency

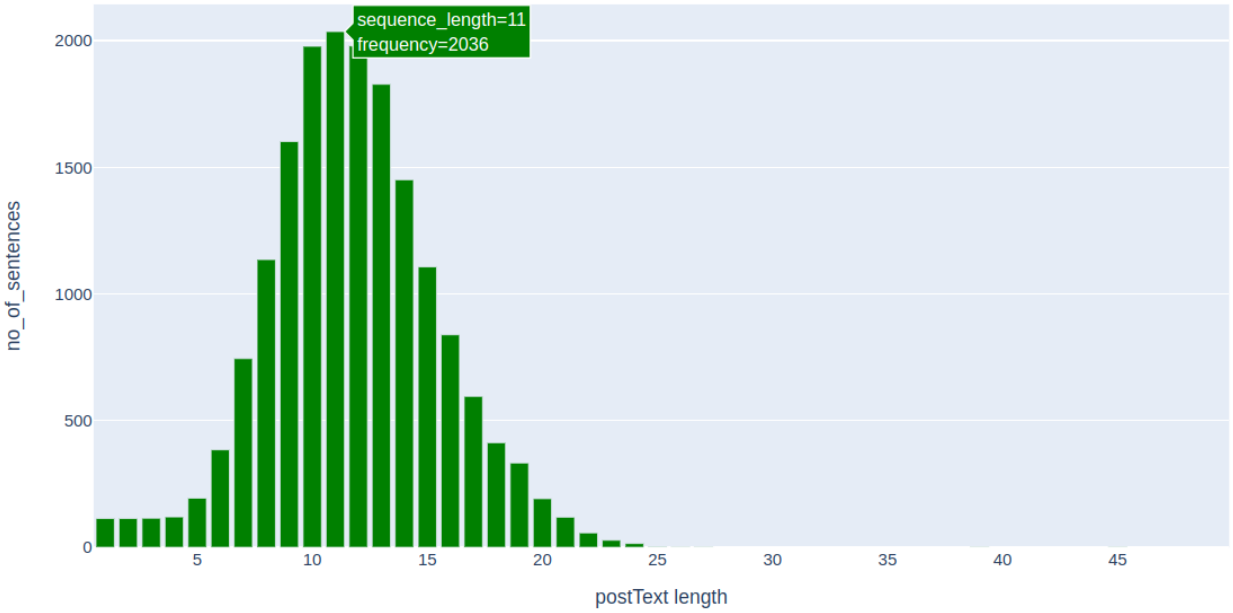


Figure 1: Number of Tokens in Text Vs Frequency

The distribution follows normal distribution. We can observe from the figure that there are 2036 samples in the training data with number of tokens equal to 11.

2.3.2 Intenstity Distribution

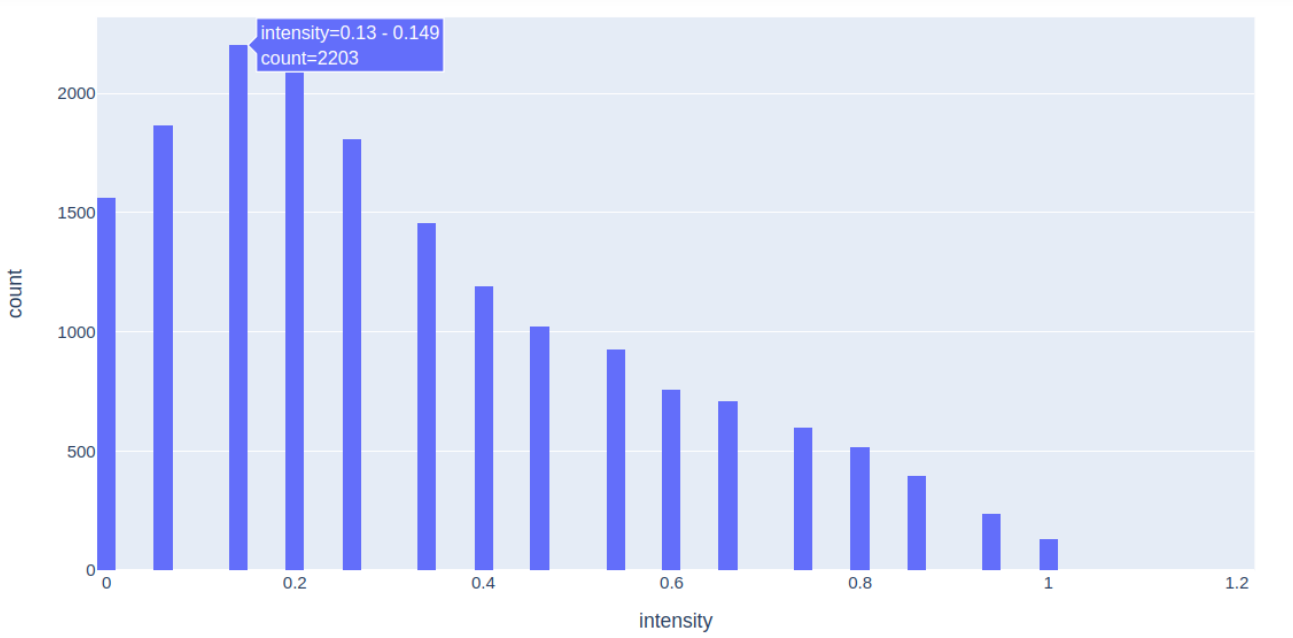


Figure 2: Intensity Range Vs Frequency

We can observe from the figure that there are 2203 samples in the train dataset withing the intensity range 0.13 to 0.149.

2.3.3 Vocabulary Vs Frequency

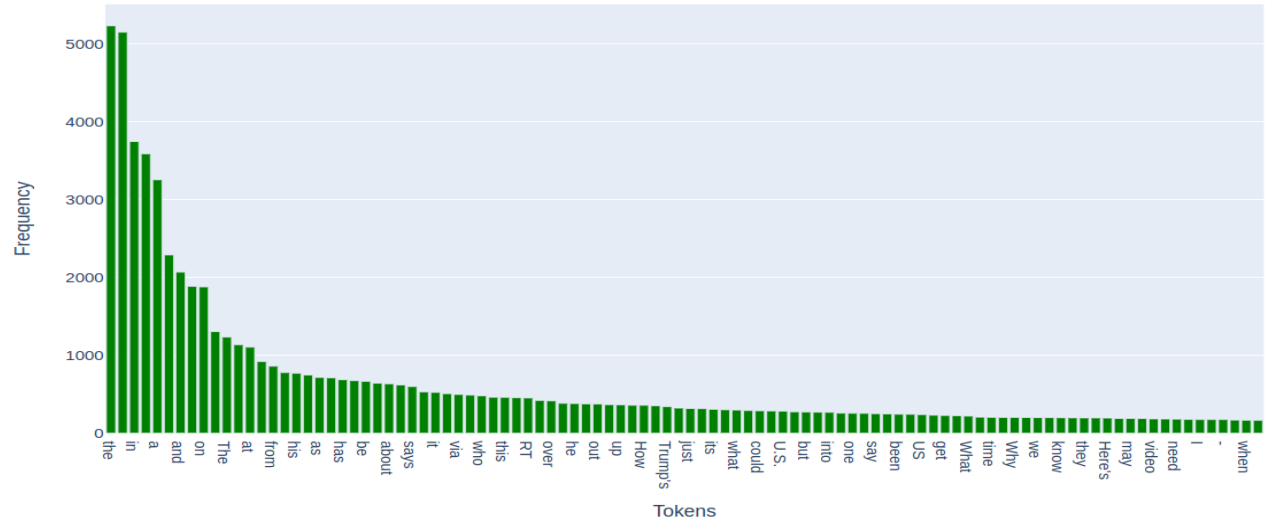


Figure 3: Vocabulary Vs Frequency

The most frequent words along with their frequencies in descending order.

3 Baseline Model Implementation

3.1 Existing Approaches

Existing approaches use classification algorithms like Logistic Regression, Support Vector Machines, Decision Trees, etc. to detect and classify the given text as clickbait or not clickbait. To measure the intensity of the clickbaityness of a text, there are approaches that uses various regression models like simple linear regression, lasso & ridge regression, support vector regression, decision trees, random forest and ensemble based regression techniques. And to represent text into vector format which will be fed as input to the models, various word embedding techniques(Contextual and Non-contextual) like BERT,BERT Large, Roberta, GPT-2, ELMO, Word2Vec, Fasttext, etc. are used.

3.1.1 Regression Models

We train the following regression models on various kinds of word, sentence and transformer representations.

1. **Simple Linear Regression (LR)**

   Linear Regression is the most basic regression algorithm. For a given word embedding vector X, it tries to find out the weights vector W which results in the minimum value of the cost function J(W). The cost function is Mean Squared Error (MSE).

2. **Ridge Regression (RR)**

   The Cost function is same as linear regression. In addition to that, it uses L2 penalty as a regularization term.

$$CostFunciton = \frac{1}{N}\sum_{i=0}^{N}(Y_i - \hat{Y_i})^2 + \lambda \sum_{k=1}^{m} ||w_k||$$

3. **Gradient Boosted Regression (GBR)**:
   It learns an ensemble of regression trees, each of which have scalar values in the leaves. The ensemble of trees is produced by computing, in each step, a regression tree that approximates the gradient of the loss function, and adding it to the previous tree with coefficients that minimize the loss of the new tree. The output of the ensemble on a given instance is the sum of the tree outputs.

4. **Random Forest Regression (RFR)**
   A random forest regressor is an ensemble model which uses multiple decision trees to learn the regresser. Here, each decision tree uses subset of the total data. The output will be the average of all trees outputs.

5. **Adaboost Regression (ABR** AdaBoost regressor is another ensemble learning algorithm that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but the weights of instances are adjusted according to the error of the current prediction, thereby subsequent regressors focus more on the difficult cases.

3.1.2 Word/Sentence Embeddings

3.1.3 BERT

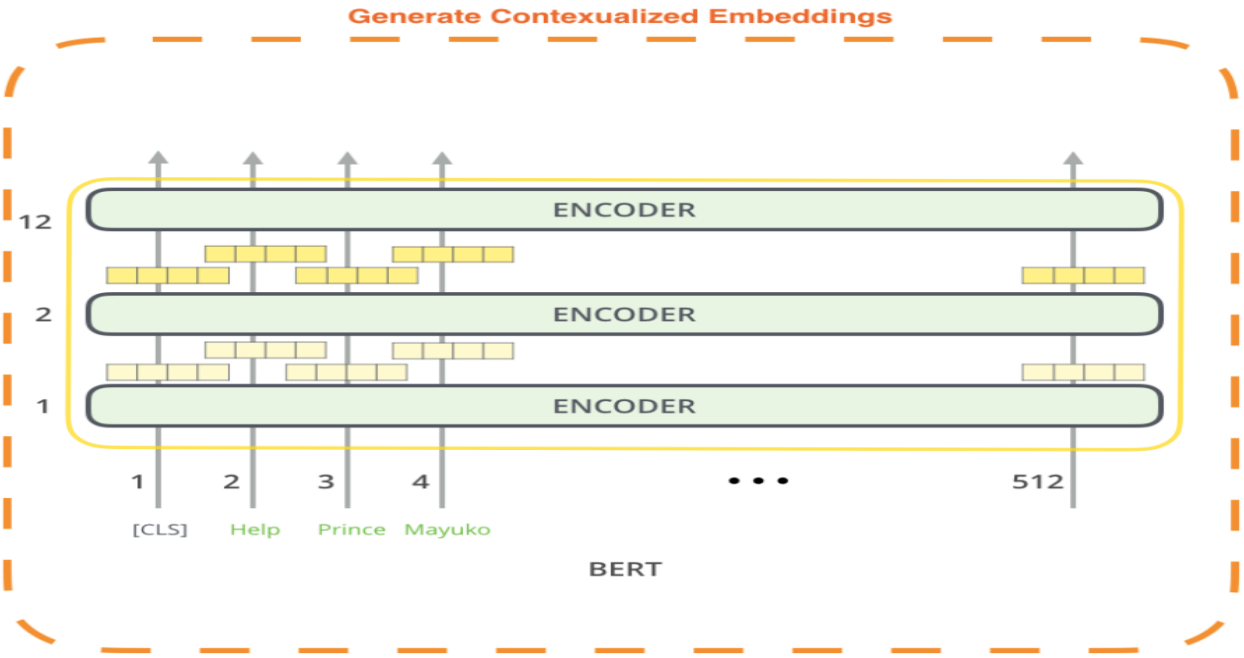


Figure 4: BERT

BERT is a transformers model pretrained on a large corpus of English data in a self-supervised fashion. It was pretrained with two objectives:
i) Masked language modeling (MLM): taking a sentence, the model randomly masks 15% of the words in the input then run the entire masked sentence through the model and has to predict the masked words.
ii) Next sentence prediction (NSP): The model then has to predict if the two sentences were following each other or not.
This way, the model learns an inner representation of the English language that can then be used to extract features useful for downstream tasks.

We use two variants of BERT for getting sentence embeddings, which are BERT base and BERT Large with 110 Million and 340 Million parameters respectively

3.1.4 RoBERTa

RoBERTa stands for "Robustly Optimized BERT pre-training Approach". In many ways this is a better version of the BERT model. The key points of difference are as follows: It uses Dynamic Masking, It is trained only with the MLM objective, not NSP and it is trained on large corpus than BERT. We use two variants of RoBERTa for getting sentence embeddings, which are RoBERTa base and RoBERTa Large.

3.1.5 Universal Sentence Embeddings

One of the most well-performing sentence embedding techniques right now is the Universal Sentence Encoder. It can be used for multitasking, this means that the sentence embeddings we generate can be used for multiple tasks like sentiment analysis, text classification, sentence similarity, etc, and the results of these asks are then fed back to the model to get even better sentence vectors that before.

4 Evaluation Metrics

4.1 Mean Squared Error

Mean Squared Error(MSE) is the main evaluation metric as we would be using regression techniques to measure the intensity of the clickbaityness of the text. MSE is computed using the formula

$$MSE = \frac{1}{N}\sum_{i=0}^{N}(Y_i - \hat{Y_i})^2$$

Here, Y_i is ground truth value of sample-i, $\hat{Y_i}$ is the value predicted by the model for sample-i and N is the total number of samples in the dataset.

4.2 Median Absolute Error

The median absolute error is particularly interesting because it is robust to outliers. The loss is calculated by taking the median of all absolute differences between the target and the prediction. If $\hat{y}$ is the predicted value of the i^{th} sample and y_i is the corresponding true value, then the median absolute error estimated over n samples is defined as follows:

$$MedAE(y, \hat{y}) = median(|y_1 - \hat{y_1}|, ..., |y_n - \hat{y_n}|)$$

4.3 F1-Score

For classifying the headline as clickbait or not we are going to use F1-Score metric. It is the standard classification task metric that balance both precision and recall. F1-Score is computed using the formula below.

$$F_1 - Score = \frac{2 * precision * recall}{precision + recall}$$

Here, Precision quantifies the number of positive class predictions that actually belong to the positive class and Recall quantifies the number of positive class predictions made out of all positive examples in the dataset. Since we are predicting the intensity of the clickbait, which outputs a real number between 0 and 1, to compute the F1-Score, the headlines with intensity score greater than 0.5 are considered to be clickbait and the rest are considered as non clickbait.

4.4 Accuracy

Accuracy is one of the popular metric that is used for classification. It is defined as the percentage of samples that are correctly classified by the model and is given by the formula given below.

$$Accuracy = \frac{NumberofCorrectlyClassifiedSamples}{TotalNumberofSamples}$$

5 Baseline Results

5.1 Without Replacing/Masking Words

In this approach have reproduced the results in the baseline research paper by experimenting with various regression models and word/sentence embedding techniques. The results are mentioned in the below table. We observe that that combination of (RoBERTa, RR) and (RoBERTa, LR) produced better results, while the former has produced slightly better results than the latter.

Embedding	Model	MSE	MedAe	F1-Score	Accuracy
BERT_Large	LR	0.02935	0.1096	0.64638	0.84459
BERT_Large	RR	0.0292	0.10893	0.64451	0.84459
BERT_Large	GBR	0.03157	0.1196	0.61249	0.83837
BERT_Large	RFR	0.04623	0.15415	0.32181	0.77842
BERT_Large	ABR	0.04325	0.16596	0.5614	0.82707
RoBERTa	LR	0.02765	0.10627	0.67324	0.85474
RoBERTa	RR	0.02741	0.10668	0.67535	0.85635
RoBERTa	GBR	0.02983	0.115	0.63494	0.84436
RoBERTa	RFR	0.04243	0.14348	0.41534	0.78902
RoBERTa	ABR	0.03934	0.15569	0.63169	0.84298
RoBERTa_Large	LR	0.02723	0.10461	0.68863	0.8605
RoBERTa_Large	RR	**0.02695**	0.1037	**0.68705**	**0.86073**
RoBERTa_Large	GBR	0.02987	0.11596	0.62916	0.84344
RoBERTa_Large	RFR	0.04127	0.13801	0.5018	0.80862
RoBERTa_Large	ABR	0.04043	0.16543	0.59579	0.83606
UniversalSentenceEmbeddings	LR	0.03108	0.11351	0.63563	0.84298
UniversalSentenceEmbeddings	RR	0.03074	0.11423	0.63276	0.84183
UniversalSentenceEmbeddings	GBR	0.03147	0.11891	0.62179	0.83675
UniversalSentenceEmbeddings	RFR	0.04483	0.15534	0.41237	0.78972
UniversalSentenceEmbeddings	ABR	0.04397	0.1671	0.56019	0.82984
BERT	LR	0.02881	0.1119	0.6649	0.85382
BERT	RR	0.02876	0.11202	0.66526	0.85405
BERT	GBR	0.03168	0.11822	0.60346	0.83606
BERT	RFR	0.0448	0.14852	0.32143	0.78095
BERT	ABR	0.04256	0.16794	0.56838	0.83191

Table 2: Results Table Without Replacing Words

5.2 With Replacing/Masking Words

In this approach, We use Named Entity Recoginition(NER) one of the most popular data preprocessing task in NLP. It involves the identification of key information(entity detection) in the text and classification into a set of predefined categories(entity classification). An entity is basically the thing that is consistently talked about or refer to in the text. Example : Place, Name, Organization, Date, Currency, Number etc.

We apply NER on the given text and identify the non-named entities and on top of these we apply Parts Of Speech(POS) tagging and select the words/tokens that are 'PONOUNS', and 'DETERMINERS' and remove/mask these tokens in the original text and get the intensity score of the modified text.

The results of this approach using various regression and embedding models are mentioned in the table below. We observe that for some of the samples the intensity has increased for some the intensity has reduced.

Embedding	**Model**	**MSE**	**MedAe**	**F1-Score**	**Accuracy**
BERT	LR	0.03116	0.11328	0.59839	0.83906
BERT	RR	0.03111	0.11323	0.59827	0.83929
BERT	GBR	0.03446	0.11891	0.53358	0.82223
BERT	RFR	0.0469	0.14786	0.37022	0.78741
BERT	ABR	0.04312	0.16878	0.50882	0.82015
BERT_Large	LR	0.03177	0.11365	0.59804	0.83883
BERT_Large	RR	0.03161	0.11327	0.59804	0.83883
BERT_Large	GBR	0.03449	0.12138	0.52497	0.82015
BERT_Large	RFR	0.0485	0.15774	0.19703	0.7632
BERT_Large	ABR	0.04461	0.16862	0.44415	0.80724
RoBERTa_Large	LR	0.02964	0.11332	0.66153	0.84805
RoBERTa_Large	RR	0.02935	0.11305	0.65517	0.84782
RoBERTa_Large	GBR	0.03231	0.11724	0.58636	0.83214
RoBERTa_Large	RFR	0.04212	0.13645	0.52656	0.81093
RoBERTa_Large	ABR	0.0418	0.16634	0.51209	0.81854
RoBERTa	LR	0.03014	0.11132	0.60579	0.83975
RoBERTa	RR	0.02985	0.10961	0.6072	0.8416
RoBERTa	GBR	0.03247	0.11654	0.55003	0.82684
RoBERTa	RFR	0.04518	0.14465	0.38706	0.78165
RoBERTa	ABR	0.04084	0.15743	0.55601	0.82546

Table 3: Results With Replacing/Masking Words

6 Proposed Methods (Baseline Plus)

6.1 Problems With Existing Approaches

Existing approaches use regression algorithms to predict the intensity of the clickbait sentence which helps in determining the amount of clickbaityness contained in the sentence. However, Intensity is not alone useful for controlling the style of the clickbait. We need systems that can change the style of the clickbait using the intensity score. That is, we need to implement a deep learning model that can reduce the intensity of a given clickbait sentence by changing it's style.

To achieve that we start by using naive approach of replacing the words randomly to check how the presence or absence of words is changing the intensity and reported the results. The problem with this approach is that when we randomly replace words, the meaning of the sentence changes and also leads to the loss of important information. For example, consider the actual text "Hillary Clinton's gut-wrenching day", when we randomly replace a word, say "gut-wrenching" it becomes "Hillary Clinton's day", which is completely changing the meaning of the original text, so randomly replacing words is not a good idea.

A better solution than the previous approach (randomly replacing approach) is by using Named Entity Recognition (NER). NER is the task of identifying proper nouns in the given sentence and classifying them into their respective classes, for example, person, location, number, etc. We apply NER on the given sentence and mask the non-entity word(s) and predict the intensity. For example, the actual text is "How President Obama spent his final day in office". When we replace non entity words, the new text becomes "President Obama spent final day office". Even though it seems like a better approach than randomly replacing words, it also suffers from the following problem that is, the sentence is not grammatically correct and it is not fluent.

One common and main problem with the above mentioned approaches is that they operate at word level, hence fail to retain the overall meaning of the original sentence. The key challenge here for us is to come up with a system which generates a new text that retains the overall meaning of the actual text, which in turn reduces the intensity. To achieve this we use a better approach using paraphrasing which is discussed in the following section.

6.2 Proposed Approach Using Paraphrasing

Paraphrasing a sentence means, you create a new sentence that expresses the same meaning using a different choice of words. Recent advances in deep learning has achieved state of the art results in tasks like paraphrasing, text summarization, machine translation etc. We use PEGASUS model released by Google to generate paraphrase for a given input sentence.

Our proposed model flow is mentioned in the below figure

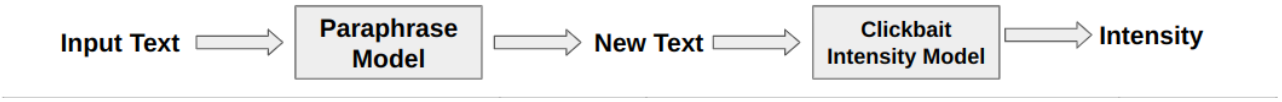


Figure 5: Proposed Approach Using Paraphrasing

We pass the input text to the paraphrasing model and it generates a new text which we use for predicting the clickbait intensity.

This approach using paraphrasing has produced decent results compared to the previously discussed approaches. The below figure contains some of the examples of the proposed approach using paraphrasing.

Input Text	Actual Intensity	Paraphrased Text	Predicted Intensity
Hillary Clinton's gut-wrenching day	0.733	Hillary Clinton had a difficult day.	0.362
How President Obama spent his final day in office	0.666	President Obama spent his last day in office.	0.301
Why you 100% need to start watching British crime drama 'Broadchurch'	1	British crime drama 'Broadchurch' is a must watch.	0.332

Figure 6: Example Outputs Using Paraphrasing Approach

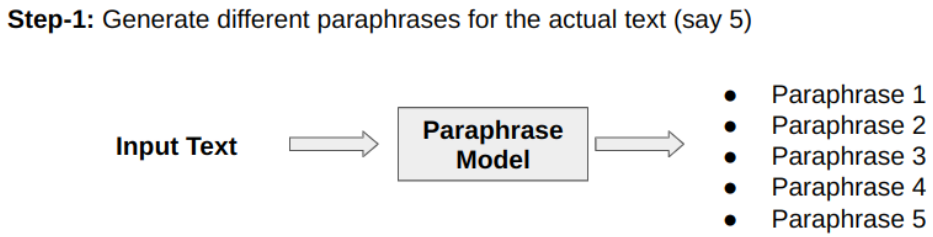
We can clearly observe from the examples in the above figure that, the paraphrased text is able to retain the meaning of the text and also reduced the intensity when compared with intensity of the actual text.

6.3 Proposed Approach Using T5 on top of Paraphrasing

Even though the paraphrasing approach seems to be a good approach, it also suffers from few problems, the main one is the quality of the paraphrase. Not all the paraphrases retain the important information of the actual text.
For example, Actual Text is ”Donald Trump senior adviser Kellyanne Conway: I’m getting death threats” and the Paraphrased Text: ”I’m getting death threats.” Here the meaning and important information of the actual text is lost, as a result the intensity increased from 0.1 to 0.66.

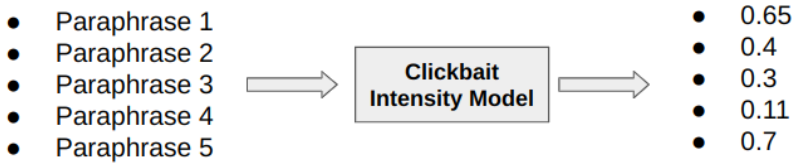
And we also observe that out of the 4339 sentences in the test set, For 1493 sentences the intensity has reduced by paraphrasing the sentence, For 2846 sentences the intensity has increased.

The key challenge is to generate a text (paraphrase) such that it always reduces the intensity. To achieve this we use T5 (Text To Text Transfer Transformer) model on top of the paraphrases. This approach works as follows...



In step1: we generate 5 different paraphrases of given text using PEGASUS model.

Step-2: Predict the Intensity of each paraphrased text



Step-3: Select the paraphrased sentences whose predicted intensity is less than the actual intensity

Say actual intensity is 0.6, then select the sentences

- Paraphrase 2
- Paraphrase 3
- Paraphrase 4

Step-4: Form a training set for each actual sentence

Actual Sentence (Source)	Paraphrased Sentence (Target)
actual sentence	Paraphrase 2
actual sentence	Paraphrase 3
actual sentence	Paraphrase 4

Step-5 : Train a T5 (Text-To-Text Transfer Transformer) model. After training, i.e during inference the model should generate a less clickbaity sentence when given a clickbaity sentence.

In step-2, We predict the Intensity of each paraphrased text.

In step-3, We select the paraphrased sentences whose predicted intensity is less than the actual intensity

In step-4, We Form a training set for each actual sentence, using the selected paraphrase sentences as targets.

In step-5, We train a T5 (Text-To-Text Transfer Transformer) model. After training, i.e during inference the model should generate a less clickbaity sentence when given a clickbaity sentence.

We have done the work till step-4 and we are experimenting with the T5 model right now.

7 Conclusion & Future Work

We have implemented the baseline paper and reproduced the results in the paper by experimenting with various regression models with different word/sentence embedding techniques. To control the intensity of the sentence by changing the style, We have explored and experimented with replacing/masking some words in the text and documented the results. And also experimented with NER based approach and discussed the problems in these approaches. To overcome these problems, we came up with a novel idea of using paraphrasing. We prove that by experiments, that the proposed paraphrasing approach has not only produced better results in reducing the intensity but also in retaining the meaning of the actual text.

We also discuss the problems involved with paraphrasing approach and to overcome that we also came up with another novel idea of using T5 model on top of the paraphrases generated by the model to ensure that the final model will always generate a less clickbaity sentence by taking more clickbaity sentence as input. We were in line with the scope document submitted earlier and We hope that the proposed approaches helps the research community to take the research on this problem statement further.

8 References/Reading Materials

References

[1] Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Gupta, and Vasudeva Varma. 2020. Predicting Clickbait Strength in Online Social Media. In Proceedings of the 28th International Conference on Computational Linguistics, pages 4835–4846, Barcelona, Spain (Online). International Committee on Computational Linguistics.

[2] Indurthi, Vijayasaradhi and Oota, Subba Reddy. 2017. Clickbait detection using word embeddings. arXiv.org

[3] M. Marreddy, S. R. Oota, L. S. Vakada, V. C. Chinni and R. Mamidi, "Clickbait Detection in Telugu: Overcoming NLP Challenges in Resource-Poor Languages using Benchmarked Techniques," 2021 International Joint Conference on Neural Networks (IJCNN), 2021, pp. 1-8, doi: 10.1109/IJCNN52387.2021.9534382.

[4] Vivek Kaushal and Kavita Vemuri. 2020. Clickbait in Hindi News Media : A Preliminary Study. In Proceedings of the 17th International Conference on Natural Language Processing (ICON), pages 85–89, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLPAI).

[5] Amjad, Maaz and Butt, Sabur and Amjad, Hamza Imam and Zhila, Alisa and Sidorov, Grigori and Gelbukh, Alexander. 2022. Overview of the Shared Task on Fake News Detection in Urdu at FIRE 2021. arXiv.org