**NATURAL LANGUAGE PROCESSING**

**CSE4022**

**PROJECT REPORT**

**PROJECT TITLE-**

Using Transformers and Recurrent Neural Network   
(Bi- LSTM and Bi-GRU) to Identify Clickbaits.

**TEAM MEMBERS- (Group 13)**

1. Soumyaraj Roy , 19BCE0200

2. Kaustubh Dwivedi, 19BCE0249

3. Rebello Nishma Avalon, 19BCE0253

4. Debangsu Sarkar, 19BCE0902

**1. ABSTRACT-**

Clickbait is a widespread problem that troubles online readers and misleads the readers to an irrelevant site. Currently, detection of clickbait on tweets remains a challenging task.

In this project, we propose to build a Clickbait Detector using Bidirectional Encoder Representations from Transformers (BERT), which can effectively identify click-baits using the latest developments in advanced training methods like BERT and Longformer and using Recurrent Neural Networks (Bi. - LSTM and Bi-GRU) with a parallel model structure.

Our model will support end-to-end training without incorporating any manual features and achieve efficient results. We will approach this task as a regression problem in our two parallel baseline models for benchmarking with previous models. The model will take the post title and the linked content as input and will output a clickbait score in the range of [0, 1] with 0 indicating non-clickbait and 1 indicating clickbait.

By training on a large twitter posts corpus with annotations of their ‘click-baitness’ on a scale of [0, 1], we expect our model to be capable of capturing clickbait patterns in the headline and the content.

**2. INTRODUCTION-**

Clickbait refers to a certain kind of headline that attracts people to click but gives something uncorrelated in that link. The motivation behind clickbaiting is to boost site traffic (and therefore, advertisement revenue) by exploiting the curious nature of human readers. The clickbait technique works by dangling a hyperlink with enticing headlines to lure people into clicking; and then redirect them to the publishers’ own websites which are uncorrelated to the headline.

The discrepancy between the headline and the destination content wastes online readers significant amount of time on contents of which they have no interest. To address this problem, we propose Click-BERT\*: (Clickbait Detector with Bidirectional Encoder Representations from Transformers), which could effectively identify clickbaits utilizing state-of-the-arrt pre-training methods and self-attentive network. We approach this task as a regression problem in our two parallel baseline models for benchmarking with previous models. The model takes the post title and the linked content as input and will output a clickbait score in the range of [0, 1] with 0 indicating non-clickbait and 1 indicating clickbait. By training on a large twitter posts corpus with annotations of their ‘clickbaitness’ on a scale of [0, 1], we expect our model to be capable of capturing clickbait patterns in the headline and the content.

Main challenges of the clickbait detection problems lies-

in how well our model capture the meaning and correlation of the input headline/content (which differs greatly in length) and how properly the followed analysis gets performed. And our main contributions include:

1. We applies advanced pre-trained models BERT and Longformer to extract sequence (headline/content) embedding to form a better understanding of the headline and the content.

2. We proposes a parallel model structure to integrate both prediction of whether the headline is luring people to click and prediction of whether the headline is related with the content into final judgement.

**3. LITERATURE REVIEW**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S. No.** | **Paper Title & Details** | **Method/Algorithm** | **Challenges** | **Observations** |
| 1. | Clickbait Detection in YouTube Videos.  Authors- Ruchira Gothankar, Fabio Di Troia, Mark Stamp.  Year- 2021 | The authors performed clickbait detection experiments are based on a set of labeled videos. The problem is formulated as a binary classification problem where for each video a machine learning algorithm classifies it is clickbait or non-clickbait. The information from multiple sources (e.g., title, description, comments) are combined and fed to the classification model. The performance is evaluated and analyzed by multiple measures, specifically, precision, recall and the F-score. BERT, Word2Vec, and DistilBERT were used for word embeddings | They confirmed that the accuracy of the models could be increased by adding more features. For future work, more features have to be included and also DocToVec embeddings could be considered. | Multiple classification techniques were considered, including logistic regression, random forest, and MLP, and we employed Word2Vec, BERT, and DistilBERT as language models.  The best accuracy was achieved using an MLP classifier based on BERT embeddings which is 94.5 %, but a the more lightweight DistilBERT performed almost same. |
| 2. | exBAKE: Automatic Fake News Detection Model Based on Bidirectional Encoder Representations from Transformers (BERT).  Authors- Heejung Jwa , Dongsuk Oh, Kinam Park, Jang Mook Kang and Heuiseok Lim. | In this paper, the authors focus on data-driven automatic fake news detection methods. First they apply the Bidirectional Encoder Representations from Transformers model (BERT) model to detect fake news by analyzing the relationship between the headline and the body text of news.  To further improve performance, additional news data are gathered and used to pre-train this model. | They experiment with various cases of fake news detection tasks using the pre-trained BERT model proposed in this study. They only analyzed the relationship between the headline and the body text of an article. But, Further experimentation is needed to apply data from other fake news detection tasks to BERT model, which will use additional news data in the pre-training phase. | They determine that the deep-contextualizing nature of BERT is best suited for this task and improves the 0.14 F-score over older state-of-the-art models |
| 3. | Clickbait Headline Detection in Indonesian News Sites using Multilingual Bidirectional Encoder Representations from Transformers (M-BERT).  Authors-Muhammad Noor Fakhruzzaman , Sa’idah Zahrotul Jannah, Ratih Ardiati Ningrum, Indah Fahmiyah.  Year 2021. | This study contributes to show that Multilingual BERT, a state-of-the-art model is able to classify Indonesian clickbait headlines.  By using BERT, the whole model looks simplified, using only a BERT layer and a hidden standard dense layer, finally topped with a sigmoid activated neuron, the classifier worked remarkably well with an average accuracy of 92%. | A further study is needed to evaluate the model versatility. Moreover, training a Neural Network with M-BERT took a lot of computing resource. | If efficiency is the priority, XGBoost can perform moderately well (80% avg.).  The additional evaluation shows average accuracy of 0.83, precision of 0.82, recall of 0.83, and f1-score of 0.83 |
| 4. | Stop clickbait: Detecting and preventing clickbaits in online news media. Chakraborty, A., Paranjape, B., Kakarla, S. and Ganguly, N. IEEE | 1. Authors did a definite phonetic investigation on the 15, 000 features both in the misleading content and non-misleading content, utilizing the Stanford CoreNLP tool. They examined how semantic and syntactic subtleties which are explicit to misleading content sources like Sentence Structure, Stop words, Determiners, Word N Grams, POS Tags.  At last they characterized utilizing Feature selections like Word patterns, clickbait content language, N Gram features. At last they implement the classifier through a Browser extension. | 1. Manually identify the clickbait articles from Clickbait-y sites, and to avoid false negatives we need multiple opinions as an article is a clickbait or not is a subjective opinion - We need to take majority vote.  2. Need to manually compiled a list of most commonly used bait phrases.  3. One issue about earlier works is that they either work on a single domain, or the fixed ruleset does not capture the nuances employed across different websites | 1. Conventional non-clickbait headlines contain much larger proportion of proper nouns.  2. Clickbait headlines contain more adverbs and determiners  There’s a lot of extreme positive or negative words in clickbait sites, called Hyperboles.  3. Informal Punctuations. |
| 5. | Bert: Pre-training of deep bidirectional transformers for language understanding, J., Chang, M.W., Lee, K. and Toutanova, K., 2018 | There are two steps in the BERT framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For finetuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters | 1. Deployment of BERT models in dynamic commercial environments often yields poor results. This is because commercial environments are usually dynamic, and contain continuous domain shifts (e.g. new themes, new vocabulary or new writing styles) between inference and training data, thus the challenge of dealing with dynamic cross-domain setups in which there is no labeled target-domain data, still remains.  2. BERT can be used only for answering questions from very short paragraphs and a lot of key issues need to be addressed. NLP as a general task is way too complex and has many more meanings and subtleties. BERT solves only a part of it but is certainly going to change entity Recognition models soon. | 1. A distinctive feature of BERT is its unified architecture across different tasks. There is minimal difference between the pre-trained architecture and the final downstream architecture.  2. Recent empirical improvements due to transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems. In particular, these results enable even low-resource tasks to benefit from deep unidirectional architectures.  3. To improve the training procedure, RoBERTa removes the Next Sentence Prediction (NSP) task from BERT’s pre-training and introduces dynamic masking so that the masked token changes during the training epochs. It was also trained on an order of magnitude more data than BERT, for a longer amount of time. |
| 6. | Detecting and Categorization of Click Baits, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NTASU – 2020 (Volume 09 – Issue 03). Sainath Patil, Mayur Koul, Harikrishan Chauhan, Prachi Patil, 2021. | The authors propose a completely unique approach considering all information found during a social media post.  We train a bidirectional Long Short Term Memory(LSTM) with an attention mechanism to learn the extent to which a word contributes to the posts clickbait score in a differential manner.  Sequence Followed: Data Collection, Word Embedding, Developing the Deep Learning Models. | 1. The aforementioned Attention mechanism wasn’t implemented into the paper, leading to us thinking that it might be hard to do so.  2. However, work done specifically for Twitter had to be expanded since clickbait was available throughout the Internet, and not just social networks.  3. Again, the definition of what a clickbait is and what isn’t is vague and is an issue that needs to be discussed before approaching the required problem. | 1. The primary instance of detecting clickbait across social media can be traced , hand-crafting linguistic features, including a reference dictionary of clickbait phrases, over a data set of crowdsourced tweets.  2. The features need to be more nuanced to avoid flagging non-clickbait articles. |
| 7. | Unified Medical Language System resources improve  sieve-based generation and BERT–based ranking  for concept normalization  2020  Dongfang Xu ,1 Manoj Gopale,2 Jiacheng Zhang ,3 Kris Brown,4 Edmon Begoli,4  and Steven Bethard | Authors designed a sieve-based system over the training data, Unified Medical Language  System (UMLS) preferred terms, and UMLS synonyms to generate a list of possible concepts for each mention.They  then design a list-wise classifier based on the BERT neural network to rank the candidate concepts, integrating UMLS semantic types through a regularizer. | A major challenge is the unseen mentions and concepts: 50.76%  (29.85%) of test mentions (concepts) were not seen in the training data. Systems that memorize the training data or rely on it to determine the space of output concepts will thus perform poorly.  Also Lexical and grammatical variations are pervasive in such text, posing key challenges for data interoperability and the development of natural  language processing (NLP) techniques. | Analysis of the model shows that prioritizing UMLS preferred terms yields better performance, that  the UMLS semantic type regularize results in qualitatively better concept predictions, and that the model performs well even on concepts not seen during training. |
| 8. | A transformer based approach for fighting  COVID-19 fake news  2021  S.M. Sadiq-Ur-Rahman Shifath1, Mohammad Faiyaz Khan2, and Md. Saiful Islam3 | Authors performed experiments primarily on traditional language models such as Bidirectional LSTM(Bi-LSTM) with attention, 1 dimensional CNN(1D-CNN), Hierarchical Attention Networks(HAN), Recurrent convolutional Neural Networks(RCNN),  and Multichannel CNN with Attention(AMCNN) on the competition dataset. We also experiment with transformer-based pre-trained models like BERT and RoBERTa. | Authors tested different hyper-parameters like the number of layers, number of units in a layer, learning rate, weight decay, dropouts, normalization, etc. within a feasible range which was a very difficult job, also they faced resource limitation for experimenting with larger models. | Authors have presented our overall workflow for the fake news detection task. They have conducted a number of experiments and provided a comprehensive solution based on modified transformers with additional layers and  An ensemble classifier. |
| 9. | A Comparative Analysis Of Classifiers Used For Detection of Clickbait In News Headlines.  Aaryaman Bajaj , Himanshi Nimesh , Raghav Sareen , Dinesh Kumar Vishwakarma.  Proceedings of the Fifth International Conference on Intelligent Computing and Control Systems (ICICCS 2021). | The authors compare the performance of different classifiers in detecting the clickbait headlines of news articles by performing the extraction of new features from a multi-source dataset.  Random Forest classifier yields a better accuracy than Naïve Bayes and Logistic Regression models in identifying headlines disseminating misleading information. | New clickbait  formats are added each year, and many new methods can be  incorporated into the model, to further improve accuracy.  There is a high degree of similarity between the evaluation performance of the proposed model and other existing models. | They obtained scores on applying the various methods. We got the best results from Random Forests.  Random Forest accuracy 0.891. |
| 10. | BERT, XLNet or RoBERTa: The Best Transfer Learning Model to Detect Clickbaits  2021  Authors : PRABODA RAJAPAKSHA , (Student Member, IEEE), REZA FARAHBAKHSH , (Member, IEEE), AND NOEL CRESPI, (Member, IEEE) | :Based on the author’s knowledge, this is the first attempt to adapt Transfer Learning to classify Clickbaits in social media. In this work they have fine-tuned BERT, XLNet and RoBERTa models by integrating novel configuration changes into their default architectures such as model expansion, pruning and data augmentation strategies.  Authors have used three fine-tuning approaches, namely; model generalization, expansion and pruning. The analysis has shown that pruning performed better than model expansion. In the expansion, the best result is achieved when we generated the output from hidden states without directly using pooled output (the default model output). | There is no significant performance improvement when each model expanded by adding an extra RNN layer(s).  Apart from that, we experimented with another labelled clickbait dataset (Kaggle clickbait challenge) to explore the performance of our fine-tuned models under different scenarios. | The results shown that, RoBERTa outperformed the BERT and XLNet in many experiments mainly when we fine-tuned the model using hidden outputs to generate the output vector without using the pooled output and adding a non-linear layer at the end.  This model architecture is considered to be the best performed model in our experiments. |

**4. PROBLEM STATEMENT-**

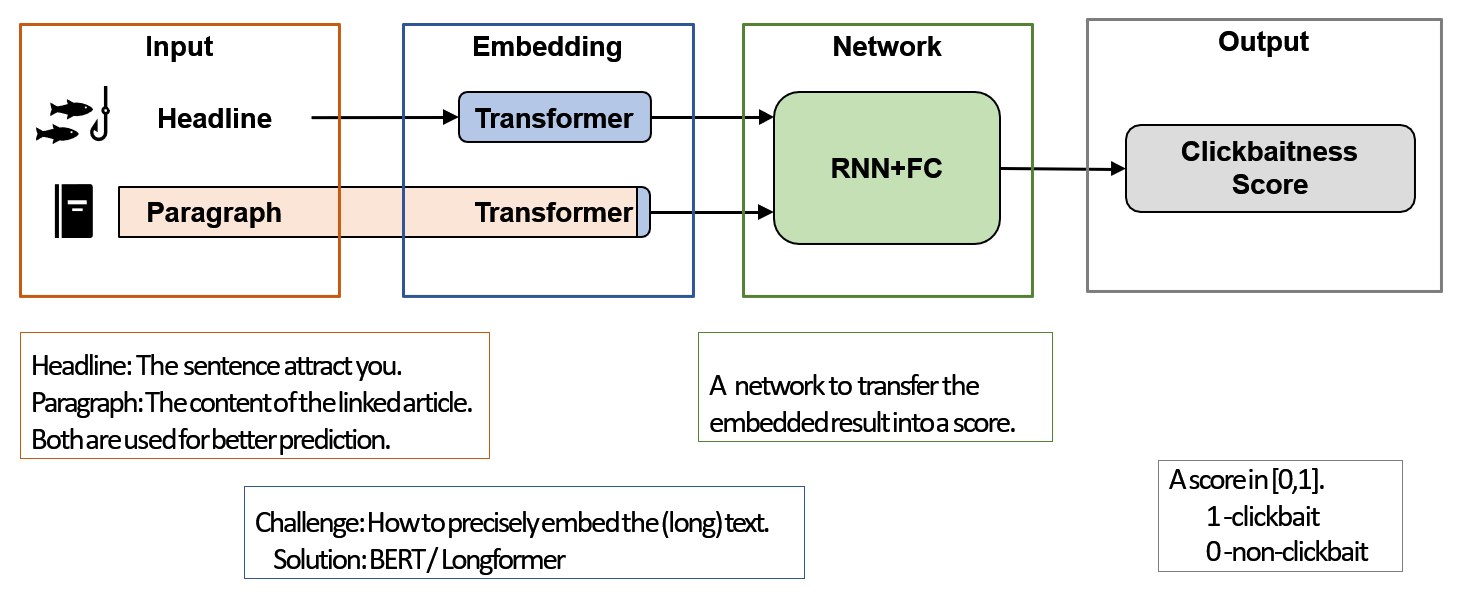
Clickbait refers to a certain kind of headline that attracts people to click but gives something uncorrelated in that Link. The motivation behind click-baiting is to boost site traffic (and therefore, advertisement revenue) by exploiting the curious nature of human readers.

This technique works by dangling a hyperlink with enticing headlines to lure people into clicking; and then redirect them to the publishers’ own websites which are uncorrelated to the headline.

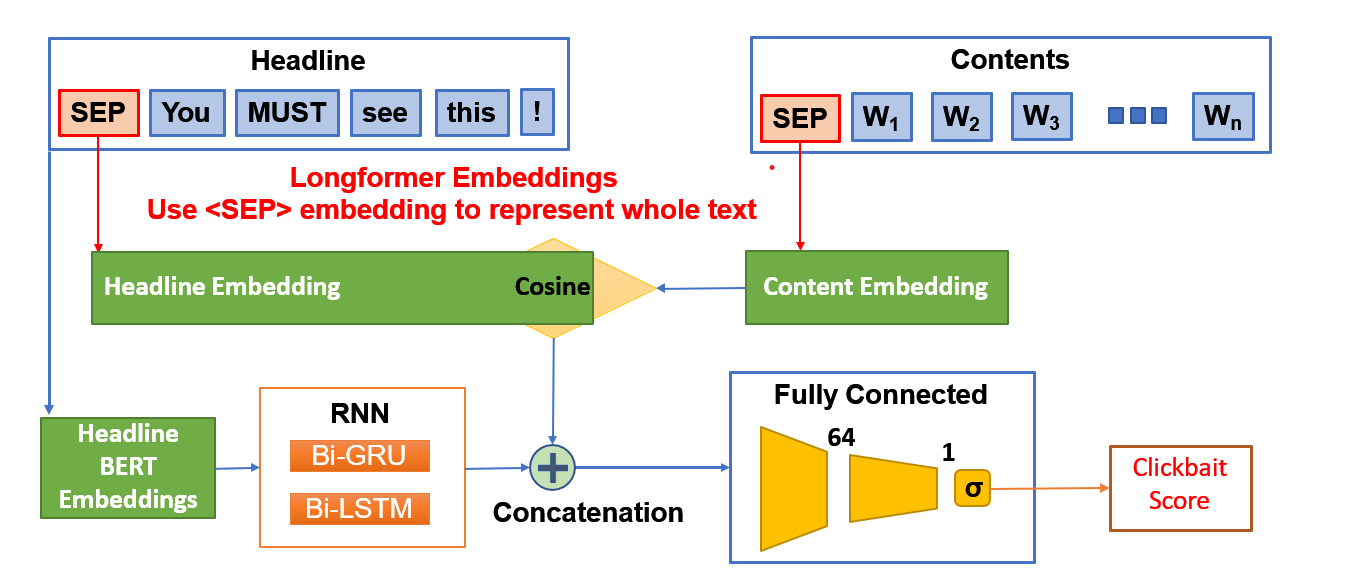
The discrepancy between the headline and the destination content wastes online readers significant amount of time on contents of which they have no interest.

To address this problem, we propose a Clickbait Detector with Bidirectional Encoder Representations from transformers which could effectively identify clickbait.

**4.1 ARCHITECTURE DIAGRAM**



**4.2 FLOW DIAGRAM**



**4.3 PSEUDO CODE**

**Step 1 : Read the data**

- train\_file = instances.jsonl

- test\_file = truth.jsonl

- df\_train, df\_test = read train, read\_test

- size = train.shape[0]

**Step 2 : Create the dictionary**

- truth\_id, truth\_mean = list test(id), list test(mean)

- truth\_dict = truth\_id[i]:truth\_mean[i] for all I

- train\_id, train\_post, train\_text = list(id, heading, content)

- creating corpus = join(id, post , text) with truth\_dict

**Step 3 : Cleaning of data (discard tweets with 0.3 < score < 0.7)**

- initial\_length = size = 19538

- cleaned\_web17 = new List [ ]

- iterating from i = 0 to size and if condition match append in cleaned\_web17

- condition = 0.3 < mean score < 0.7

- new\_final\_length = 12963

**Step 4 : Bert Embedding**

Download BERT

bert\_tokenizer = BertTokenizer.from\_pretrained("bert-base- uncased")

bert\_model = BertModel.from\_pretrained("bert-base-uncased")

save BertTokenizer, BertModel.

now encode text into sequence of IDs.

encode1 = torch.tensor(bert\_tokenizer.encode(web17.corpus[0][0]))

print(encode1.shape)

**Step 5 : Data Profiling**

- extract data; title\_all,content\_all,score\_all = [data in web17.corpus]

-title\_all\_token = bert\_tokenizer()

- Print the mean no. Of token, ID etc.

-Average # of tokens = 17.628058143105743

- content\_all\_token = bert\_tokenizer()

- Print the mean no. Of token, ID etc.

-Average # of tokens = 791.2599037772546

**Step 6 : Extract embeddings & divide train/val/test set**

-title\_all\_tokenized = bert\_tokenizer()

-Save it as a Py Torch file

-Give train\_size and val\_size

-shape gives batch size of 800

**Step 7 : Process by patches and combine**

-import torch and gc

-extract\_size=800 // One batch

-for loop form I =0 to num\_data//800

-outputs = bert\_model()

-check the shape

-save the title\_all\_embed as per the shape

-save the last 501 content and title separately (from 19200 to 19538)

**Step 8 : Loading Data**

- importing TensorDataset and DataLoader

- Xt\_all = torch.load ('titles\_all.pt')

- yt\_all = torch.load ('scores.pt')

- diving training, validation size = 10000, 2000

- test size = total - training - val = 963

- batch\_size = 64

- using TensorDataset on train, val and test

- using DataLoader on train, val and test

**Step 9 : Defining LSTM Model Architecture**

- importing torch.nn

- class LSTM inherit base class nn.Module

- \_init\_ constructor : batch size , num\_tokens, embed\_dim, hidden\_dim, n\_layers, dropout

- self.LSTM = embed\_dim, hidden\_dim, n\_layers, batch\_first=True, dropout=dropout, bidirectional=True

- defining two fully-connected layers :

self.fc1=nn.Linear(2\*hidden\_dim, 64)

self.fc2=nn.Linear(64, 1)

- defining forward function LSTM

- lstm\_out, hidden = self.lstm(x.unsqueeze(1), hidden)

flat = lstm\_out.squeeze()

out1 = self.fc1(flat)

out2 = self.fc2(torch.relu(out1))

out = torch.sigmoid(out2)

- iterating over the parameters

- defining init\_hidden(batch\_size) :

hidden = (weight.new(self.n\_layers\*2, batch\_size, self.hidden\_dim).zero\_()

weight.new(self.n\_layers\*2, batch\_size, self.hidden\_dim).zero\_()

- initializing the weights using xavier uniform (normal)

- torch.nn.init.xavier\_uniform\_(m.weight)

m.bias.data.fill\_(0.0)

**Step 10 : Hyper-parameters Initialization**

- hidden\_dim = 10

- dropout = 0.2

- optimizer = Adam Optimizer

- learning rate = 3e-4

- n\_layers = 2

- importing learning rate scheduler

- hyper-parameters of lr\_scheduler :

optimizer, 'min', factor=0.25, patience=0, threshold=0.05,min\_lr=3e-5, verbose=True

**Step 11 : Training and Testing**

- define training function with parameters as :

train\_dataloader, y\_truth, model, loss\_fn, optimizer, mute = False

- y\_pred\_train = []

- enumerate over train\_dataloader

- for batch, (X, y) in enumerate(train\_dataloader):

- Compute prediction error

pred, hidden = model(X, hidden)

y\_pred\_train.extend(pred.squeeze().cpu())

loss = loss\_fn(pred.squeeze(), y)

- Backpropagation

loss.backward( )

optimizer.step( )

- define testing function with parameters same as training function and mode

- mode = 0: validation when training (lr\_scheduler)

mode = 1: validation

mode = 2: test

- evaluating the model using four metrices

Loss, Accuracy, F1Score, Pearson Coefficient

**Step 12 : Running the model for 5 epochs**

- epochs = 5

- model.train()

- best\_val\_performance = 1.0

- train(train\_dataloader, yt\_all[:train\_size], model, loss\_fn, optimizer)

- val\_performance = test(val\_dataloader, yt\_all[train\_size:train\_size+val\_size], model, loss\_fn, lr\_scheduler)

**Step 13 : Loss Function Optimizer and Accuracy**

- hidden\_dim = 10 # num of tokens is typically 20

- \_ , num\_tokens, embed\_dim = Xt\_all.shape

- dropout = 0.2

- Using MSELoss as a loss function

loss\_fn = nn.MSELoss()

- Using Adam Optimizer with learning rate 3e-4

optimizer = torch.optim.Adam (model.parameters(), lr=3e-4)

- using learning rate scheduler

lr\_scheduler = ReduceLROnPlateau(optimizer, 'min', factor=0.25, patience=0, threshold=0.05, min\_lr=3e-5, verbose=True)

**Step 14 : Testing on validation and test data**

- \_ = test(val\_dataloader, yt\_all[train\_size: train\_size+val\_size], model, loss\_fn, lr\_scheduler, mode = 1)

- \_ = test(test\_dataloader, yt\_all[train\_size+val\_size:], model, loss\_fn, lr\_scheduler, mode = 2)

**5. EXPERIMENTS AND RESULTS**

**5.1 DATASET**

Webis-Clickbait-17 Dataset (19538 Tweets)

Link-

<https://zenodo.org/record/5530410#.YjIb5XpBxhF>

On the Webis 17 dataset, we conduct experiments. contains a total of 38,517 tweets from 27 major US news organisations. They'd been categorised according to how clickbaity they were. The title and content of the article were included in these tweets, as well as supplemental information like the target description, target keywords, and related photos. The data has previously been divided into two sets: a train set (19,538 posts, 4761 of which are clickbaits and 14,777 non-clickbaits) and a test set (19,538 posts, 4761 of which are non-clickbaits) (18,979 posts).

Five Amazon Mechanical Turk annotators rated each post on a 4-point scale [not click baiting (0.0), mildly click baiting (0.33), significantly click baiting (0.66), and heavily click baiting (1.0). Most annotators estimate a total of 9,276 postings to be clickbait.

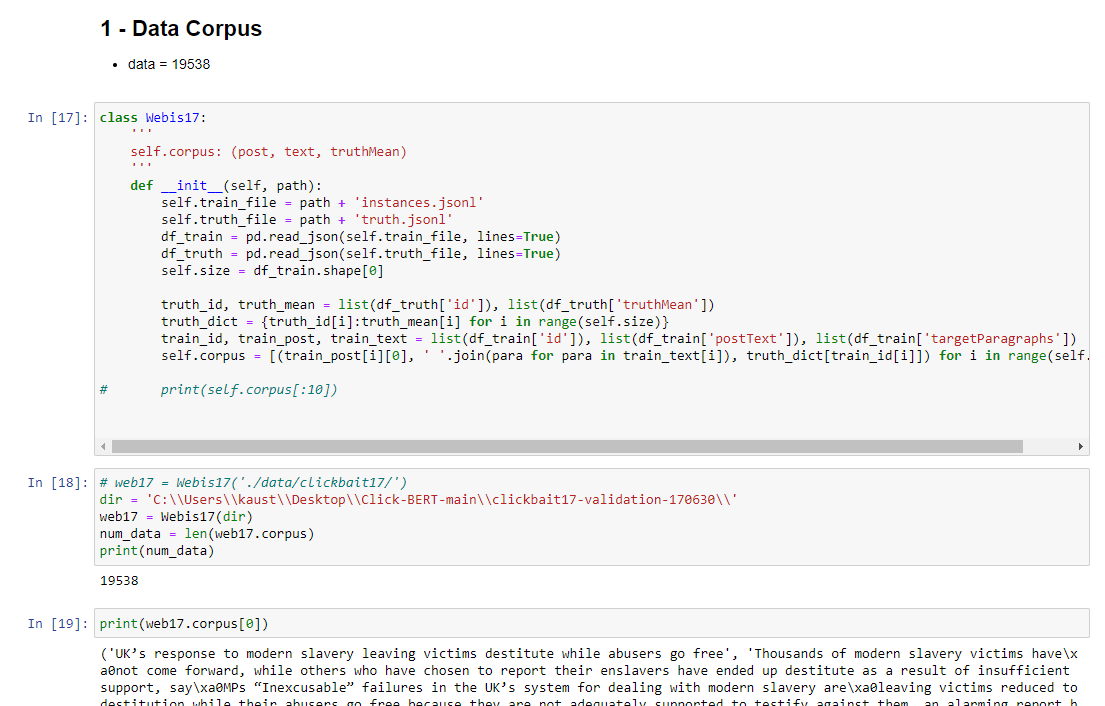
**5.1.1 METHODOLOGY**

ERT and Longformer with Parallel Structure-

On top of the two baseline models, we construct our final model. The first baseline model depicts the relationship between the headline and the text. The second baseline model focuses on the headline interpretation. Both factors should be considered when determining if a tweet is a clickbait or not, according to the clickbait definition stated in our introduction section. As a result, we set up the two models in a parallel structure and combine their outputs. We effectively create an ensemble of the two baseline models in this way. Finally, to calculate the clickbait score, the outputs are mapped via a fully linked layer that is activated by the Sigmoid function. To get around the BERT model's input length limitation, we modified the encoding layer in the second baseline model to Longformer, which works well with long texts. Because the performance difference between Bi-LSTM and Bi-GRU is minor, we only tried both Bi-LSTM and Bi-GRU in the recurrent neural network block.

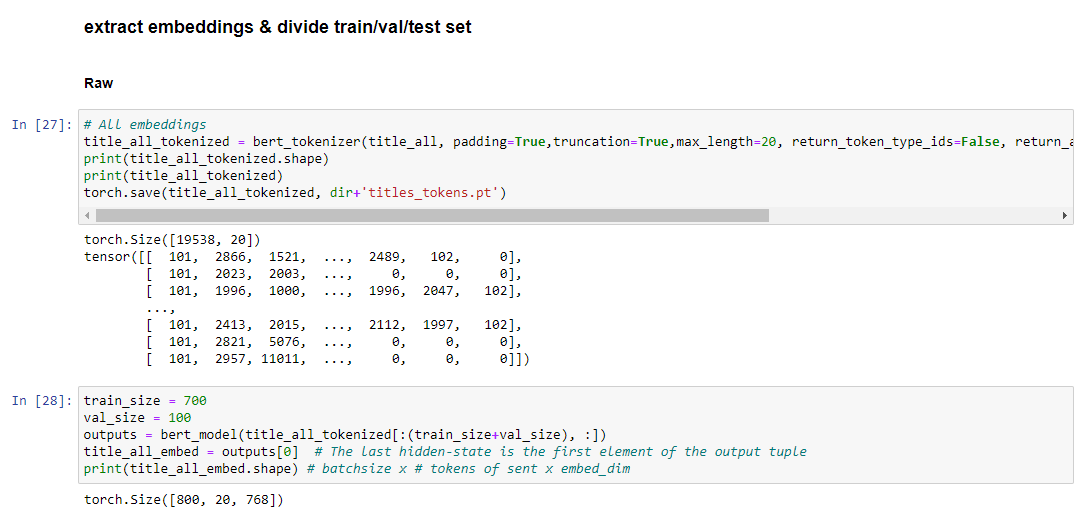
**5.1.2 OUTPUT**

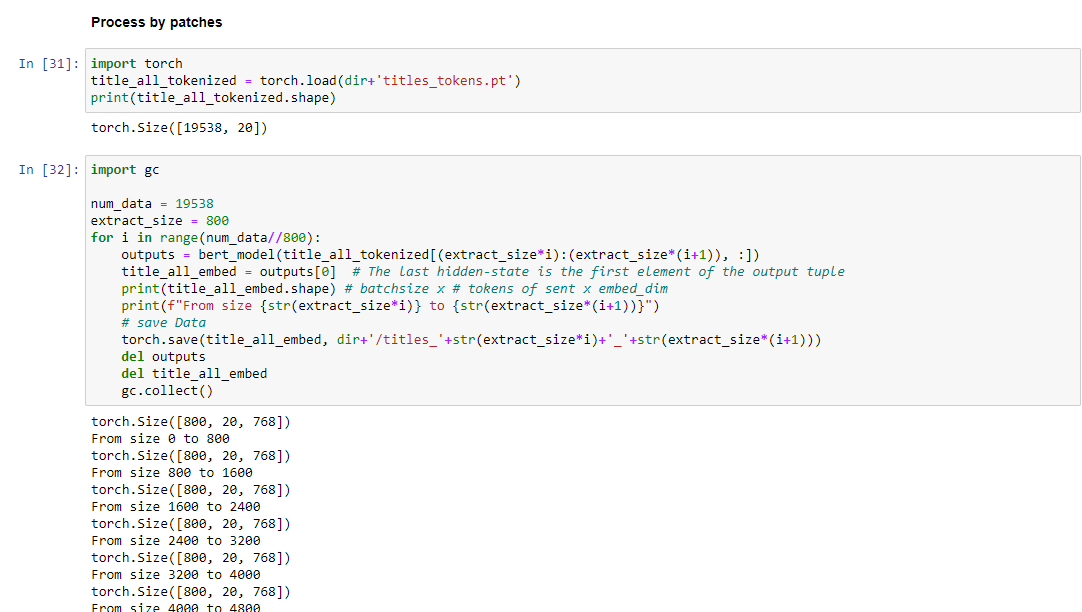




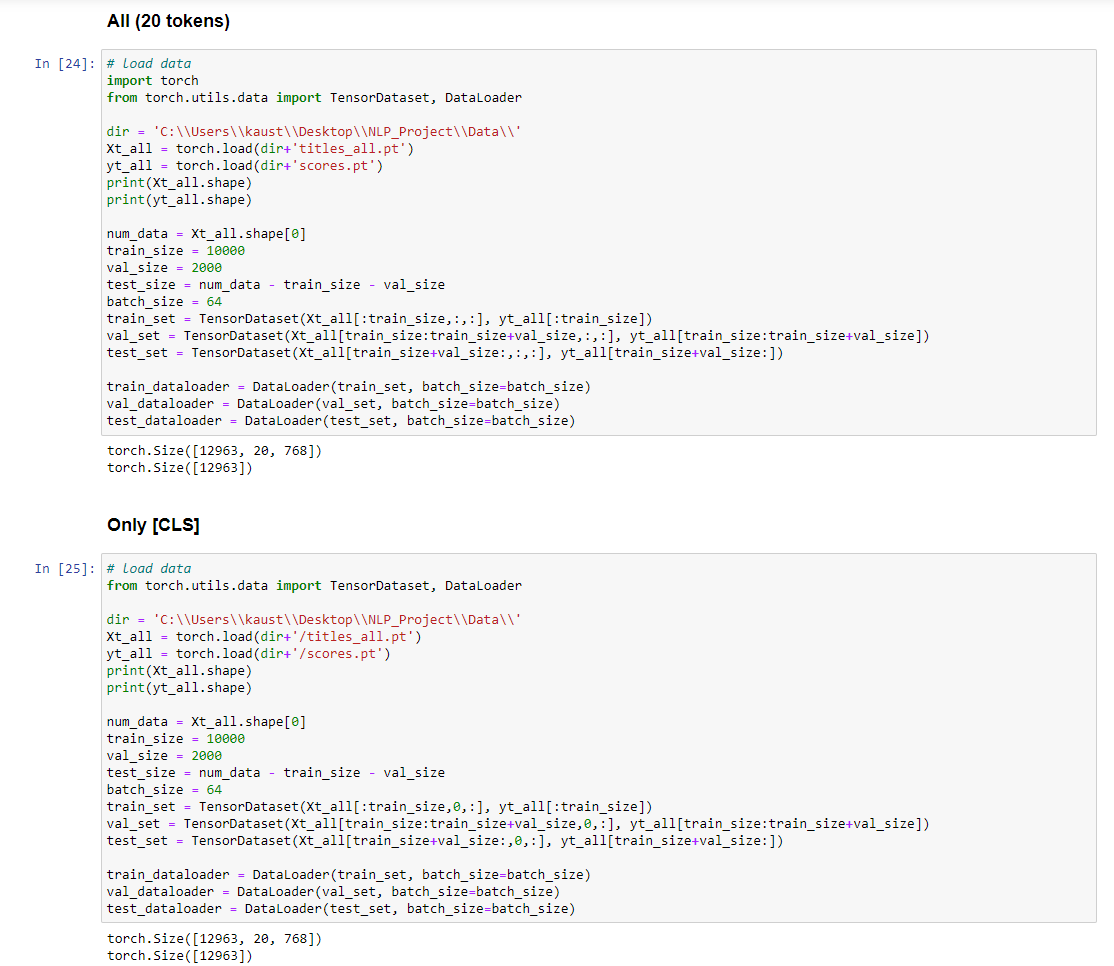


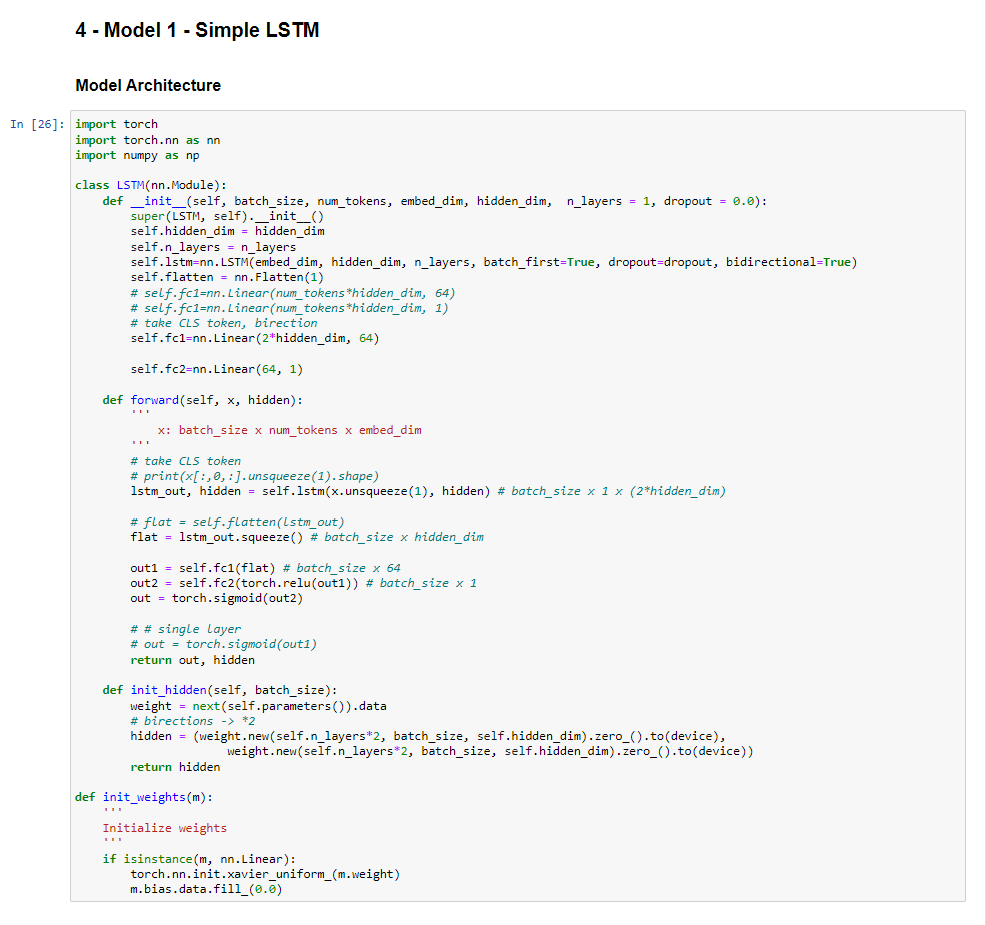


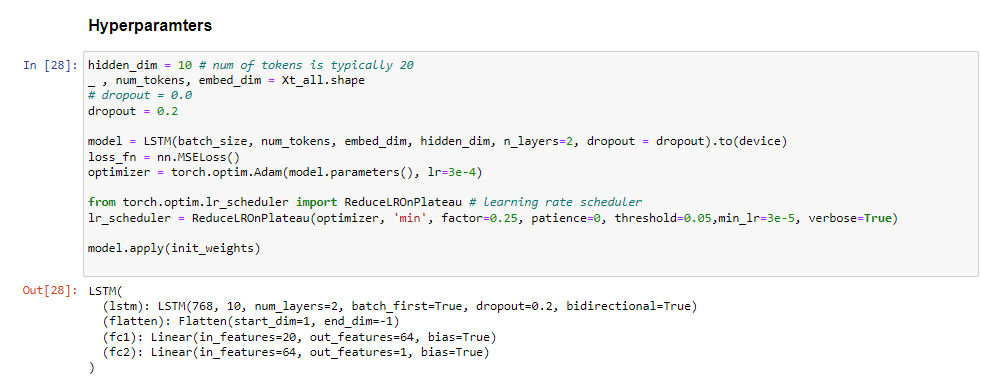




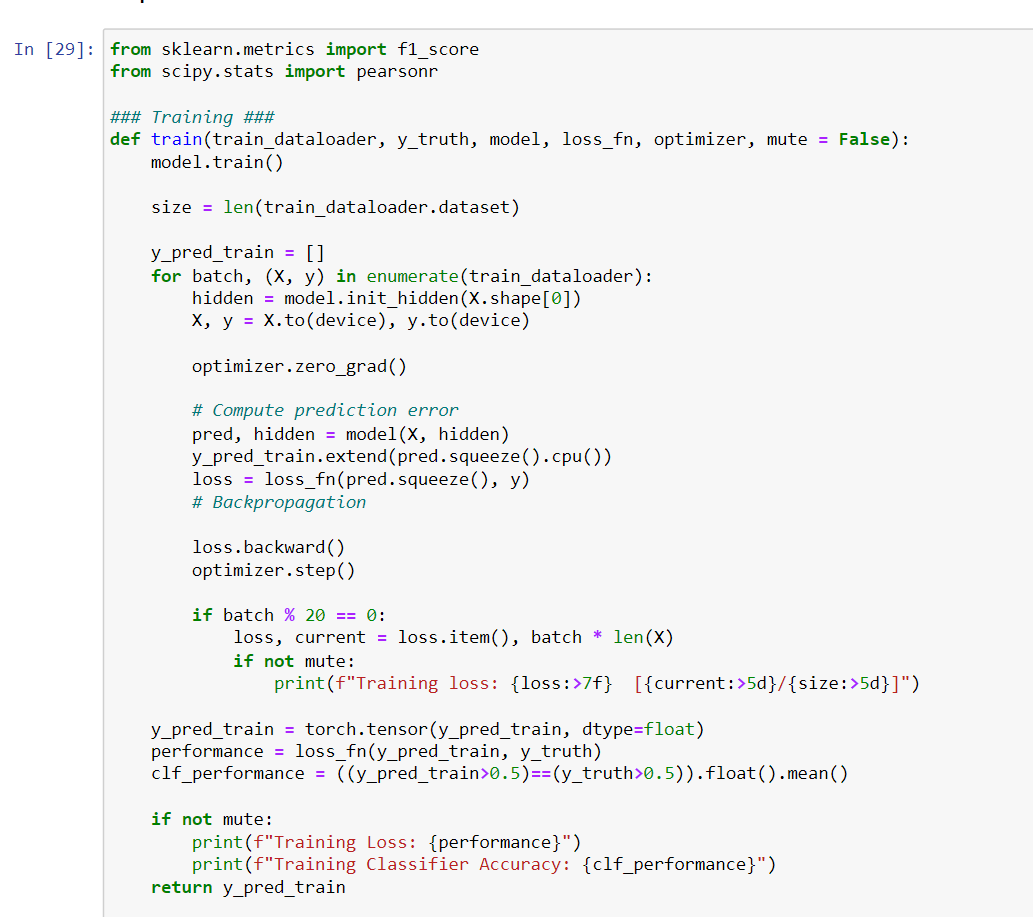




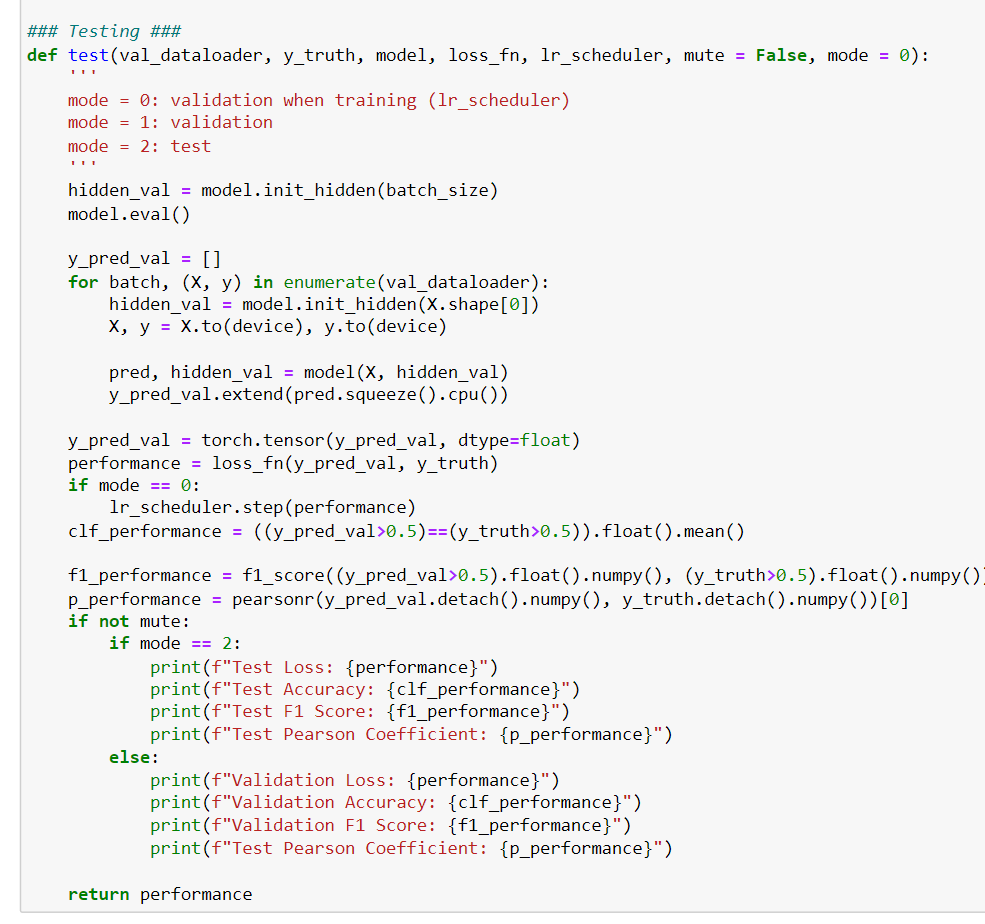


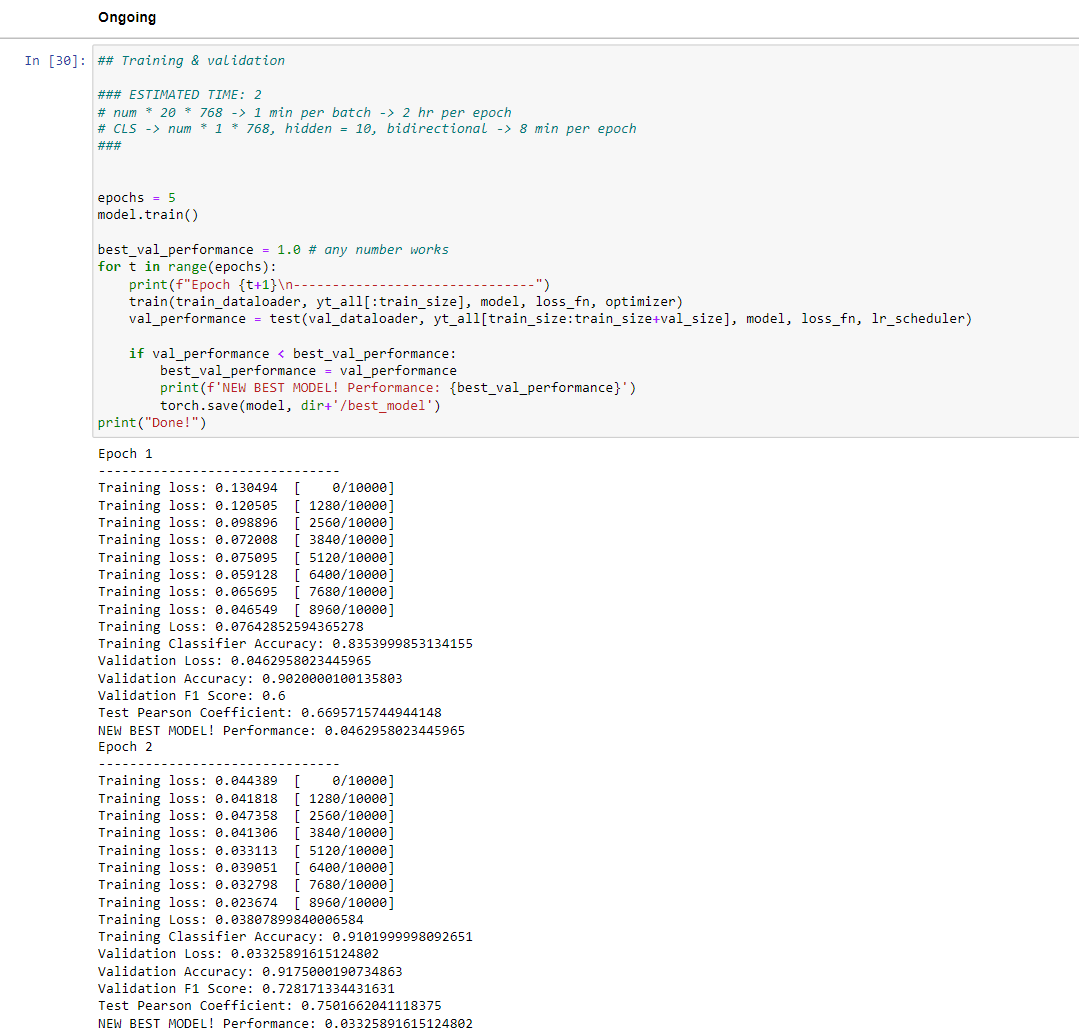


**Training**



**Testing**







**6. CONCLUSION**

In this project, we proposed to build a Clickbait Detector with Bidirectional Encoder Representations from Transformers. It could be trained from end-to-end without involving any manual feature engineering. It will effectively identify clickbaits and non-clickbaits with high accuracy.

**7. REFERENCES**

[1] Clickbait Detection in YouTube Videos.

Authors- Ruchira Gothankar, Fabio Di Troia, Mark Stamp.

[2] exBAKE: Automatic Fake News Detection Model Based on Bidirectional Encoder Representations from Transformers (BERT). Authors- Heejung Jwa , Dongsuk Oh, Kinam Park, Jang Mook Kang and Heuiseok Lim.

[3] Clickbait Headline Detection in Indonesian News Sites using Multilingual Bidirectional Encoder Representations from Transformers (M-BERT). Authors-Muhammad Noor Fakhruzzaman , Sa’idah Zahrotul Jannah, Ratih Ardiati Ningrum, Indah Fahmiyah. 2021.

[4] Stop clickbait: Detecting and preventing clickbaits in online news media. Chakraborty, A., Paranjape, B., Kakarla, S. and Ganguly, N. IEEE.

[5] Bert: Pre-training of deep bidirectional transformers for language understanding, J., Chang, M.W., Lee, K. and Toutanova, K., 2018

[6] Detecting and Categorization of Click Baits, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NTASU – 2020 (Volume 09 – Issue 03). Sainath Patil, Mayur Koul, Harikrishan Chauhan, Prachi Patil, 2021.

[7] Unified Medical Language System resources improve

sieve-based generation and BERT–based ranking

for concept normalization, 2020, Dongfang Xu ,1 Manoj Gopale,2 Jiacheng Zhang ,3 Kris Brown,4 Edmon Begoli,4

and Steven Bethard.

[8] A transformer based approach for fighting

COVID-19 fake news, 2021 , S.M. Sadiq-Ur-Rahman Shifath1, Mohammad Faiyaz Khan2, and Md. Saiful Islam3.

[9] A Comparative Analysis Of Classifiers Used For Detection of Clickbait In News Headlines. Aaryaman Bajaj , Himanshi Nimesh , Raghav Sareen , Dinesh Kumar Vishwakarma.

[10] BERT, XLNet or RoBERTa: The Best Transfer Learning Model to Detect Clickbaits

2021. Authors : PRABODA RAJAPAKSHA , (Student Member, IEEE), REZA FARAHBAKHSH , (Member, IEEE), AND NOEL CRESPI, (Member, IEEE) .

[11] Paula Branco, Lu´ıs Torgo, and Rita P. Ribeiro. SMOGN: a pre-processing approach for imbalanced regression. In First International Workshop on Learning with Imbalanced Domains: Theory and Applications, LIDTA@PKDD/ECML 2017, 22 September 2017, Skopje, Macedonia, volume 74 of Proceedings of Machine Learning Research, pages 36–50. PMLR, 2017.

[12] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. Stop clickbait: Detecting and preventing clickbaits in online news media. CoRR, abs/1610.09786, 2016.

[13] Martin Potthast, Tim Gollub, Matthias Hagen, and Benno Stein. The clickbait challenge 2017: Towards a regression model for clickbait strength.

[14] Philippe Thomas. Clickbait identification using neural networks. CoRR, abs/1710.08721, 2017.

[15]A Comparative Analysis Of Classifiers Used For Detection of Clickbait In News Headlines. Aaryaman Bajaj, Himanshi Nimesh; Raghav Sareen, Dinesh Kumar Vishwakarma.