IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 30, 2021, accepted October 23, 2021, date of publication November 16, 2021,
date of current version November 24, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3128742

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

PRABODA RAJAPAKSHA, (Student Member, IEEE), REZA FARAHBAKHSH, (Member, IEEE), AND NOEL CRESPI, (Member, IEEE)
CNRS Lab UMR5157, Institut Polytechnique de Paris, 91764 Palaiseau, France
Corresponding author: Praboda Rajapaksha (praboda.rajapaksha@telecom-sudparis.eu)

ABSTRACT Clickbait can be a spam or an advert which more often provides a link to commercial website and it can also be a headline to news media website which makes money from page views by providing eye-catchy headlines with deceptive news. This paper focuses on the latter definition in order to identify news clickbaits that are published in Twitter. The aim of this work is to use recent Transfer Learning models to detect news clickbaits by adding various configuration changes to the existing models. Based on the author's knowledge, this is the first attempt to adapt Transfer Learning to classify Clickbaits in social media. In this work we 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. Webis Clickbait dataset was used to train these models and the best performed model at the Webit Clickbait competition 2017 was considered as our benchmark. The analyses in this work are mainly focused on eight different scenarios after applying several fine-tuning approaches and model configuration changes to the default Transfer Learning models. The results shown that, our modified Transfer Learning approaches outperformed the considered benchmark. In our experiments, the best performed Transfer Learning model was RoBERTa with the integration of an additional non-linear layer with the hidden output tensor. this configuration has achieved 19.12% more accuracy in compared to the benchmark model for the binary classification. 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.

INDEX TERMS Clickbait, fake news, transfer learning, BERT, RoBERTa, XLNet, Twitter, news clickbaits, deep learning.

I. INTRODUCTION

The extensive spread of news in social media is a double-edged sword. On one hand, social media platforms provide easy access and fast news dissemination, but on the other these platforms permit to spread fake news within a short period of time. The notoriety of fake news can be partially attributed to Clickbait. Clickbait is a form of false advertisement intends to attract readers' attention via thumbnail links that lead them to read, view or listen to the content available in the respective web pages. Clickbait employs an appealing headline (that is deceptive, misleading or sensationalised) aiming to attract many readers and encourage them to click on the provided link. Most popular Clickbaits are in the form of spam and adverts that are used to redirect users to the commercial website pages. Another common type of clickbaits are in the form of news headlines intended to increase number of page views in order to increase their revenue. In this scenario, a reader can easily be a victim as she assume the news source as a legitimate news, but in reality those news can be deceptive, sensationalised, misleading, unverified and provide irresponsible information [1]. Hence, news media in social media often use clickbaits to generate more value than the actual content hidden behind the headline in order to obtain more clicks. Since news media behave as content providers and consumers for exchanging news content among them [2], [3]. A part of these news content can be categorised as misinformation that are deliberately created to mislead and hence, require much more attention due to security and information reliability concerns. Clickbaits become popular

The associate editor coordinating the review of this manuscript and approving it for publication was Christos Anagnostopoulos.

in online platforms with the use of appealing headlines and hence, highly likely to be shared by other users without checking its legitimacy. Therefore, an automated mechanism is required to explore and identify the likelihood of a news headline and content being a clickbait.

The clickbait detection is a challenging task and limited number of research works are available in the literature. There were two major competitions designed to explore various approaches to classify clickbait news items shared in Twitter. The first competition is the Webis clickbait challenge[1] which has designed to develop a classifier to identify the clickbait level of a Tweet using 21,997 labelled dataset. The second competition has published at the Kaggle[2] in 2019 to classify news articles, clickbaits and other textual content, and their dataset consists of 24,870 labelled data. We use both datasets in our analyses to training and evaluate our proposed models. Few other recent studies have tackled the clickbait detection with various state-of-the-art techniques. For example, Potthast *et al.* [4] conducted an initial study on detecting clickbait titles in Twitter using machine learning. They found that top news publishers have used clickbaits on a regular basis where 26% of the tweets were considered to be clickbaits. Few other research works have used deep learning and neural network models [5]–[8]. One disadvantage of using deep and machine leaning models in this task is the lack of training data.

One solution to overcome the limitation of training data is to adapt Transfer Learning models that can be optimised and improved for the clickbait classification task through the transfer of knowledge from a related task that has already been learned. BERT [9], one of the first language representation modelling based on Transfer Learning introduced by Google, is achieved state-of-the-art performance on a number of natural language understanding tasks [12]. Subsequently, there are many Transfer Learning models that are improved over BERT. In this work, we consider three Transfer Learning models that are widely used in NLP tasks namely, BERT [9], XLNet [13] and RoBERTa [14]. We believe that these models represent most of other Transfer Learning models in terms of the architectural properties and training datasets. RoBERTa has the same architecture as BERT while it is pre-trained on a large data corpus, whereas XLNet has a different unique architecture, but exhibited higher performance than BERT. According to the author's knowledge, there is a limited number of works on using Transfer Learning for clickbait detection and no any comparative analysis of many Transfer Learning models focused on clickbait classification.

The main objective of this research is to apply several fine-tuning strategies on BERT, XLNet and RoBERTa models such as generalisation, model compression (pruning) and model expansion techniques to identify best configuration parameters for the clickbait classification. The contributions of this paper are: 1) how to use Transfer Learning to improve news clickbait detection in social media?, 2) comprehensive analyses on different fine-tuning strategies that can be applied on BERT,XLNet and RoBERTa with the focus on clickbait classification, 3) compare and evaluate the performance of modified Transfer Learning models with our benchmark model, the best clickbait detection model presented at the Webis Clickbait challenge and 4) apply the proposed models in an outer-domain dataset (Kaggle dataset) to observe how they behave in an unseen clickbait dataset.

II. RELATED WORK

With the rapid growth of social media clickbaits became very popular research topic and therefore, there several competitions have introduced to explore clickbait content. One successful competition was Webis clickbait challenge, introduced in 2017 [15], which was launched to detect clickbait posts in Twitter. In this competition, competitors asked to develop classifiers to rate how clickbaiting a social media post is using 19,538 labelled posts and 18,979 unlabelled posts. We use this dataset in our analyses to train and evaluate proposed Transfer Learning models for clickbait detection. There were 28 groups participated to this challenge[3] and groups were ranked based on the mean square error (MSE) of suggested models by each group. The highest accuracy is achieved by Zhou *et al.* [40] and therefore, we consider this as our benchmark. We observed that almost all the proposed approaches were mainly based on machine learning and deep learning techniques.

Another clickbait classification competition was introduced by Kaggle [16] to explore various semi-supervised and transfer learning approaches where competitors should classify articles into news, clickbait and other. Their dataset includes separate training (24,870 entries), testing (5,646 entries) and validation (3,551 entries) datasets. We observed from the leader-board that only 9 participants have participated[4] and achieved considerable results. We noted that, competitors mainly used transfer learning models without any additional improvement. In our experiments, we use their training dataset to evaluate how our models perform in an unseen dataset. After observing the results of the competition, we noted that clickbait detection still can be improved with the use of transfer learning approaches by adding several modifications to the default architectures.

During last year, number of studies have worked on clickbait detection in social media platforms mainly focused on Twitter and Instagram platforms [1], [4], [5], [7], [8]. Few other studies have used data obtained from news headlines extracted from news articles, blogs, and other sources [1], [6], [17], [18]. Some studies have focused only on the textual content while others have used both text and images to characterise clickbaits [19]. Researches have worked on detecting clickbaits with natural language processing techniques, machine learning [8], [40] and using linguistic differences

[1]https://www.clickbait-challenge.org/

[2]https://www.kaggle.com/c/clickbait-news-detection/overview

[3]https://www.tira.io/task/clickbait-detection/dataset/clickbait17-test-170720/

[4]https://www.kaggle.com/c/clickbait-news-detection/leaderboard

or features [20]. As per authors knowledge very limited number of works detected clickbait content using transfer leaning techniques. This research aims to apply and evaluate transfer learning models that are designed for NLP tasks to discern clickbaits from non-clickbaits by mainly modifying and improving the default architecture. Many NLP based transfer learning models were introduced with the disclosure of Transformers by Vaswani *et al.* [11]. One of the most popular language modelling algorithm that use deep bidirectional transformers is BERT [9]. The initial use of BERT is to predict the next sentence while it broke the records of previous state-of-the-art methods in eleven different NLP tasks [12]. Following that, many extensions to the BERT model have proposed such as RoBERTa, DistilBERT and ALBERT.[5] XLNet [13] is another transformer modelling approach that is designed to overcome the issues came up with the BERT as BERT uses extra tokens that are not important in the training phase. RoBERTa [14] is another transfer learning model that has the same architecture as BERT, but trained on a large data corpus. Few recent studies have used BERT for multitask classification [23], [24] covering common text classification tasks such as sentiment analysis, question and topic classification, intention classification [26], and Fake news detection [27]–[29]. We identified that only one study conducted on detecting stances of Fake News using BERT, XLNet and RoBERTa without adding many configuration changes to the existing models [30].

The most important aspect of using transfer learning for NLP tasks is to fine-tune models appropriately with the limited number of labelled data. One approach is model compression which is a mechanism used in the literature to compressed certain components of BERT that are unnecessary when training the model [31]. Pruning is a popular compression techniques introduced to remove or identify less important components in the model. Element wise pruning and structural pruning are the two main categories of pruning. Element wise pruning is more focused on locating least important weights in the model [32], and structural pruning focuses on pruning the architectural components of the model such as pruning layers and attention heads [41]. In this work we consider layer pruning when fine-tuning BERT, XLNet and RoBERTa models and compare their performance individually on the clickbait classification task. Other fine-tuning techniques includes data quantisation (reduces the number of bits used to represent weights) and knowledge distillation (trains a smaller model using outputs from one or more larger models) [31]. Few of other studies have also tried to modify the architecture by expanding the model [33].

To the best of authors knowledge, this research work is the first attempt to apply multiple transfer learning model for clickbaits detection and the first study to compare the performance of multiple transfer learning models integrating different architectural parameters.

[5]https://github.com/huggingface/transformers

III. METHODOLOGY

Typically, clickbaits spread in social media in the form of short messages that refers to certain web content advertisements. Content publishers, mainly news media, discovered clickbait as an effective way of drawing attention to their news websites [4]. After reading such a message, readers get the impression that something is refereed to or some emotional reaction is delivered. One example for a clickbait is 'Here is What Actually Reduces Gun Violence'. These types of messages easily attract readers' attention and entice them to click on the provided link. Hence, detection of clickbait messages are challenging as it is required to observe text syntactic and associated link references. Therefore, natural language understanding techniques can be adapted to inspect clickbait content in social media. One of the biggest challenges in NLP related tasks is the lack of training data. In order to overcome this issue we can rely on recent Transfer Learning models on NLP tasks such as BERT [9]. BERT is a new language representation model, introduced by Google in 2018, which is designed to pre-train deep bidirectional representations from unlabelled text using Transformers [11]. In recent years, a set of new other algorithms were also proposed by advancing the BERT model and they outperformed BERT on many NLP benchmark datasets, usually within a large margin [12]. In this study, we will use three popular and representative transfer learning models for clickbait classification task by applying several fine-tuning strategies.

A. TRANSFER LEARNING MODELS FOR CLICKBAIT CLASSIFICATION

At present, there exists two pre-training objectives that have been successful in transfer learning NLP models, namely; autoregressive (AR) and auto-encoding (AE) language models. AR language modelling cannot model with bidirectional context and encode text in uni-directional, either forward or backward, but this has been successful in several downstream tasks such as question answering and sentiment analysis. On the other hand, AE based pre-training models can work with bidirectional context and therefore ease of reconstructing original data from corrupted data. A popular example of such modelling is used in BERT (Google AI) [9], an effective state-of-the-art technique used to address several NLP tasks. RoBERTa (Facebook AI) [14] is another model that uses AE language modelling which has the similar architecture as BERT, but pre-trained with a large data corpus. One example for a model which uses AR language modelling is XLNet (Google AI) [13] and the architecture of XLNet is different than BERT and RoBERTa. One similarity among BERT, XLNet and RoBERTa is that they rely on independent layers stacked on top of each other and use only the Transformer encoder within the model. Due to these architectural similarities and different modelling properties of three models mentioned above, we believe they can be considered as representative models and therefore we will consider them in the clickbait classification task by fine-tuning with

TABLE 1. Comparisons of BERT, XLNet and RoBERTa.

	BERT	XLNet	RoBERTa
No of parameters - Base (Million)	110 (12 layers, 12 attention heads, 768 hidden size)	110 (12 layers, 12 attention heads, 768 hidden units)	125 (12 layers, 12 attention heads, 768 hidden units)
No of parameters - Large (Million)	340 (24 layers, 16 attention heads, 1024 hidden size)	340 (24 layer, 16 attention heads, 768 hidden units)	355 (24 layer, 16 attention heads, 768 hidden units)
Performance	Outperforms SoTA in October 2018	2-15% improvement over BERT	2-20% improvement over BERT
Sequence length-SL & batch size-BS during training	SL: 128 for 90% of the steps, 512 for remaining 10% & BS: 256	SL: 512 & BS: 8192	SL: 512 & BS: 256
Data	16GB data Wikipedia+ Books corpus, 3.3 Billion words	Base: 16GB BERT data Large: 16GB BERT+ 16GB Giga5+ 19GB ClueWeb 2012-B+ 110GB Common Crawl , 33 Billion words	16GB BERT + 76GB Common Crawl-News + 38GB Open web text+ 31GB Stories
Method	Bidirectional transformers with Masked Language Model-MLM & Next Sentence Prediction-NSP	Bidirectional transformers with permutation language modelling-PLM	BERT with improved NSP

appropriate mechanisms. More details about BERT, RoBERTa and XLNet models are given in the following sections.

1) RoBERTa

RoBERTa (Robustly optimized BERT approach) [14] is developed by improving BERT and therefore share many similar configurations. We can observe from the GLUE leaderboard [12] that RoBERTa performs better than BERT. The main adjustments made by RoBERTa over BERT are bigger training data, using dynamic masking patterns, training on longer sequences and replacing the next sentence prediction. Hence, we can say that RoBERTa tuned BERT by increasing data size and hyper-parameters only. In RoBERTa, dynamic masking was made for each training instance in every epoch by duplicating the training dataset 10 times, so that each sequence is masked in 10 different ways over the 40 epochs of training.

2) BERT

BERT [9] is the first approach that uses deeply bidirectional self-attention mechanism which pre-trained on a large data corpus including a BookCorpus, a dataset consisting of 11,038 unpublished books (plain text corpus) from 16 different genres and 2,500 million words from text passages of English Wikipedia. BERT uses bidirectional contextual model that consider word's previous and next context and hence, referred to as contextual models. Contextual models consider the neighboring words in a sentence and therefore has different representations based on the context of the sentence whereas, word-embedding representations like Word2Vec are context-free models in which, one word in two different sentences in different context has the same representation.

There are two main steps in any Transfer Learning model: pre-training and fine-tuning. During pre-training, model-m trains on a dataset A and during fine-tuning, we use some parameters from the model-m which trained on the dataset A and then, trains the model-m on a new dataset B where it transfers some knowledge obtained from dataset A to dataset B. Pre-training phase in BERT replaces few original tokens with mask tokens ([MASK]) and later, predict the original sentence with the use of AE language modelling by considering the context of the [MASK] token in both backward and forward directions. In addition, BERT assume that the predicted [MASK] tokens are independent from each other. As a result, in order to obtain a better relationship among all the tokens, it is necessary to have a correlation in between unmasked tokens and predicted masked tokens. Initially, BERT model is trained on many unlabelled data corpus (Table 1) considering different scenarios and next, during fine-tuning, the model initializes with pre-trained parameters.

As explained earlier, BERT uses [MASK] symbol to predict missing tokens. For example, suppose for a text sequence x, BERT constructs a corrupted version as $\hat{x}$ by randomly replacing a set of tokens in x with a symbol [MASK]. If the set of marked tokens are $\bar{x}$, the training objective is to reconstruct $\bar{x}$ from $\hat{x}$ as follows where, $m_t = 1$ indicates that x_t is masked, H is the Transformer that maps a given sentence of length T into hidden vectors.

$$\begin{aligned} max\ (log\, \mathfrak{p}\,(\bar{X}|\hat{X})) &\approx \Sigma_{t=1}^{T}\ m_t\ log\, \mathfrak{p}\,(x_t|\hat{X}) \\ &\approx \Sigma_{t=1}^{T}\ m_t\ log\ \frac{exp(H(\hat{X}_t^T)\ e(x_t))}{\Sigma_{\acute{x}}\ exp(H(\hat{X}_t^T)\ e(\acute{x}))} \end{aligned} \quad (1)$$

Two main disadvantages over BERT are; 1) all the masked tokens - $\bar{x}$ and corrupted version - $\hat{x}$ in the joint conditional probability $\mathfrak{p}(\bar{x}|\hat{x})$ are reconstructed separately, and 2) masked tokens are not appeared in the downstream tasks, which creates a pre-train fine-tune discrepancy. The main advantage of the AE language modelling used in BERT is the ability to capture bidirectional context.

There are several approaches to fine-tune BERT. In this study, we modify the architecture of the BERT model during fine-tuning phase by merging additional output layer(s) and also, performing layer pruning to reduce number of layers in the model focusing on clickbait classification task.

3) XLNet

XLNet [13] is another Transfer Learning model introduced by Google AI in 2019, which is a BERT-like a model but, generalized with AR pre-training method and outperforms BERT on several benchmark datasets [12]. In order to overcome the limitation of AE models, mainly the issue of capturing bidirectional context, XLNet has introduced Permutation Language Modelling (PLM) as explain below. Since XLNet uses permutations of occurrences for a considered word, it trains through every possible word in a sequence and hence take much longer time to converge than BERT.

The main idea of XLNet is to use PLM by adding more features to capture bidirectional contexts. If a sentence has x tokens having length T, then in total $T!$ number of different orders can be constructed to perform AR factorization by considering all positions on both sides of a token. Assume Z_T is the all possible permutation of the sequences having length T.

$$max\ \mathbb{E}_z \backsim z_T[\Sigma_{t=1}^{T}\ log\ \mathfrak{p}\ (x_{z_t}|X_{z<t})] \tag{2}$$

where z_t and $z < t$ denotes t-th element and t-1 elements of a permutation Z_T. The XLNet auto-regressive permutation method is shown in equation 2 which calculates the probability of token x_{z_t} given preceding tokens $X_{z<t}$ from any order from Z_T. XLNet only permutes the factorization order not the sequence order where it keeps the original sequence order and use Transformers to achieve the positional encoding corresponding to the original sequence. This is a useful property for fine-tuning as it consider only the natural order in a given sequence. Hence, the architecture of BERT and XLNet are different and as a result, we use XLNet in our analysis in order provide a comparative analysis of different models targeting the clickbait detection task.

A comparison among BERT, XLNet and RoBERTa is presented in Table 1. RoBERTa and XLNet uses larger mini-batches, learning rates and step sizes for longer training along with differences in masking procedure [13], [14]. However, pre-training on more data does not guaranteed that the performance of the model will be high and also, very hard to say which model performs better for a specific task that were pre-trained on different datasets. The model performance is mainly based on the number of model parameters, size of the dataset and the amount of computational power that is necessary for training the model and fine-tuning for a new task. Moreover, Talmor *et al.* [34] found that different models with identical structure and objective functions differ not only quantitatively but also qualitatively. Therefore, it is worth to understand the best model for a clickbait classification task by introducing different fine-tuning strategies for each model separately.

The main focus of this research is to explore clickbaits appeared in the news on social media. One main advantage of using RoBERTa and XLNet for the clickbait classification task is that they were pre-trained on Common Crawl news dataset which contains 63 Million English news articles collected between September 2016 and February 2019. This may help our clickbait classification models to perform better than BERT as RoBERTa and XLNet were pre-trained on news related textual content.

B. FINE-TUNING STRATEGIES USED IN THIS RESEARCH

The models we use in this work were already pre-trained on existing datasets, as shown in Table 1. One main advantage of Transfer Learning is that pre-trained model can reuse on a new task by fine-tuning them appropriately and hence, the next step is to apply several fine-tuning strategies focus on clickbait classification task. We shortlist and propose six fine-tuning strategies, those can be applied on any Transfer Learning model, with the aim of comparing model performances on clickbait classification. In the literature, different ways to fine-tune Transfer Learning models have been presented. Fine-tuning strategies consider in this work are based on three different aspects known as generalization, compression and expansion. Apart from these fine-tuning strategies we use data augmentation strategies in order to balance the training dataset. We modify the default architecture of each model to explore the best model architecture for clickbait classification.

1) GENERALIZATION

Generalization is important for Transfer Learning as the model trains on an unsupervised manner using a large dataset and then, fine-tune the same model to apply in a real-world task using a related supervised dataset. Transfer Learning models need to generalize (in-domain generalization or outer-domain generalization) in order to achieve high accuracy, usually by adding an extra task-specific final layer and fine-tuning on a supervised dataset for the task of interest [35]. The accuracy of the model can be improved after generalization by training multiple times on the same supervised data with different random seeds [36]. Generally, distinct random seeds can lead to substantially different results when fine-tuning the model even with the same hyper-parameters. In our experiments, we execute each model multiple times using the same hyper-parameter values but, modifying the random seed value that control the initialization of the weights of the final classification layer. In our clickbait classification task we will do the generalization and then train multiple times to achieve higher accuracy for each model.

2) COMPRESSION

BERT, XLNet and RoBERTa are exponentially large models since they use huge datasets and millions of parameters during pre-training (Table 1). These pre-trained models can be fine-tuned by applying compression techniques to make them smaller and faster. Compression reduces number of parameters in the model in both during-training and post-training. Post-training does not need any training data while during-training uses training data preserving higher accuracy.

Pruning is one of the compression techniques that modifies the model architecture. There are three different types

of pruning strategies, namely, head pruning - remove less important heads for a specific task, weight pruning - remove unnecessary weights in the architecture, and layer pruning - remove full layer of the transformer and/or dropouts. In this research, our main focus is on the layer pruning and then assess whether the pruning accelerates inference when classifying clickbaits.

The base models of BERT, XLNet and RoBERTa consists of 12 transformer layers and 12 attention heads, resulting in a total of 144 unique attention mechanisms. A non-linear feed-forward layer takes the output from each attention head and operate parallel to one another. Therefore, these models can capture a wide range of relationship among the words in a sentence which leads to form a rich representation as it traverse to the deepest layers of the model. Each attention head learns unique parameters and do not share parameters among other attentions. In general, each attention head is composed of four distinct matrices (Wv, Wo, Wk, Wq) generated during training: Wo, Wv - weighted average of output and value vectors and Wq, Wk - query and key vectors that are necessary to compute Wo, Wv having dimension of d vectors. Then each attention head is used to make the computation of multi-headed attention as follows.

$$MultiheadAttention(x, q) \tag{3}$$
$$= \Sigma_{h=1}^{N_h} Attention(W_k^h W_q^h W_v^h W_o^h(x, q)) \tag{4}$$

where N_h is the number of independent parametrized attention heads, h indicates an attention head, d_h is the dimension of a head and each head projects down to a different subspace of size d_h where $d_h = \frac{d}{N_h}$, d is the dimension of the input vector, x represents the input, query q to represent a newly computed sequence of representation, $W_k^h W_q^h W_v^h \in \mathbb{R}^{d_h \times d}$ and $W_o^h \in \mathbb{R}^{d \times d_h}$. To allow all attention heads to interact among them, a non-linear feed-forward layer is used at each transformer layer. Each attention head takes input sequence $x = [x_1, \ldots x_n]$ corresponding to n tokens and each x_i is transformed into query q_i, key k_i, value v_i and output o_i. Weight pruning and head pruning can be done on these weights.

The base models of BERT, XLNet and RoBERTa consist of 12 Transformer layers with 12 attention heads followed by a feed-forward (FF) sublayer which is followed by layer normalization operation. The normalization computes the average and standard deviation of the output activations of a given sublayer and normalizes them. Therefore, the output y_t of a Transformer layer is as follows.

$$y_t = AddNormalization(FF(z_t)), \tag{5}$$
$$z_t = AddNormalization(MultiheadAttention(x_t, q)) \tag{6}$$

The output of each layer (y_t) is a normalized layer obtained from the output of self-attention layers including residual connections and bias. The entire model stack consists of those 12 layers having a dimension of 768 hidden units. The final transformation is applied on the [CLS] token at the final hidden state which has the size of $d * d$ linear transformation, named as a pooled output. In general, by default, output from the BERT, RoBERTa, XLNet returns a pooled output of dimension [1, 1, 768] which is the embedding of [CLS] token. We can access linear transformation of each layer that returns from the model as a sequence output in which the output dimension could be [1, n, 768] where n is the maximum number of tokens. Moreover, we can access the hidden states of all three models that is the output of each layer and therefore, each model consists of 12 hidden states. We apply several layer pruning strategies in the scenario of during-pruning by considering the sequence outputs and hidden outputs of the models.

3) EXPANSION

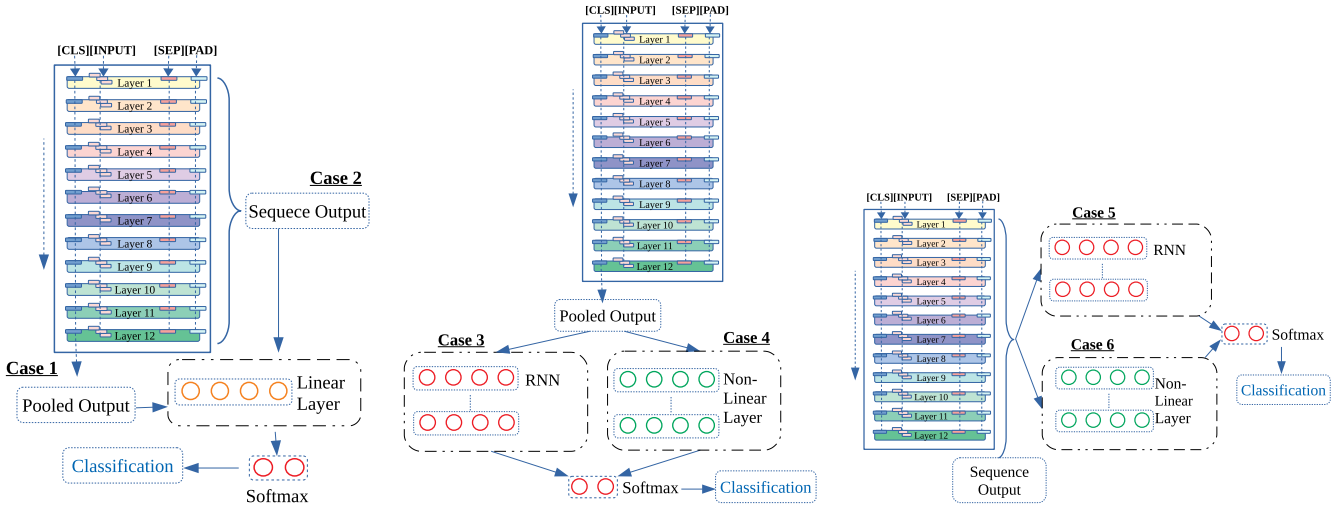
Another experiment we conduct in this research is the analysis on the behaviour and the accuracy of the model after adding additional layers to the output (both pooled or hidden) from the model. By default, these models use a linear layer after obtaining a pooled-output for any classification as shown in Figure 2(a) as Case 1. The expansion of the model by means of adding new layers at the latest layer might have a significant improvement to the accuracy of the model when it is used for classification. Hence, we will analyse whether the addition of novel layers affects the performance of the model.

4) DATA AUGMENTATION

In any classification task, a balanced dataset helps to generate clear decision boundaries with respect to each class and help models to classify the data more accurately. Data augmentation techniques can be adapted to make any unbalanced dataset to a balanced dataset which makes the dataset consistence across different labels. The SMOTE algorithm [37], a popular data augmentation strategy that can be applied for any dataset without biasing predictions on a specific label. SMOTE over samples the minority class using a k-Nearest Neighbors classifier by selecting samples that are close in the feature space and create synthetic data points. A general downside of this approach is that synthetic examples are created without considering the majority class, possibly resulting ambiguous examples and might overfit without proper fine-tuning. In this research we use SMOTE to make the dataset balanced across each label and then evaluate how it affects on the performance of models.

C. CLICKBAIT DETECTION METHODOLOGY USING BERT, XLNet AND RoBERTa

This section discusses about the series of experiments we conduct for the clickbait classification task by modifying default output parameters of each model using fine-tuning strategies mentioned in the previous section. As detailed in Table 1, BERT, XLNet and RoBERTa were trained on different datasets including the news media data. In this research, as the focus is to investigate best clickbait classification approach to discern clickbaits from non-clickbaits, the generalization is necessary due to the learning is performed from one domain to another. We use supervised learning techniques to generalize each transfer learning model to a



(a) *Case 1*: Pooled output + linear layer. *Case 2*: Sequence output + linear layer

(b) *Case 3*: Pooled output + RNN layer. *Case 4*: Pooled output + non-linear layer

(c) *Case 5*: Sequence output + RNN layer. *Case 6*: Sequence output + non-linear layer

FIGURE 1. The architectural modifications of the Transfer Learning models.

clickbait classification task and use two different labelled datasets obtained from Twitter where they contain news items labelled as clickbait or non-clickbait.

We consider 8 different experiments in this work and for ease of referring, we entitled them with 8 Cases. Figure 1 elucidates 6 configuration changes done on the default BERT, XLNet and RoBERTa models for exploring a better fine-tuning parameters for our generalization task. We expand the default models by adding a new layer to the pooled output and hidden output as explained below.

Case 1: The default architecture of BERT, XLNet and RoBERTa is shown by the Case 1 in Figure 1(a). By default, models use the pooled output, which is the output of [CLS] token and then use a linear layer followed by a softmax layer to make the classification. The linear layer is a fully connected neural network that projects the vector produced by the [CLS] token in to a large vector called as logits vector and then softmax layer turns the scores into probabilities. Next, we conduct a series of experiments in which we try to modify the default output of each model with five different possibilities by expanding the model architecture as shown in Figure Figure 1 from Case 2-6.

Case 2: The only change we did in Case 2 compared to Case 1 is that, instead of using the pooled output, we concatenated the hidden outputs extracted from all 12 layers to make the final vector as shown in Figure 1(a). With this experiment, we can understand whether the output from [CLS] performs better or considering the output from all hidden outputs.

Case 3: In Case 3, our aim is to explore the performance change when we integrate a Recurrent Neural Network (RNN) to the model's output as elaborated in Figure 1(b). In our experiments, we use a BiLSTM layer with 2 recurrent layers. BiLSTM layer takes the input as the output of each model that is the number of encoded tokens return by models. BiMSTM uses two LSTM layer which process the input back and forth and produces a sequence of hidden states which encodes the current token the previous knowledge. In Case 3, we try to explore how the performance of the model changes after integrating a BiLSTM layer.

Case 4: In Case 4, instead of RNN layer we added a non-linear layer to the output of the model as shown in Figure 1(b). We replaced the linear layer with a LeakyReLU layer as it allows models for better gradient propagation and efficient computation.

Case 5: Figure 1(c) shows the architectural changes for Case 5 where we consider the sequence output and then add BiLSTM layer before the classification. *Case 6:* In Case 6 our aim is to explore model performances when we integrate non-linear layer with the sequence output as indicated in Figure 1(c).

Case 7: Models can be compressed after downstream training and during pre-training. In this research, we use pruning strategies in the downstream training to change the distribution of the weights in an entire attention head as there can be redundancy in the transformer models. Michel *et al.* [38] showed that up to 40% of attention heads can be pruned from BERT without affecting the test accuracy due to the redundancy. Hence, it is worth to analyse the performance of the models after compressing on our downstream task after pruning entire layer(s). This fine-tuning approach is named as Case 7 in our analyses. Thus, we prune layers without affecting the output of the network but, it might give a different training loss and performance than the default architecture. The results represented by Case 7 uses models by integrating layer pruning strategies.

Case 8: We observed that the Webis-Clickbait training dataset is imbalanced, as shown in Table 2, in which the ratio of clickbait:non-clickbait is almost 1:3. Hence, the

TABLE 2. Number of posts and clickbait to non-clickbait ratio in each sub-dataset of Webis-Clickbait-17 dataset.

Dataset	#posts	Clickbait:Not
A (Labelled)	2,459	1 : 2.23
B (Labelled)	19,538	1 : 3.10
C (Unlabelled)	18,979	N/A

validation results and predictions are highly likely to classify as non-clickbaits while having less predictions as clickbaits. To overcome this issue, we use data augmentation method called SMOTE that can oversample the minority class of the dataset and this experiment is listed as Case 8 in our analyses. Case 8 represents the results obtained from the models after integrating data augmentation techniques.

IV. DATASET AND EXPERIMENTS

In our experiments, we use two different labelled clickbait datasets, for training and evaluating the trained models. Next, we utilize several performance metrics, such as accuracy, recall, precision, F1-score and Matthews Correlation Coefficient (MCC), to evaluate the models rather than evaluating with a single metric.

A. DATASET DESCRIPTION

1) TRAINING DATASET

The Webis Clickbait Corpus 2017[6] (Webis-Clickbait-17) is used as the training dataset in this work which comprises a total of 40,976 Twitter posts in JSON format obtained from 27 major US news publishers. JSON files contain instances such as post text (tweet), title of target article and description tag of target article. We use only the post text in this experiment and will extend our analysis by considering the other instances in the future research. The data corpus includes two labelled datasets for training (A: 2,495 posts and B: 19,538 posts) and one unlabelled dataset for testing (18,979 posts). Table 2 summarizes the exact size of each of these datasets and ratios of clickbait to non-clickbait.

Tweets were annotated on a 4-point scale: not click baiting (0.0), slightly clickbaiting (0.33), considerably clickbaiting (0.66) and heavily clickbaiting (1.0) by five annotators from Amazon Mechanical Turk. Among all the posts, a total of 9,276 posts are considered clickbait by the majority of annotators considering only the post text. As our aim is to discern clickbaits from non-clickbaits we need only binary classification and hence, we consider clickbait posts as the ones with a score of 0.5 or greater and non-clickbait posts with a score below 0.5. The dataset C is only accessible through TIRA evaluation board [7] which cannot be downloaded at present. We use the raw training dataset as it is without following any pre-processing techniques. We merge dataset A and B (total of 21,997 labelled data) for training and validating the models and also, compare the performance matrix with the best models proposed at the competition [21].

[6]https://webis.de/data/webis-clickbait-17.html

[7]https://www.tira.io/task/clickbait-detection/dataset/clickbait17-test-170720/

2) TESTING DATASET

We use Kaggle dataset 'Train a clickbait detector'[8] as the testing dataset. The Kaggle Clickbait detection task aims to classify news articles into three different categories: 'news', 'clickbait' and 'other'. It is provided with a separate training (24,870 entries), testing (5,646 entries) and validation (3,551 entries) datasets. We use this training dataset to evaluate how our models will perform in a new environment as they were trained and fine-tuned using the Webis-Clickbait-17 dataset. Since this dataset consist of three labels, in order to make it a binary classification we re-labelled 'news' and 'other' as 'non-clickbaits'. In addition, we pre-processed the testing dataset in order to eliminate posts that are in different languages and hence, only 18,397 labelled posts are considered for the analyses.

B. EXPERIMENTS

As detailed in the previous sections, we conduct a set of different experiments using three models fine-tuned with different strategies to distinguish clickbaits from non-clickbaits. Each model is evaluated with several metrics that help us to understand the most effective Transfer Learning model for clickbait detection. The model evaluation metrics are Accuracy, Precision, F1 score, Recall and MCC-Matthews correlation coefficient. Classification accuracy gives the correct predictions made as a ratio of all predictions made by the model, precision or positive predicted value refers to the fraction of related predictions among all retrieved predictions, recall is also referred as the sensitivity of a model which computes fraction of related predictions retrieved over the total amount of relevant instances, F1 score uses both precision and recall parameters to compute the score and MCC is a correlation coefficient between the observed and predicted binary classification which returns a value between -1 and 1 where -1 indicates incorrect prediction and +1 represents a perfect prediction. MCC produces a high score when the prediction results are considerable in all confusion matrix categories (true positives, false negatives, true negatives, and false positives), proportionally to the size of positive and negative elements in the dataset, and therefore MCC gives better results than accuracy and F1 score [39], and as a result model comparisons using MCC value is much accurate.

In addition, our analyses use the results of the confusion matrix and present the result of an epoch in Table 2 when the model receive maximum number of True Positive values. Hence, with all these metrics we can properly evaluate the performance of each model and fine-tuning strategy with the focus on clickbait classification.

1) FINE-TUNING BERT, XLNet AND RoBERTa

This part includes the fine-tuning strategies and experimental results used in this work for the clickbait classification downstream task.

[8]https://www.kaggle.com/c/dlinnlp-spring-2019-clf

The previous solutions to Webis Clickbait challenge were mainly based on the deep neural networks. Creating a very efficient neural network for a classification task can be very expensive as it is required to train using millions of parameters and also mandatory to train the neural network from the scratch at each execution. In addition, it is obligatory to have a large corpus of a training dataset in order to achieve better performance. Nevertheless, for many benchmark tasks [12], Transfer Learning models executed within much less time than training a neural network and utilized small training datasets to train lower layers of the model by fine-tuning for any downstream task. We execute our modified transfer learning algorithms in Google Colab.[9] In order to make fair comparisons with the best performed model of the Webis Clickbait challenge, which is proposed by Zhou *et al.* [40], we executed their model in the Google Colab, the same platform where we execute our fine-tuned models for clickbait classification.

Before fine-tuning BERT, XLNet and RoBERTa, we first tokenized and formatted the input where input sentences are splitted into tokens, prepend with the [CLS] token to the start, appended the [SEP] token to the end, padded to maximum length and finally, created the attention masks. The maximum sequence length set to be 128 for each model and training and validation split is assigned as 90% for training and 10% for validation. As our clickbait detection task is a classification problem, we modified classification classes of each model (*BertForSequenceClassification*, *XLNetForSequenceClassification*, *RobertaForSequenceClassification*) in the huggingface pytorch interfaces. We use these classification classes provided by all the models for fine-tuning.

1) *BertForSequenceClassification* is a Bert Model transformer with a sequence classification/regression head on top (a linear layer on top of the pooled output),

2) *XLNetForSequenceClassification* is a XLNet Model with a sequence classification/regression head on top and

3) *RobertaForSequenceClassification* is a RoBERTa sequence classification model with a linear layer on top of the pooled output.

The outputs of these default models are loss, logits (classification) and hidden states (one for the output of the embeddings + one for the output of each layer). By default, all three classification models act as a sentence classifier and use a single linear layer on top for classification. As we feed input data, the entire pre-trained model and the additional untrained classification layer is trained on the target specific task. Hence, one of the main fine-tuning strategies used in this research is to change the last layer of the classifier and output layer (analysing eight different cases) in order to evaluate any improvement of the performance of the model over the default layered architecture.

There are few different pre-trained models exist such as base (12 layers), large (16 layers), uncased (only lower case letters) and cased models. In our experiments we use 'base' models (bert-base-uncased, xlnet-base-cased and roberta-base) in order to make all the architectures comparable. BERT has both cased and uncased models but we observed that the bert-base-uncased model exhibited higher performance than bert-base-cased and therefore we used bert-base-uncased in our experiments. XLNet has only the cased model while RoBERTa do not have any cased or uncased versions. Next, we need to choose the training hyper-parameters within the stored default models including learning rate - 2e-5, and batch size - 32 for each execution. We trained all modes multiple times (maximum of 50 times) and obtained the average comparison matrix for the best performed epoch of each execution.

In addition to the default fine-tuning strategies, as explained in the previous chapter we use set of added fine-tuning approaches by modifying the default architecture of the models. Initially we used six different architectural modifications as shown in Figure 1 where we add new layers and expanded existing model to explore any performance improvement compared with the default configurations. Then, we use another experiment with pruning and with data augmentation techniques.

2) EXECUTION RESULTS AND OBSERVATIONS FROM THE PRE-TRAINED MODELS

In this experiments, we have used the dataset explained in Section IV-A1. The execution results are shown in Table 3 using five different metrics: MCC value, Accuracy, Precision, Recall and F1 score and also, provided True Positive and False Positive values for the validation test as well. We executed each model for 50 times and Table 3 shows the average results of all the epochs that had the highest number of True Positive values. Few observations from the validation results are explained below.

As explained in Section III-C, we executed 8 Cases separately and then we identified which model performed best for each Case based on the performance matrix mainly considering MCC value and F1 score. The higher the values received for F1 score and MCC, the higher the model performance will be. We can observe from the results that, RoBERTa shown higher performance than BERT and XLNet for 6 Cases and XLNet performed better for the remaining 2 cases (Case 4 and Case 8). The best performance for RoBERTa is exhibited with Case 2 and Case 6 showing the highest MCC value and accuracy. In both these cases, the output is generated by considering the hidden outputs. RoBERTa in Case 6 exhibited higher value for the F1 score than Case 2. XLNet models presented their higher values for MCC in Case 4, but its accuracy is lower than many models considered in our experiments and also had the largest number of False Positive value in the Table 3. In addition, as XLNet is based on permutations, the convergence time is much higher than the other models. Hence, we can conclude that RoBERTa performed better for clickbait classification tasks and its performance can be improved when we consider the hidden outputs. Apart from that, we can observe by expanding the RoBERTa

[9]https://colab.research.google.com/notebooks/intro.ipynb

TABLE 3. The execution results for eight different fine-tuning cases introduced in Section III-C.

Model		MCC	F1 score	Accuracy	Precision	Recall	TP	FP
Case 1	BERT	0.5416	0.6473	0.8368	0.6643	0.6312	63.12	9.92
	XLNet	0.5714	0.5649	0.8436	0.4182	0.8704	64.94	9.67
	RoBERTa	***0.5727***	***0.6725***	***0.8468***	***0.6824***	***0.6628***	***66.28***	**9.59**
Case 2	BERT	0.5741	0.6762	0.8448	0.6695	0.6830	68.30	10.49
	XLNet	0.5769	0.6852	0.8350	0.6392	0.7383	73.83	13.39
	RoBERTa	***0.5947***	***0.6817***	***0.8593***	***0.7358***	***0.6351***	***63.51***	**7.09**
Case 3	BERT	0.5587	0.6618	0.8418	0.6716	0.6523	65.23	9.92
	XLNet	0.5855	0.5697	0.8457	0.4222	0.8755	66.19	9.41
	RoBERTa	***0.5882***	***0.5699***	***0.8492***	***0.4219***	***0.8778***	***66.25***	**9.22**
Case 4	BERT	0.5611	0.6660	0.8402	0.6607	0.6715	67.15	10.73
	XLNet	***0.5916***	***0.6938***	***0.8459***	***0.6713***	***0.7178***	***71.78***	**11.29**
	RoBERTa	0.5885	0.6822	0.8543	0.7071	0.6590	65.90	8.49
Case 5	BERT	0.5519	0.6580	0.8380	0.6590	0.6571	65.71	10.58
	XLNet	0.5811	0.5738	0.8399	0.4283	0.8692	67.32	10.13
	RoBERTa	***0.5843***	***0.6827***	***0.8498***	***0.6843***	***0.6810***	***68.10***	**9.77**
Case 6	BERT	0.5561	0.6612	0.8395	0.6625	0.6600	66.00	10.46
	XLNet	0.5877	0.6889	0.8473	0.6826	0.6953	69.53	10.39
	RoBERTa	***0.5982***	***0.6901***	***0.8575***	***0.7130***	***0.6686***	***66.68***	**8.37**
Case 7	BERT	0.5863	0.6814	0.8530	0.7011	0.6628	66.28	8.79
	XLNet	0.5913	0.5704	0.8522	0.4218	0.8809	66.39	8.98
	RoBERTa	***0.5922***	***0.6871***	***0.8541***	***0.6994***	***0.6753***	***67.53***	**9.03**
Case 8	BERT	0.5571	0.6602	0.8416	0.6723	0.6485	64.85	9.83
	XLNet	***0.5968***	***0.6915***	***0.8550***	***0.6982***	***0.6849***	***68.49***	**9.21**
	RoBERTa	0.5574	0.6608	0.8414	0.6706	0.6513	65.13	9.95

TABLE 4. The comparison of the results between the best performed model at the *Webis Clickbait challenge* and our best performed transfer learning model (The binary classification results in this Table is same as the scores presented for the binary classification in Table 3).

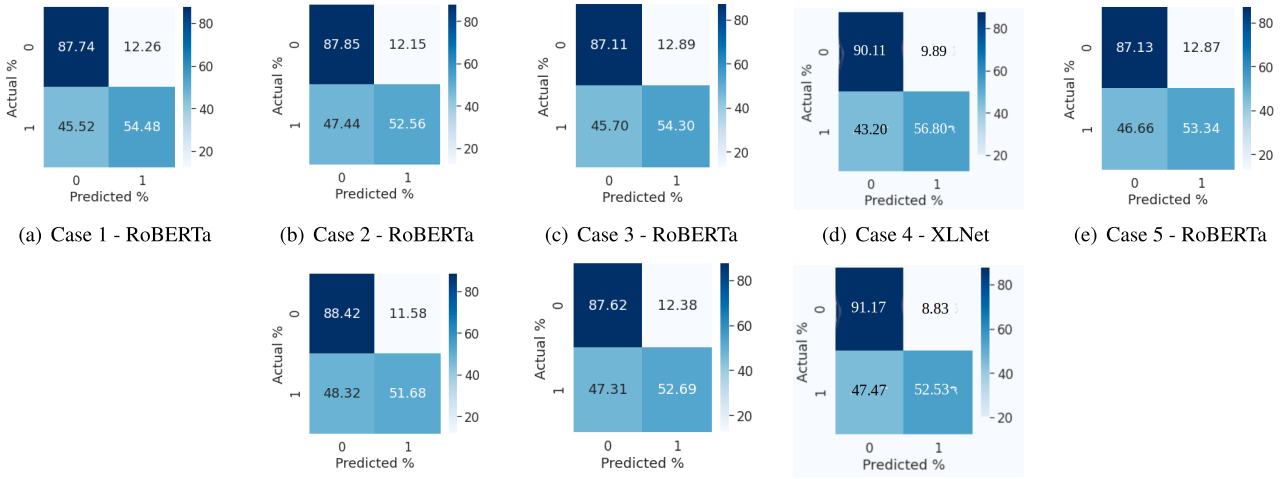
	F1 Score	Precision	Recall	Accuracy	Execution Environment
Zhou et al. (4-label classification)	0.683	0.719	0.650	0.856	TIRA platform
Zhou et al. (binary classification)	0.569	0.558	0.582	0.666	Google Colab
RoBERTa (Case 6-binary classification)	0.690	0.713	0.668	0.858	Google Colab

model adding RNN layers do not significantly improve the performance but, replacing a linear layer with a non-linear layer improved the performance. Hence, by changing the architecture of the default models, it is possible to improve the performance of the classification task.

Next, we executed (at the same platform where we executed our Transfer Learning models) the best scored model proposed at the Webis clickbait challenge [40] that exhibited the highest accuracy compared to the other proposals in the competition.[10] The model proposed by Zhou *et al.* [40] was a multi-class classification approach which classifies news content into 4 different classes. They submitted multi-class classification approach to the TIRA platform and achieved 0.856 accuracy as shown in Table 4. However, our clickbait classification method is a binary classification approach and execution platform is Google Colab. In order to make a reasonable comparison among our proposed models and the Zhou's model, we modified and optimised their code to make binary classifications and executed in the Google Colab. The results are shown in Table 4 as *binary classification*. We can observe that, performance of the binary classification results of the Zhou's model are not satisfactory and shows low values for the F1 score and accuracy as indicated in Table 3. For binary classification, Transfer Learning model achieved significant improvement over F1 score compared with the traditional deep learning models. This indicates that, transfer learning models perform well in the clickbait classification task.

Results for Case 7 in Table 3 indicates the performance after applying layer pruning. We conduct a set of experiments by pruning the layers of the models as, considering only the latest 1 layer, 2 layers, 3 layers, 4 layers, 5 layers and 6 layers. The best performance achieved when we considered last 4 layers and therefore we presented this result in Table 3. Another major observation from our fine-tuned models is that layer pruning (Case 7) exhibited higher accuracy for BERT, XLNet and RoBERTa compared to other fine-tuned strategies. Hence, we will expand the pruning strategies of the models with novel approaches in the future studies to improve the performance of the models.

[10]https://www.clickbait-challenge.org/



(a) Case 1 - RoBERTa (b) Case 2 - RoBERTa (c) Case 3 - RoBERTa (d) Case 4 - XLNet (e) Case 5 - RoBERTa

(f) Case 6 - RoBERTa (g) Case 7 - RoBERTa (h) Case 8 - XLNet

FIGURE 2. Evaluation results of the models in outer-domain environment (Kaggle clickbait dataset). Label 1 and 0 represents clickbait and non-clickbait, respectively.

After adding SMOTE for the training dataset in Case 8, the results shown that there is no significant improvement on the model performance when we compared with the results obtained for the default model parameters in Case 1. To conclude here, from all the scenarios we considered in our analyses, RoBERTa and XLNet performed better than BERT and in many scenarios RoBERTa performed better than XLNet.

3) MODEL EVALUATION WITH AN OUTER-DOMAIN DATASET

One major challenge for transfer learning models is to apply fine-tuned models in an real environment. Because, we train models in one domain and then apply those models in an outer-domain where this leads to give less performance than what we expected. Hence, in order to understand how our models are performing in outer-domain classification we tested our models on another dataset obtained from Kaggle and explained in Section IV-A2 'Train a clickbait detector'.[11] This dataset consists of 3,748 clickbaits out of 18,398 samples and the remaining are non-clickbaits.

In this experiment we considered only the model that performed best for each Case. Figure 2 shows confusion matrix for all Cases after applying Kaggle dataset with our trained models. We can observe from all confusion matrices that True Positive values are always larger than 51% and exhibits at most 13% False Positive values. However, False Negative values are in the rage from 43-47% for all cases and Case 4 (XLNet) shows the least False Negative value. As a result, we still need to improve model performance with different fine-tuning strategies and use a large labelled dataset. We explain some of the advancement that we can conduct to improve model performances in Section IV-C. To conclude here, at least a half of the clickbaits from the Kaggle dataset can be classified with the Transfer Learning models. The best result is achieved from the Case 4 which uses XLNet adding a bi-directional LSTM layer with the pooled layer.

[11] https://www.kaggle.com/c/dlinnlp-spring-2019-clf

4) CONCLUDING REMARKS

In this work, we have used 8 different Cases to explore BERT, XLNet and RoBERTa with the aim of exploring a best transfer Learning model for clickbait classification task. We 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). The addition of a new bi-directional LSTM layer do not exhibited any significant improvement over the other configuration changes but, when we changed the non-linear layer to a linear layer models performed better. In addition, we observed that RoBERTa performed better than other two models in many cases. This is obvious that the RoBERTa is pre-trained on a large data corpus than other models and also training data includes news media data as well. RoBERTa has detected the least number of false positive when we fine-tuned it by considering hidden outputs together with non-linear layer at the end. As a result, we can conclude that, model performance can be improved further by experimenting with advanced pruning techniques and considering hidden output parameters of the model.

C. POSSIBLE FUTURE EXPERIMENTS

The experiments we conducted in this research were mainly based on the transfer learning models. The main fine-tuning and configuration changes we did on BERT, XLNet and RoBERTa are; model expansion by adding new layer to the existing architecture, use data augmentation methods for training the model and applying layer pruning strategies.

Apart from these changes, in future research works, we will evaluate the performance of the model by modifying the transfer layer architecture by adding changes to the attentions such as add/remove attentions in each layer and keeping necessary weight matrix without dropping them and pruning based on the weights. Another future research direction we can consider with this down-stream task is to adapt other features in the dataset such as the headline of the news and keywords that might give a higher prediction results than considering only the postText. We can consider the syntactic features of the clickbait posts to understand the correlation between postText, headline and news articles that help to classify clickbait content very accurately than focusing only on the social media post. The trained model then can be used in the real-world dataset to explore clickbait vs non-clickbait content shared by the news media in social media. Apart from that, we will also try to adapt other transfer learning models that are more efficient than these models for exploring a better model for clickbait tasks. In addition, since the Webis dataset consists of information about the level of a text being clickbait under four different labels (not click baiting (0.0), slightly clickbaiting (0.33), considerably clickbaiting (0.66) and heavily clickbaiting (1.0)), a novel multi-label classifier model can be proposed to classify under 4 defined classes. Another important research direction is the analysis of fake news in terms of how clickbaity they are and this will help to understand the propergation behaviours of fake news if they are clickbaits.

V. CONCLUSION

Clickbaits are usually used to attract readers attention to news articles by using eye-catchy headlines. Identification of a news headline or social media post as a clickbait or a non-clickbait is a challenging task. Previous research works tried to adapt various machine learning and deep learning models for clickbait detection. In this study, we used transfer learning models to explore clickbait content in Twitter that are posted by news media. We experimented with three well known transfer learning models, namely BERT, XLNet and RoBERTa, and used two labelled data corpus; the Webis Clickbait Corpus 2017 for training and validation as well as the Kaggle Clickbait challenge dataset for experimenting with the trained models. We used 3 fine-tuning stratergies, namely model generalization, model compression (layer pruning) and model expansion, to experiment each model with 8 different cases. In addition, we also experimented with data augmentation strategies using well known SMOTE algorithm. We changed model configurations during fine-tuning by expanding them using a BiLSTM layer, applying a non-linear layer at the output where output vector is generated from hidden outputs and/or sequence outputs, or used the default pooled output. 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. The XLNet model convergence time is higher than the other models due to applying permutation mechanisms in the training. We also, compared our model performances with the best performed model at the Webis clickbait challenge. Our proposed transfer learning models outperformed the proposed models at the Webis clickbait challenge that were mainly based on the classic deep learning approaches. The results of binary classification shows that our models performed better with more than 19% accuracy and significantly increased the True Positive values. Finally, we considered our best performed models explored with 8 cases and applied them to another labelled dataset (Kaggle clickbait challenge) to understand the behaviour of the model in an outer-domain environment. Experimental results shows that, with the outer domain dataset, fine-tuned transfer learning models exhibited more than 52% of False Positive values and around 12% False Negative values in all 8 scenarios.

REFERENCES

[1] A. Anand, T. Chakraborty, and N. Park, "We used neural networks to detect clickbaits: You won't believe what happened next!" in *Proc. Eur. Conf. Inf. Retr.* Cham, Switzerland: Springer, 2017, pp. 541–547.

[2] P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Scrutinizing news media cooperation in Facebook and Twitter," *IEEE Access*, early access, Mar. 7, 2019, doi: 10.1109/ACCESS.2019.2902491.

[3] P. Rajapaksha, R. Farahbakhsh, N. Crespi, and B. Defude, "Inspecting interactions: Online news media synergies in social media," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2018, pp. 535–539.

[4] M. Potthast, S. Köpsel, B. Stein, and M. Hagen, "Clickbait detection," in *Proc. Eur. Conf. Inf. Retr.* Cham, Switzerland: Springer, 2016, pp. 810–817.

[5] V. Kumar, D. Khattar, S. Gairola, and Y. K. Lal, "Identifying clickbait: A multi-strategy approach using neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2018, pp. 1225–1228.

[6] H.-T. Zheng, J.-Y. Chen, X. Yao, A. Sangaiah, Y. Jiang, and C.-Z. Zhao, "Clickbait convolutional neural network," *Symmetry*, vol. 10, no. 5, p. 138, May 2018.

[7] M. Dong *et al.*, "Similarity-aware deep attentive model for clickbait detection," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Springer, 2019, pp. 56–69.

[8] A. Omidvar, H. Jiang, and A. An, "Using neural network for identifying clickbaits in online news media," in *Proc. Annu. Int. Symp. Inf. Manage. Big Data*. Cham, Switzerland: Springer, 2018.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[10] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*.

[11] A. Vaswani, N. Shazeer, and N. Parmar, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[12] GLUE. *The General Language Understanding Evaluation (GLUE) Benchmark*. Accessed: Sep. 15, 2021. [Online]. Available: https://gluebenchmark.com/leaderboard

[13] Z. Yang *et al.*, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[14] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[15] Bauhaus-Universität Weimar. *Webis Clickbait Challenge*. Accessed: Sep. 15, 2021. [Online]. Available: https://www.clickbait-challenge.org/

[16] iPavlov Research Group. *Clickbait News Detection*. Accessed: Sep. 15, 2021. [Online]. Available: https://www.kaggle.com/c/clickbait-news-detection

[17] A. Chakraborty, B. Paranjape, S. Kakarla, and N. Ganguly, "Stop clickbait: Detecting and preventing clickbaits in online news media," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2016, pp. 9–16.
[18] M. M. U. Rony, N. Hassan, and M. Yousuf, "Diving deep into clickbaits: Who use them to what extents in which topics with what effects?" in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Jul. 2017, pp. 232–239.
[19] Y. Ha, J. Kim, D. Won, M. Cha, and J. Joo, "Characterizing clickbaits on Instagram," in *Proc. Int. AAAI Conf. Web Social Media*, 2018, vol. 12, no. 1.
[20] M. Glenski, E. Ayton, D. Arendt, and S. Volkova, "Fishing for clickbaits in social images and texts with linguistically-infused neural network models," 2017, *arXiv:1710.06390*.
[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[22] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018, *arXiv:1802.05365*.
[23] X. Liu, P. He, W. Chen, and J. Gao, "Multi-task deep neural networks for natural language understanding," 2019, *arXiv:1901.11504*.
[24] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?" in *Proc. China Nat. Conf. Chin. Comput. Linguistics*. Cham, Switzerland: Springer, 2019, pp. 194–206.
[25] C. Zhang and M. Abdul-Mageed, "Multi-task bidirectional transformer representations for irony detection," 2019, *arXiv:1909.03526*.
[26] S. Tang, Q. Liu, and W. Tan, "Intention classification based on transfer learning: A case study on insurance data," in *Proc. Int. Conf. Hum. Centered Comput.* Cham, Switzerland: Springer, 2019, pp. 363–370.
[27] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending against neural fake news," 2019, *arXiv:1905.12616*.
[28] H. Jwa, D. Oh, K. Park, J. Kang, and H. Lim, "ExBAKE: Automatic fake news detection model based on bidirectional encoder representations from transformers (BERT)," *Appl. Sci.*, vol. 9, no. 19, p. 4062, Sep. 2019.
[29] W. Antoun, F. Baly, R. Achour, A. Hussein, and H. Hajj, "State of the art models for fake news detection tasks," in *Proc. IEEE Int. Conf. Informat., IoT, Enabling Technol. (ICIoT)*, Feb. 2020, pp. 519–524.
[30] M. Guderlei and M. Aßenmacher, "Evaluating unsupervised representation learning for detecting stances of fake news," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 6339–6349.
[31] P. Ganesh, Y. Chen, X. Lou, M. Ali Khan, Y. Yang, H. Sajjad, P. Nakov, D. Chen, and M. Winslett, "Compressing large-scale transformer-based models: A case study on BERT," 2020, *arXiv:2002.11985*.
[32] M. A. Gordon, K. Duh, and N. Andrews, "Compressing BERT: Studying the effects of weight pruning on transfer learning," in *Proc. 5th Workshop Represent. Learn. NLP (RepL NLP)*, 2020, pp. 143–155.
[33] G. A. Vlad, M. A. Tanase, C. Onose, and D. C. Cercel, "Sentence-level propaganda detection in news articles with transfer learning and BERT-BiLSTM-capsule model," in *Proc. 2nd Workshop Natural Lang. Process. Internet Freedom, Censorship, Disinformation, Propaganda*. 2019.
[34] A. Talmor, Y. Elazar, Y. Goldberg, and J. Berant, "OLMpics-on what language model pre-training captures," *Trans. Assoc. Comput. Linguistics*, vol. 8, pp. 743–758, Dec. 2020.
[35] D. Yogatama, C. de Masson d'Autume, J. Connor, T. Kocisky, M. Chrzanowski, L. Kong, A. Lazaridou, W. Ling, L. Yu, C. Dyer, and P. Blunsom, "Learning and evaluating general linguistic intelligence," 2019, *arXiv:1901.11373*.
[36] R. Thomas McCoy, J. Min, and T. Linzen, "BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance," 2019, *arXiv:1911.02969*.
[37] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, 2002.
[38] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does BERT look at? An analysis of BERT's attention," 2019, *arXiv:1906.04341*.
[39] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over f1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, pp. 1–13, Dec. 2020.
[40] Y. Zhou, "Clickbait detection in tweets using self-attentive network," 2017, *arXiv:1710.05364*.
[41] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," 2019, *arXiv:1909.11556*.

PRABODA RAJAPAKSHA (Student Member, IEEE) received the B.Eng. degree in computer engineering from the University of Peradeniya, Sri Lanka, in 2010, the M.Sc. degree in communication networks and services from the Institut Mines-Telecom, France, in 2016, the M.Eng. degree in computer science from the Asian Institute of Technology, Thailand, in 2016, and the Ph.D. degree in computer science from the Institut Polytechnique de Paris, France, in 2021. She is currently a Senior Lecturer at the Uva Wellassa University, Sri Lanka, and a Researcher at the Institut Polytechnique de Paris. Her research interests include AI, data science, transfer learning, machine learning, deep learning, and text mining.

REZA FARAHBAKHSH (Member, IEEE) received the Ph.D. degree from Paris VI (UPMC) jointly with the Institut-MinesTelecom, Telecom SudParis (CNRS Lab UMR5157), in 2015. He is currently an Invited Research Fellow at the Institut-Mines Telecom, Telecom SudParis, and a Data Scientist at TOTAL SA. His research interests include AI and data science in scale, online social networks, the IoT, predictive maintenance, large scale measurement, and user behavior analysis. He is actively involved in collaborative projects and represents France in two EU Cost actions.

NOEL CRESPI (Member, IEEE) received the master's degree from the Universities of Orsay and Kent, the diplom-ingnieur degree from Telecom ParisTech, and the Ph.D. and Habilitation degrees from Paris VI University. In 1993, he worked at CLIP, Bouygues Telecom, and then France Telecom Research and Development, in 1995. He joined the Institut Mines-Telecom, Institut Polytechnique de Paris, in 2002, where he is currently a Professor and the M.Sc. Program Director and leading the Data Intelligence and Communication Engineering Laboratory (DICE). He coordinates the standardization activities for the Institut Mines-Telecom at ETSI, 3GPP, and ITU-T. He is also an Adjunct Professor at KAIST, South Korea, an Affiliate Professor at Concordia University, Canada, and a Guest Researcher at the University of Goettingen, Germany. He is the Scientific Director of the French-Korean Laboratory ILLUMINE.