Predicting Clickbait Strength in Online Social Media

Vijayasaradhi Indurthi*, Bakhtiyar Syed, Manish Gupta†, Vasudeva Varma
IIIT Hyderabad, India
{vijaya.saradhi,syed.b}@research.iiit.ac.in
{manish.gupta,vv}@iiit.ac.in

Abstract

Hoping for a large number of clicks and potentially high social shares, journalists of various news media outlets publish sensationalist headlines on social media. These headlines lure the readers to click on them and satisfy the curiosity gap in their mind. Low quality material pointed to by clickbaits leads to time wastage and annoyance for users. Even for enterprises publishing clickbaits, it hurts more than it helps as it erodes user trust, attracts wrong visitors, and produces negative signals for ranking algorithms. Hence, identifying and flagging clickbait titles is very essential. Previous work on clickbaits has majorly focused on binary classification of clickbait titles. However not all clickbaits are equally clickbaity. It is not only essential to identify a clickbait, but also to identify the intensity of the clickbait based on the strength of the clickbait. In this work, we model clickbait strength prediction as a regression problem. While previous methods have relied on traditional machine learning or vanilla recurrent neural networks, we rigorously investigate the use of transformers for clickbait strength prediction. On a benchmark dataset with $\sim$39K posts, our methods outperform all the existing methods in the Clickbait Challenge[1].

1 INTRODUCTION

Clickbait refers to those sensational, provocative or controversial posts which appear to be informative and objective, but are designed to entice its readers into clicking the link accompanying with the post. Fig. 1 shows popular clickbait types with examples.

According to a survey of 53 Stanford students, 96.2 percent of Stanford students encounter clickbait articles on the Internet at least once per day[2]. Rony et al. (2017) estimate that 19.46% of headlines were "clickbait" in 2014; 23.73% in 2015; and 25.27% in 2016. Beyond the increased prevalence, clickbait is also a challenge across multiple modes of data, text, images and even videos[3].

The economic model of the contemporary online news industry (Dvorkin, 2015) incentivizes more content views. A report by the Columbia Journalism Review highlighted the case of online magazine Slant, which pays writers $100 per month, plus $5 for every 500 clicks on their stories. This clearly motivates journalists to write catchy and suspenseful headlines. Table 1 lists some of the popular social media outlets publishing clickbait content and their followers. The numbers are indicative of how much people are easily falling to the bait.

Cognitively, human minds have a tendency to satisfy and bridge their curiosity gap by clicking on the link. Marketing companies have been using clickbaits to attract and engage more number of users resulting in getting more page views. Websites need more page views to promote their content or to create more opportunities to show advertisements which increase their revenue. Moreoever, it is a well

[*] The author is also a research engineer at Teradata.

[†] The author is also an applied researcher at Microsoft.

[1] https://webis.de/events/clickbait-challenge/shared-task.html

[2] https://www.stanforddaily.com/2017/03/20/clickbait-and-conscientiousness/

[3] https://www.cjr.org/analysis/the_mission_sounds_simple_pay.php

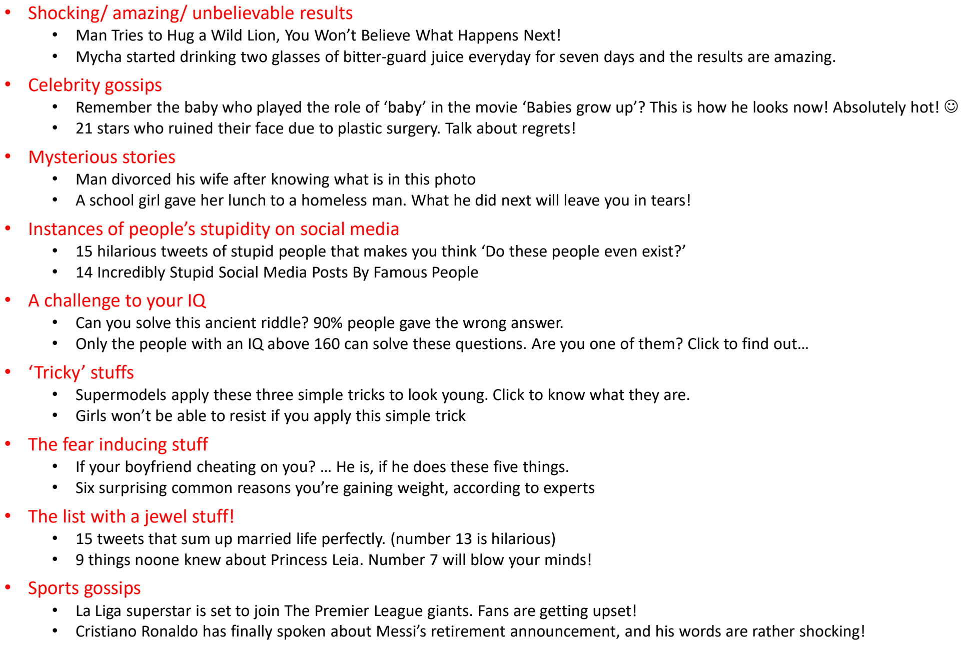
Proceedings of the 28th International Conference on Computational Linguistics, pages 4835–4846
Barcelona, Spain (Online), December 8-13, 2020

Publisher	Followers	Link
BuzzFeed	12M	https://www.fb.com/pg/BuzzFeed/
Upworthy	11M	https://www.fb.com/pg/Upworthy/
ViralStories	8.5M	https://www.fb.com/pg/DailyViralStories/
ScoopWhoop	4.7M	https://www.fb.com/pg/Scoopwhoop/
BuzzFeed India	3M	https://www.fb.com/pg/BuzzFeedIndia/
ViralNova	2.4M	https://www.fb.com/pg/ViralNova/

Table 1: Some of the popular social media outlets publishing clickbait and followers of their Facebook pages (as on 1st April 2020)

known fact that clickbaity content has a higher likelihood of being socially shared leading to more page views.

Clickbaits play with human psychology and sometimes are a wastage of time. They create a curiosity gap for the users through the short post, but do not make judicious attempt to fill it in the clicked article, thereby creating an information void. It is quite annoying to have social feeds spammed by over-promising headlines that lead users to under-delivering half-stories. Even for enterprises which use clickbaits for effective marketing, clickbaits are hurtful more than being helpful for the following reasons: (1) misleading clickbait damages brands and erodes user trust, (2) clickbaits attract wrong visitors rather than interested ones, (3) user interaction with clickbaits produces negative signals for ranking algorithms, (4) clickbait muddles the website's important data, and (5) sensationalism is now seen as more disappointing by smart users.



- Shocking/ amazing/ unbelievable results
  - Man Tries to Hug a Wild Lion, You Won't Believe What Happens Next!
  - Mycha started drinking two glasses of bitter-guard juice everyday for seven days and the results are amazing.
- Celebrity gossips
  - Remember the baby who played the role of 'baby' in the movie 'Babies grow up'? This is how he looks now! Absolutely hot! ☺
  - 21 stars who ruined their face due to plastic surgery. Talk about regrets!
- Mysterious stories
  - Man divorced his wife after knowing what is in this photo
  - A school girl gave her lunch to a homeless man. What he did next will leave you in tears!
- Instances of people's stupidity on social media
  - 15 hilarious tweets of stupid people that makes you think 'Do these people even exist?'
  - 14 Incredibly Stupid Social Media Posts By Famous People
- A challenge to your IQ
  - Can you solve this ancient riddle? 90% people gave the wrong answer.
  - Only the people with an IQ above 160 can solve these questions. Are you one of them? Click to find out…
- 'Tricky' stuffs
  - Supermodels apply these three simple tricks to look young. Click to know what they are.
  - Girls won't be able to resist if you apply this simple trick
- The fear inducing stuff
  - If your boyfriend cheating on you? … He is, if he does these five things.
  - Six surprising common reasons you're gaining weight, according to experts
- The list with a jewel stuff!
  - 15 tweets that sum up married life perfectly. (number 13 is hilarious)
  - 9 things noone knew about Princess Leia. Number 7 will blow your minds!
- Sports gossips
  - La Liga superstar is set to join The Premier League giants. Fans are getting upset!
  - Cristiano Ronaldo has finally spoken about Messi's retirement announcement, and his words are rather shocking!

Figure 1: Clickbait Examples

Clickbait classification is a very subjective task. While there are some terrible headlines that qualify as clear clickbaits (e.g., "You won't believe what happened!"), there is also an enormous gray area[4]. Since the very purpose of teaser message is to attract the attention of readers, every message containing a link baits the user to click the link. The question is whether this baiting is perceived as immoderate or deceptive by the reader (Potthast et al., 2017). Hence, in this work, we focus on the task of predicting the intensity or degree of clickbaity-ness of an article rather than a vanilla binary classification.

Predicting the degree of clickbaity-ness is challenging as clickbaits are short headlines often written in obscured ways which requires high-order semantic understanding. The strength of a clickbait could be defined as a function of how much attention-grabbing the post is, and the gap between what is promised

[4] https://techcrunch.com/2016/09/25/wtf-is-clickbait/

in the headline and what is delivered by the article linked from it. Both of these are difficult to measure automatically. In most of the cases, we may need to predict clickbaity-ness just based on the content of the post. Using the content alone brings in further challenges: (1) it is usually very short, (2) it is often written in convoluted ways, and (3) it requires high-order semantic understanding, often with support of facts from some knowledge base.

In this paper, we make the following main contributions.

- We build multiple regressor models using the current state-of-the-art word embeddings and evaluate the performance of the classifiers over the current state-of-the art methods for clickbait strength prediction.
- We present the first work to investigate application of transformer regression models for the clickbait intensity prediction task.
- We augment transformer-based methods with multiple traditional machine learning regression methods to further improve the regression performance.
- Our experiments with a benchmark dataset result into a new state-of-the-art for the clickbait intensity prediction task.

2 Related Work

2.1 Clickbait Classification

The origin of clickbaits can be traced back to the advent of tabloid journalism. (Rowe, 2011; Blom and Hansen, 2015; Chen et al., 2015) are some of the earliest studies on analysis of linguistic aspects of clickbait, But they did not perform automatic classification. Most of the existing works on automated clickbait detection have been done in the context of binary classification, i.e. predicting whether a given news article's title is a clickbait or not. Traditionally, feature engineering based methods have been proposed (Biyani et al., 2016; Chakraborty et al., 2016; Wei and Wan, 2017). Feature sets include content features, textual similarity features between the headline and the body, informality and forward reference features, sentence structure features, word pattern features, clickbait language features and N-gram features. Machine learning methods like Gradient Boosted Decision Trees (Biyani et al., 2016), Support Vector Machine (SVM) classifier (Chakraborty et al., 2016), co-training (Wei and Wan, 2017) are then use to leverage these features and train a classifier. Features for clickbait detection can be derived from three sources: the teaser message or the post text, the linked article, and metadata for both. While all reviewed approaches derive features from the teaser message, the linked article and the metadata are considered only by (Potthast et al., 2016) and (Biyani et al., 2016). Besides the post text, Zheng et al. (2017) additionally took the user behaviour information into consideration, to improve the performance of clickbait detection on Chinese news articles. Also, recently deep learning techniques have been proposed. Anand et al. (2017) and Rony et al. (2017) use bidirectional Recurrent Neural Network (RNN) (Schuster and Paliwal, 1997) and fastText (Joulin et al., 2016) on word distributed representations, respectively for clickbait detection.

Most of the initial efforts on clickbait detection focused only on news headlines. Recently, there have been efforts at identifying clickbaits from social media like Twitter. Potthast et al. (2016) trained a random forest classifier by extracting various features from the post texts, linked webpages and associated meta information of tweets, to decide if a tweet was a clickbait. Agrawal (2016) trained a Convolutional Neural Network (CNN) (Kim, 2014), using the post texts only, to detect clickbait posts in Reddit, Facebook and Twitter. In (Chakraborty et al., 2017), researchers analysed the differences in content, sentiment, consumers, etc., between the clickbait and non-clickbait tweets.

2.2 Clickbait Regression

Binary clickbait classification is not sufficient. Rather, it is useful to predict the finegrained intensity of the clickbait which can enable ranking of clickbaits, thereby providing a knob for elimination of clickbaits rather than a blanket binary elimination. The Clickbait Challenge (Potthast et al., 2017) has

Team Name	Method	Paper
carpetshark	Ensemble of Linear SVMs	(Grigorev, 2017)
whitebait	LSTMs, word2vec (Mikolov et al., 2013)	(Thomas, 2017)
pike	Hand-crafted 331 features, Linear, Logistic, Random Forest regression	(Cao et al., 2017)
tuna	Character level embeddings using CNNs, word2vec (Mikolov et al., 2013), LSTMs	(Gairola et al., 2017)
torpedo	Hand-crafted features, GloVe (Pennington et al., 2014), Linear Regression	(Indurthi and Oota, 2017)
salmon	Hand-crafted features, XGBoost	(Elyashar et al., 2017)
snapper	65 Hand-crafted features, Stacking	(Papadopoulou et al., 2017)
albacore	Bidirectional GRUs (Cho et al., 2014), GloVe (Pennington et al., 2014)	(Omidvar et al., 2018)
pineapplefish	CNN and Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997)	(Glenski et al., 2017)
zingel	Bidirectional GRUs (Cho et al., 2014) with attention (Bahdanau et al., 2014), GloVe (Pennington et al., 2014)	(Zhou, 2017)

Table 2: Some of the best teams who participated in the Clickbait Challenge. (LSTMs = Long Short Term Memory Networks, CNNs = Convolutional Neural Networks)

been devised in 2017 to enable benchmarking of solutions for the clickbait strength prediction problem. In this challenge, the goal is to predict the intensity of clickbaits rather than just predicting if a particular item is a clickbait or not. Table 2 shows various approaches that have been proposed for the clickbait intensity prediction task. Some approaches use traditional machine learning regression methods using a large set of hand crafted features, while others look at neural architectures (like RNNs, LSTMs, GRUs and CNNs) supported by word embeddings like word2vec and GloVe.

3 Dataset

The Webis Clickbait Corpus consists of 38517 tweets (Potthast et al., 2018). Restricting to English-language publishers, Potthast et al. (2018) obtain a ranking of the top-most retweeted news publishers from the NewsWhip social media analytics service[5]. Taking the top 27 publishers, they used Twitter"s API to record every tweet they published in the period from December 1, 2016, through April 30, 2017. They filtered and sampled from this collection of 459541 tweets to obtain a clean dataset of 38517 tweets. Each of the tweets was annotated for clickbait intensity label by five different workers from Amazon Mechanical Turk (AMT). A 4-point Likert scale was followed with these values: Not clickbaiting (0.0), Slightly clickbaiting (0.33), Considerably clickbaiting (0.66), Heavily clickbaiting (1.0). Of this, 19538 (of which 4761 are clickbaits) tweets have been released for training with labels. The maximum post size is 25. The post lengths follow a normal distribution around a mean of 12. We split the labeled data into 80:20 ratio for training and validation. We perform 5-fold cross validation and compile the results on the validation set.

The remaining 18979 (of which 4515 are clickbaits) tweets are used by the clickbait challenge server for testing the submissions. This test set is private and not accessible publicly. Moreover, there are extremely limited number of test runs which can be submitted to the test server.

Empirical observations reveal that the field postText (text of the post) in the given dataset contributes majorly to decide the intensity of the clickbait. Hence, in spite of the availability of the tweets' metadata like the post media, the title of the target linked page, the content paragraphs and keywords of the target page, the time of the post and caption of every image in the target article, we use only the post text of the tweet to train a machine learning model to predict the clickbait intensity score of each tweet. We leave further exploration of other metadata fields as part of future work.

4 Evaluation Metrics

The goal for the clickbait intensity prediction task is to develop a model that can predict how click baiting a social media post is. The score is a real number between 0 and 1. Mean Squared Error (MSE) with respect to the mean judgments of the annotators is used as the primary evaluation metric. Models whose preditions have the lowest MSE would be ranked on the top. Unlike other classification task, where F1 score or accuracy is the evaluation metric, this challenge focuses more on predicting the intensity of the title than classifying the title as clickbait or not. Official evaluation is done on the platform called TIRA (Potthast et al., 2014). This platform evaluates the predictions by running the code for predictions in a virtual machine in a sandboxed environment which ensures that the test data is kept private and not

[5] https://www.newswhip.com/

revealed to public. Moreover, there are a limited number of times one can predict on the test data to ensure that the models are not trained to overfit the test data. The evaluation platform also computes secondary evaluation metrics such as the Median Absolute Error (MedAE), the F1-Score (F1) and Accuracy (Acc) with respect to the truth class.

5 Approach

We formulate the problem of clickbait strength prediction as a regression problem. We build multiple regression models using pretrained word embeddings, pretrained transformer representations and finetuned transformer representations for clickbait strength prediction. We experiment with various regression algorithms and rigorously investigate the efficacy of these for clickbait strength prediction.

5.1 Word and Sentence Embedding Representations

Word embeddings have been widely used in modern Natural Language Processing applications as they provide semantic vector representation of words. They capture the semantic properties of words and the linguistic relationship between them. These word embeddings have improved the performance of many downstream tasks across many domains like text classification, machine comprehension etc. (Camacho-Collados and Pilehvar, 2018). Multiple ways of generating word embeddings exist, such as Neural Probabilistic Language Model (Bengio et al., 2003), word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), LexVec (Salle et al., 2016), dependency-based embeddings (DepVec) (Levy and Goldberg, 2014) and more recently ELMo (Peters et al., 2018).

ELMo can generate different word embeddings for a word that captures the context of a word – that is its position in a sentence. ELMo achieves this by using two deep unidirectional LSTMs (forward and backward) and then computing embedding for a word as a weighted combination of hidden layer outputs at that position.

Universal Sentence Encoder (Cer et al., 2018) is based on the Transformer encoder (Vaswani et al., 2017) and a deep averaging network. It is trained using unsupervised data from Wikipedia, web news, web question-answer pages and discussion forums, and supervised data from the Stanford Natural Language Inference (SNLI) corpus.

We experiment with two models – ELMo (Peters et al., 2018) and Google's Universal sentence encoder (Cer et al., 2018) representations for transforming the clickbait title into a dense numerical vector representation.

5.2 Transformer Representations

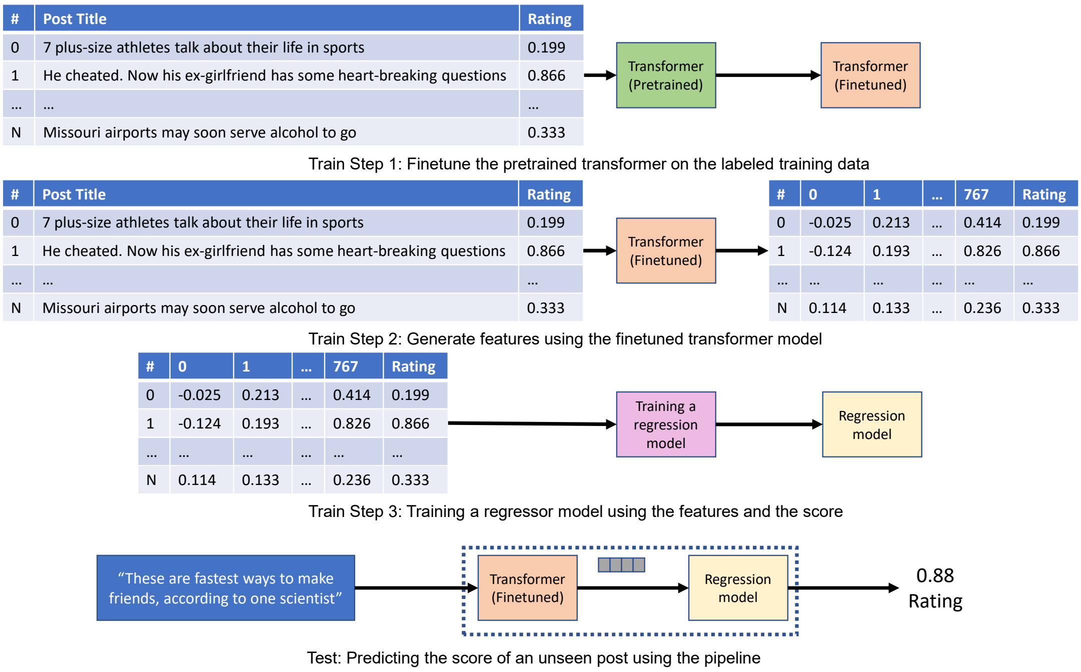
After the original Transformer work by Vaswani et al. (2017), several architectures have been proposed like BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019) and OpenAI's GPT2 (Radford et al., 2019) and T5 (Raffel et al., 2019). The GLUE (Wang et al., 2019b) and the SuperGLUE (Wang et al., 2019a) dashboards indicate the great success of the transformer models which have outperformed all of the previous methods across complex NLP tasks like text classification, textual entailment, machine translation, word sense disambiguation, etc. We present the first of its kind work to investigate the application of transformer models for the clickbait intensity prediction task.

Transformer networks follow a non-recurrent architecture with stacked self-attention and fully connected layers for both the encoder and decoder, each with six layers. They are based on concepts like self attention, multi-head attention, positional embeddings, residual connections and masked attention. While transformers follow an encoder-decoder architecture, just the encoder or the decoder have been used to define other popular architectures like BERT, GPT-2, etc.

BERT (Devlin et al., 2018) essentially is a transformer encoder with 12 layers, 12 attention heads and 768 dimensions. We used the pre-trained model which has been trained on Books Corpus and Wikipedia using the MLM (masked language model) and the next sentence prediction (NSP) loss functions. The post text sequence is prepended with a "CLS" token. The representation C for the "CLS" token from the last encoder layer is used for regression by connecting it to an output softmax layer. We also finetune the pre-trained model using labeled training data for the clickbait intensity prediction task. BERTLarge is similar to BERT but with 24 layers, 16 attention heads and 1024 dimensions.

OpenAI's GPT2 (Radford et al., 2019) uses a left-to-right Transformer, where every token can only attend to previous tokens in the self-attention layers of the Transformer. We also finetune the pre-trained model using labeled training data for the clickbait intensity prediction task. GPT model size is almost the same as the BERT$_{BASE}$ model size. GPT is trained on the BooksCorpus (800M words); BERT is trained on the BooksCorpus (800M words) and Wikipedia (2,500M words). The largest GPT-2 variant is 1.5B parameters large and could take up more than 6.5 GBs of storage space.

RoBERTa (Liu et al., 2019) is a robustly optimized method for pretraining natural language processing (NLP) systems that improves on BERT. RoBERTa was trained with much more data – 160GB of text instead of the 16GB dataset originally used to train BERT. It is also trained for larger number of iterations up to 500K. Compared to BERT, batch sizes for training were 8K instead of 256 in the original BERT base model. Further, it uses larger byte-pair encoding (BPE) vocabulary with 50K subword units instead of character-level BPE vocabulary of size 30K used for BERT. Finally, compared to BERT, it removes the next sequence prediction objective from the training procedure, and a dynamically changing masking pattern is applied to the training data. RoBERTaLarge has configuration similar to BERTLarge.



#	Post Title	Rating
0	7 plus-size athletes talk about their life in sports	0.199
1	He cheated. Now his ex-girlfriend has some heart-breaking questions	0.866
…	…	…
N	Missouri airports may soon serve alcohol to go	0.333

Train Step 1: Finetune the pretrained transformer on the labeled training data

#	Post Title	Rating
0	7 plus-size athletes talk about their life in sports	0.199
1	He cheated. Now his ex-girlfriend has some heart-breaking questions	0.866
…	…	…
N	Missouri airports may soon serve alcohol to go	0.333

#	0	1	…	767	Rating
0	-0.025	0.213	…	0.414	0.199
1	-0.124	0.193	…	0.826	0.866
…	…	…	…	…	…
N	0.114	0.133	…	0.236	0.333

Train Step 2: Generate features using the finetuned transformer model

#	0	1	…	767	Rating
0	-0.025	0.213	…	0.414	0.199
1	-0.124	0.193	…	0.826	0.866
…	…	…	…	…	…
N	0.114	0.133	…	0.236	0.333

Train Step 3: Training a regressor model using the features and the score

"These are fastest ways to make friends, according to one scientist"
Transformer (Finetuned)
Regression model
0.88
Rating

Test: Predicting the score of an unseen post using the pipeline

Figure 2: **Proposed train-test approach for the *Transformer Regression* model**

5.3 Regression Models

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

- **Simple Linear Regression (LR)**: Linear regression is the most simplest of the regression algorithms typically fitted using the least squares approach. The relationship between the independent variable is modeled as a linear combination of the attributes.
- **Ridge Regression (RR)**: Ridge Regression model is a linear regression model with L2 penalty as regularizers.
- **Gradient Boosted Regression (GBR)**: GB regression learns an ensemble of regression trees, each of which have scalar values in the leaves. The ensemble of trees is produced by computing, in

each step, a regression tree that approximates the gradient of the loss function, and adding it to the previous tree with coefficients that minimize the loss of the new tree. The output of the ensemble on a given instance is the sum of the tree outputs.

- **Random Forest Regression (RFR)**: A random forest regressor is an ensemble learning algorithm for regression which constructs multiple decision trees at training time and outputting the average of the predictions of the individual trees, there by prevents over-fitting.
- **Adaboost Regression (ABR)**: AdaBoost regressor is another ensemble learning algorithm that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but the weights of instances are adjusted according to the error of the current prediction, thereby subsequent regressors focus more on the difficult cases.

6 Experiments

In this work, we try various kinds of approaches and investigate how they perform for the clickbait prediction task. We use multiple models to transform the text into features. We train multiple regression models on the above features and evaluate the efficacy of each of the pre-trained embeddings (BERT, GPT2 or RoBERTa) for the downstream clickbait prediction task.

We experimented with the following regression techniques: (1) Simple Linear Regression (LR), (2) Ridge Regression (RR), (3) Gradient Boosted Regressor (GBR), (4) Random Forest Regression (RFR), (5) Adaboost Regression (ABR).

Empirical observations reveal that the field postText (text of the post) in the given dataset contributes majorly to decide the intensity of the clickbait. Hence, in spite of the availability of the tweets' metadata like the post media, the title of the target linked page, the content paragraphs and keywords of the target page, the time of the post and caption of every image in the target article, we use only the post text of the tweet to train a machine learning model to predict the clickbait intensity score of each tweet. We leave further exploration of other metadata fields as part of future work.

First, we experiment with pretrained word and sentence embedding representations. In this setting, we transform the text using the pretrained word embedding or the pretrained sentence encoder. These representations are used to train a regression model. Table 5 shows results using this setting on the validation set.

Next, we experiment with pretrained Transformer representations. In this setting, we transform the train and the test data using the pretrained transformer without any finetuning step. The representation C for the "CLS" token from the last encoder layer of the pretrained transformer models are used as features for these regression methods. Table 3 shows results using this setting on the validation set.

Further, we experiment with finetuned Transformer representations. In this setting, we finetune the pretrained transformer model with the labeled data. After finetuning, we use the finetuned transformer model to transform the input text into vector representations and fit a regressor model on these representations.The representation C for the "CLS" token from the last encoder layer of the finetuned transformer models are used as features for these regression methods.

The training method involves two stage training process. In the first step we finetune the Transformer model using the training data to create the finetuned Transformer model. In the next stage, the finetuned Transformer model is used to generate the representations of the training data. These representations are further used to train a regression model. Figure 2 explains these steps in detail.

For predicting the intensity of an unseen sample, first we transform the input post text into features using the finetuned Transformer model. These features are fed into the trained regressor which predicts a numeric score for the post. The predicted clickbait score is rectified by passing through a rectifier function as defined below to ensure that the clickbait score remains in the interval [0,1]. Bottom part of Figure 2 shows the flow for prediction.

For the final and official evaluation, we have used the complete training dataset for training the model. This model is used to make predictions on the unseen official test set. As there were limited number of runs allowed for the final test runs to prevent participants from over-fitting the test data, we submitted

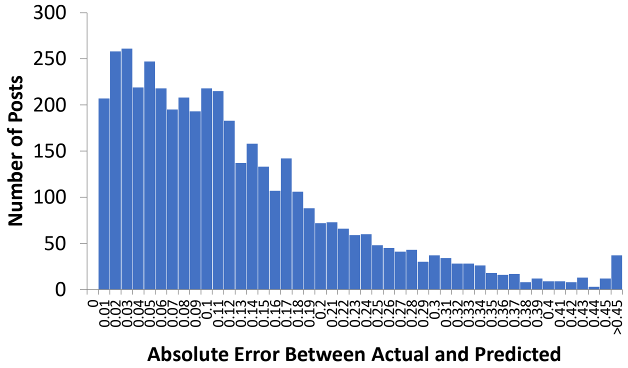


Figure 3: Histogram of the absolute values of the errors produced by the model

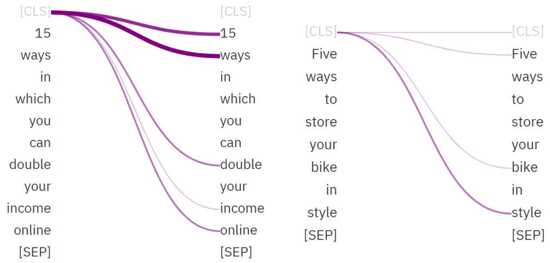


Figure 4: Attention Visualization of the last Transformer layer for two clickbait examples

only those models which have done fairly well on the validation set. Table 4 shows the results on the validation set.

7 Results and Analysis

Pretrained	Method	MSE	MedAE	F1	Accuracy
BERT	LR	0.0313	0.1151	0.5878	0.8215
BERT	RFR	0.0354	0.1266	0.5311	0.8135
BERT	RR	0.0308	0.1140	0.5906	0.8258
BERT	ABR	0.0473	0.1735	0.5104	0.8074
BERT	GBR	0.0346	0.1246	0.5455	0.8115
RoBERTa	LR	**0.0281**	**0.1100**	**0.6527**	**0.8442**
RoBERTa	RFR	0.0360	0.1333	0.5012	0.8133
RoBERTa	RR	0.0285	0.1128	0.6362	0.8404
RoBERTa	ABR	0.0443	0.1662	0.5876	0.8162
RoBERTa	GBR	0.0334	0.1253	0.5519	0.8185
RoBERTaLarge	LR	0.0286	0.1115	0.6286	0.8383
RoBERTaLarge	RFR	0.0371	0.1357	0.4641	0.8092
RoBERTaLarge	RR	0.0283	0.1116	0.6280	0.8379
RoBERTaLarge	ABR	0.0478	0.1748	0.4037	0.7976
RoBERTaLarge	GBR	0.0348	0.1273	0.4923	0.8103
BERTLarge	LR	0.0357	0.1291	0.5850	0.8210
BERTLarge	RFR	0.0380	0.1340	0.5017	0.8060
BERTLarge	RR	0.0345	0.1279	0.5854	0.8226
BERTLarge	ABR	0.0502	0.1757	0.4599	0.7974
BERTLarge	GBR	0.0383	0.1370	0.5125	0.8094
GPT2	LR	0.0669	0.1941	0.0401	0.7386
GPT2	RFR	0.0626	0.1944	0.0053	0.7461
GPT2	RR	0.0643	0.1941	0.0457	0.7434
GPT2	ABR	0.0667	0.2030	0.0141	0.7470
GPT2	GBR	0.0624	0.1942	0.0194	0.7473

Table 3: Results on the validation set using just the pretrained Transformer model representations

Finetuned	Method	MSE	MedAE	F1	Accuracy
BERT	NNR	0.0283	0.1071	0.6859	0.8514
BERT	LR	0.0270	0.1050	0.6771	0.8495
BERT	RFR	0.0280	0.1067	0.6768	0.8500
BERT	RR	0.0270	0.1049	0.6761	0.8495
BERT	ABR	0.0276	0.1047	0.6372	0.8413
BERT	GBR	0.0275	0.1061	0.6792	0.8518
GPT2	NNR	0.0275	0.1009	0.6776	0.8441
GPT2	LR	0.0271	0.1034	0.6660	0.8418
GPT2	RFR	0.0282	0.1067	0.6663	0.8411
GPT2	RR	0.0267	0.1018	0.6651	0.8404
GPT2	ABR	0.0281	0.1146	0.6618	0.8404
GPT2	GBR	0.0269	0.1014	0.6734	0.8459
RoBERTa	NNR	0.0251	0.0951	**0.7018**	0.8518
RoBERTa	LR	0.0248	0.1000	0.6952	0.8539
RoBERTa	RFR	0.0257	0.1000	0.6806	0.8468
RoBERTa	RR	0.0244	0.0984	0.6923	0.8525
RoBERTa	ABR	0.0263	0.1210	0.6974	0.8552
RoBERTa	GBR	**0.0241**	**0.0972**	0.6960	0.8539
BERTLarge	NNR	0.0267	0.1042	0.6798	0.849
BERTLarge	LR	0.0253	0.1029	0.6756	0.8527
BERTLarge	RF	0.0256	0.1046	0.6739	0.8513
BERTLarge	RR	0.0252	0.1022	0.6746	0.8522
BERTLarge	AR	0.0263	0.1060	0.6666	0.8495
BERTLarge	GBR	0.0256	0.1036	0.6690	0.8502
RoBERTaLarge	NNR	0.0289	0.1047	**0.7018**	**0.8570**
RoBERTaLarge	LR	0.0255	0.1022	0.6831	0.8545
RoBERTaLarge	RFR	0.2533	0.1013	0.6850	0.8547
RoBERTaLarge	RR	0.0252	0.1004	0.6870	0.8556
RoBERTaLarge	ABR	0.0256	0.1067	0.6799	0.8531
RoBERTaLarge	GBR	0.0253	0.1006	0.6851	0.8549
RoBERTaLarge	XGBR	0.2520	0.1003	0.6831	0.8541

Table 4: Results on the validation set using Finetuned Transformer model representations

Table 5 show results using different machine learning regression methods using word and sentence embeddings. Tables 3 and 4 show results using pretrained-transformer representations and finetuned transformer representations respectively. Among the pretrained word and sentence embedding methods in Table 5, the best MSE/MedAE is obtained using Universal Sentence Encoder and with Ridge Regression. The best F1/Acc is obtained using ELMo with Linear Regression. Among the transformer based methods in Table 4, the best MSE and MedAE is obtained using RoBERTa approach and with GB regression. On the other hand, with respect to classification metrics, RoBERTaLarge with NNR performs best. We also experimented with just the finetuning approach (without any extra regressor augmented at the last layer, i.e., just using a neuron in the output layer of the neural network). We call this method as neu-

Embedding	**Method**	**MSE**	**MedAE**	**F1**	**Acc**
ELMo	LR	0.0293	0.1130	**0.6308**	**0.8390**
ELMo	RFR	0.0368	0.1267	0.5348	0.8066
ELMo	RR	0.0292	0.1131	0.6298	0.8385
ELMo	ABR	0.0405	0.1576	0.5288	0.8194
ELMo	GBR	0.0309	0.1185	0.583	0.8260
Universal Encoder	LR	0.0286	0.1132	0.6179	0.8323
Universal Encoder	RFR	0.0349	0.1267	0.5611	0.8169
Universal Encoder	RR	**0.0283**	**0.1120**	0.6155	0.8346
Universal Encoder	ABR	0.0411	0.1643	0.5415	0.8137
Universal Encoder	GBR	0.0310	0.1192	0.5624	0.8203

Table 5: Results on the validation set using pretrained word and sentence embedding based methods

	Post	Actual score	Predicted score
Good Predictions	Pete Shotton (early John Lennon bandmate and childhood friend) has died at 75	0.07	0.07
	Lady Gaga's #Joanne album set for #SuperBowl-fueled rise on the Billboard 200 Chart	0.07	0.07
	Five ways to store your bike in style	0.73	0.73
	What % Lucky Are You?	0.87	0.87
Bad Predictions	5 things to know about the GOP health care plan's score	1.00	0.52
	#Pratyusha's former boyfriend #RahulRaj thinks this is a publicity stunt	1.00	0.52
	This animated map shows the largest company by revenue for every state @BI_Video	0.07	0.52
	What's happening with Bitcoin and where it's heading next.	0.13	0.58

Table 6: Examples from validation set: Top part shows examples where model predictions were accurate. Bottom part corresponds to examples where model predictions were wrong.

Team Name	MSE	MedAE	F1	Prec.	Recall	Acc.
goldfish_1 (ours)	**0.0242**	**0.1015**	**0.7408**	0.7394	0.7422	**0.8764**
goldfish_2 (ours)	0.0245	0.1026	0.7330	0.7389	0.7271	0.8740
goldfish_3 (ours)	0.0280	0.1117	0.7098	0.7288	0.6917	0.8654
torpedo19_1 (ours)	0.0303	0.1241	0.6774	0.7548	0.6144	0.8608
albacore (Omidvar et al., 2018)	0.0315	0.1220	0.6703	0.7315	0.6186	0.8553
torpedo19_2 (ours)	0.0325	0.1254	0.6645	0.7443	0.6002	0.8558
torpedo19_3 (ours)	0.0325	0.1253	0.6645	0.7441	0.6002	0.8558
blobfish	0.0326	0.1188	0.6457	0.7382	0.5739	0.8502
zingel (Zhou, 2017)	0.0333	0.1315	0.6827	0.7188	0.6501	0.8563
anchovy	0.0340	0.1358	0.6792	0.7170	0.6452	0.8550
icarfish	0.0357	0.1338	0.6213	0.7681	0.5216	0.8487
emperor	0.0359	0.1337	0.6406	0.7139	0.5810	0.8449
carpetshark (Grigorev, 2017)	0.0362	0.1390	0.6381	0.7282	0.5679	0.8468
ray	0.0365	0.1435	0.6913	0.6799	0.7030	0.8506
electriceel	0.0384	0.1393	0.5881	0.7266	0.4939	0.8354
arowana	0.0391	0.1412	0.6564	0.6587	0.6540	0.8371
pineapplefish (Glenski et al., 2017)	0.0414	0.1392	0.6313	0.6422	0.6208	0.8275
whitebait (Thomas, 2017)	0.0429	0.1392	0.5648	0.6990	0.4738	0.8263
clickbait17-baseline	0.0435	0.1543	0.5521	0.7582	0.4341	0.8324
pike (Cao et al., 2017)	0.0446	0.1094	0.6037	0.7111	0.5245	0.8362
tuna (Gairola et al., 2017)	0.0457	0.1123	0.6537	0.6543	0.6532	0.8354
torpedo (Indurthi and Oota, 2017)	0.0792	0.2363	0.6499	0.5297	0.8405	0.7846
houndshark	0.0994	0.3206	0.0231	**0.7794**	0.0117	0.7641
humuhumunu...	0.1174	0.3045	0.3831	0.2385	**0.9736**	0.2540
dory	0.1182	0.1991	0.4667	0.3799	0.6050	0.6712
salmon	0.1743	0.3967	0.2609	0.1673	0.5926	0.2086
snapper	0.2524	0.4272	0.4341	0.2868	0.8926	0.4464

Table 7: Official Results on the test data from the Clickbait Challenge leaderboard. Best results highlighted in bold.

ral network regression (NNR). Note that (1) the finetuning+ML regressors approach typically provides better results compared to the NNR method. (2) finetuned models have lower MSE compared to the corresponding pretrained models (especially GPT2 where the pretrained-only model performs poorly). (3) Larger Transformer models like RoBERTaLarge and BERTLarge do not lead to lower MSE/MedAE values, probably because of relatively small labeled data.

Finally, we show results on the test set by comparing them across several baselines in Table 7 also available on the Clickbait Challenge Leaderboard[6] as on 23-Nov-2019. Note that 6 of the top 7 are our approaches. Details of these approaches are as follows: goldfish_1 is RoBERTa + GBR, goldfish_2 is RoBERTa + RR, goldfish_3 is GPT2 + LR, torpedo19_1 is Universal Encoder + RR, torpedo19_2 is ELMo + RR, torpedo19_3 is ELMo + LR.

Figure 3 shows the histogram of the absolute value of the errors produced by the model on the predictions. We can observe that for most of the posts the error between the actual and the predicted value is less than 0.2. Very few samples have error in the range of 0.2 to 0.45. There are a very few posts whose error is greater than 0.45. Further, Table 6 shows examples of posts where our proposed gives good predictions as well as those where our model fails. Higher score implies high degree of clickbait. We show examples for both high as well as low clickbait strength.

Finally, in Figure 4 we show attention visualization for average attention that the [CLS] token pays to various words in the post in the last Transformer layer. For the first example, the words "15", "ways", "double" and "income" have high attention values – intuitively, these words indicate clickbaity-ness as well. Similarly, clickbaity words in the second example like "Five" and "style" have high attention values.

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

8 Conclusion

In this paper, we proposed various methods for clickbait intensity prediction based on the title of the post. Using a benchmark dataset from the Clickbait Challenge, we evaluate multiple models; we are the first to investigate effectiveness of Transformer based models for this task. As of now, we rank at the top on the official leaderboard for the challenge. We plan to work on reducing the model size and improve runtime latency using popular knowledge distillation methods.

References

Amol Agrawal. 2016. Clickbait detection using deep learning. In *Next Generation Computing Technologies (NGCT)*, pages 268–272.

Ankesh Anand, Tanmoy Chakraborty, and Noseong Park. 2017. We used neural networks to detect clickbaits: You won't believe what happened next! In *ECIR*, pages 541–547.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *JMLR*, 3(Feb):1137–1155.

Prakhar Biyani, Kostas Tsioutsiouliklis, and John Blackmer. 2016. 8 amazing secrets for getting more clicks: Detecting clickbaits in news streams using article informality. In *AAAI*.

Jonas Nygaard Blom and Kenneth Reinecke Hansen. 2015. Click bait: Forward-reference as lure in online news headlines. *Journal of Pragmatics*, 76:87–100.

Jose Camacho-Collados and Mohammad Taher Pilehvar. 2018. From word to sense embeddings: A survey on vector representations of meaning. *J. AIR*, 63:743–788.

Xinyue Cao, Thai Le, et al. 2017. Machine learning based detection of clickbait posts in social media. *arXiv preprint arXiv:1710.01977*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop clickbait: Detecting and preventing clickbaits in online news media. In *ASONAM*, pages 9–16.

Abhijnan Chakraborty, Rajdeep Sarkar, Ayushi Mrigen, and Niloy Ganguly. 2017. Tabloids in the era of social media?: Understanding the production and consumption of clickbaits in twitter. *HCI*, 1(CSCW):30.

Yimin Chen, Niall J Conroy, and Victoria L Rubin. 2015. Misleading online content: Recognizing clickbait as false news. In *Multimodal Deception Detection*, pages 15–19.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jeffrey Dvorkin. 2015. Column: Why click-bait will be the death of journalism. *pbs. org/newshour/making-sense/what-you-dont-know-aboutclick-bait-journalism-could-kill-you*.

Aviad Elyashar, Jorge Bendahan, and Rami Puzis. 2017. Detecting clickbait in online social media: You won't believe how we did it. *arXiv preprint arXiv:1710.06699*.

Siddhartha Gairola, Yash Kumar Lal, Vaibhav Kumar, and Dhruv Khattar. 2017. A neural clickbait detection engine. *arXiv preprint arXiv:1710.01507*.

Maria Glenski, Ellyn Ayton, Dustin Arendt, and Svitlana Volkova. 2017. Fishing for clickbaits in social images and texts with linguistically-infused neural network models. *arXiv preprint arXiv:1710.06390*.

Alexey Grigorev. 2017. Identifying clickbait posts on social media with an ensemble of linear models. *arXiv preprint arXiv:1710.00399*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Vijayasaradhi Indurthi and Subba Reddy Oota. 2017. Clickbait detection using word embeddings. *arXiv preprint arXiv:1710.02861*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *ACL*, volume 2, pages 302–308.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

Amin Omidvar, Hui Jiang, and Aijun An. 2018. Using neural network for identifying clickbaits in online news media. In *Annual Intl. Symp. on Info. Management and Big Data*, pages 220–232.

Olga Papadopoulou, Markos Zampoglou, Symeon Papadopoulos, and Ioannis Kompatsiaris. 2017. A two-level classification approach for detecting clickbait posts using text-based features. *arXiv preprint arXiv:1710.08528*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GLoVe: Global Vectors for Word Representation. In *EMNLP*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2014. Improving the Reproducibility of PAN's Shared Tasks: Plagiarism Detection, Author Identification, and Author Profiling. In *CLEF*, pages 268–299.

Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. 2016. Clickbait detection. In *ECIR*, pages 810–817.

Martin Potthast, Tim Gollub, Matthias Hagen, and Benno Stein. 2017. The Clickbait Challenge 2017: Towards a Regression Model for Clickbait Strength. In *Clickbait Challenge*.

Martin Potthast, Tim Gollub, Kristof Komlossy, Sebastian Schuster, Matti Wiegmann, Erika Patricia Garces Fernandez, Matthias Hagen, and Benno Stein. 2018. Crowdsourcing a Large Corpus of Clickbait on Twitter. In *27th International Conference on Computational Linguistics (COLING)*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Md Main Uddin Rony, Naeemul Hassan, and Mohammad Yousuf. 2017. Diving deep into clickbaits: Who use them to what extents in which topics with what effects? In *ASONAM*, pages 232–239.

David Rowe. 2011. Obituary for the newspaper? tracking the tabloid. *Journalism*, 12(4):449–466.

Alexandre Salle, Marco Idiart, and Aline Villavicencio. 2016. Matrix factorization using window sampling and negative sampling for improved word representations. *arXiv preprint arXiv:1606.00819*.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. on Signal Processing*, 45(11):2673–2681.

Philippe Thomas. 2017. Clickbait identification using neural networks. *arXiv preprint arXiv:1710.08721*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint 1905.00537*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Wei Wei and Xiaojun Wan. 2017. Learning to identify ambiguous and misleading news headlines. *arXiv preprint arXiv:1705.06031*.

Hai-Tao Zheng, Xin Yao, Yong Jiang, Shu-Tao Xia, and Xi Xiao. 2017. Boost clickbait detection based on user behavior analysis. In *APWeb-WAIM*, pages 73–80.

Yiwei Zhou. 2017. Clickbait detection in tweets using self-attentive network. *arXiv preprint arXiv:1710.05364*.