CARE-BERT: Clickbait Detector using Self-attentive Network with Bidirectional Encoder Representations from Transformers

Changyuan Qiu Chenshu Zhu Haoxuan Shan
University of Michigan
{peterqiu, jupiterz, shanhx}@umich.edu

0. Project Context and Background

Introduction

Clickbait is a rampant problem haunting online readers. The motivation behind clickbaiting is to boost site traffic (and therefore, advertisement revenue) by exploiting the curious nature of human readers. The click-bait technique works by dangling a hyperlink with enticing headlines to lure people into clicking; and then redirect them to the publishers' own websites which are uncorrelated to the headline. The discrepancy between the headline and the destination content wastes online readers significant amount of time on contents of which they have no interest.

To address this problem, we propose CARE-BERT*: (**C**lickb**A**it Detecto**R** using using Self-attentiv**E** Network with **B**idirectional **E**ncoder **R**epresentations from **T**ransformers), which could effectively identify clickbaits utilizing S.O.T.A pre-training methods and self-attentive network. We approach this issue as a regression problem in our two parallel baseline models. By training on a large twitter posts corpus with annotations of their 'clickbaitness' on a scale of [0,1], we expect our model to be capable of capturing clickbait patterns in the headline.

Related Works

Attempts to detect clickbaits have been made by many online media agencies. Facebook, for instance, launched a war against clickbaits in 2017. Their main strategy was to utilize users feedback to identify and block domains that are notorious for producing clickbaits.

In the academia, initial work in this domain can be traced back to [4], which relies heavily on feature engineering and does not generalize well [20]. Chakraborty et al. [7] took the lead to explore machine learning models (mainly SVM) in clickbait detection in 2016 and create a browser extension 'Stop Clickbait' for deterring clickbaits, yet they called for future work devoted to improving the classification performances.

[19] crowdsourced a standard dataset of tweets (Webis 17) as a clickbait detection benchmark. [1] proposed the first neural netowrk based approach in this field on the Webis 17 dataset. They employed various recurrent neural network architectures to model sequential data and its dependencies, taking as its inputs a concatenation of the word and character-level embeddings of the headline. Their experiments yielded that bidirectional Long Short Term Memory (Bi-LSTM [23]) were much better at this task than previous SVM S.O.T.A by [7]. [24] built Bi-LSTMs to model each textual attribute of the post (post-text, target-title, target-paragraphs, target-description, target- keywords, post-time) available in the corpus [19], concatenating their outputs and feeding it to a fully connected layer to classify the post. Recently, attention mechanisms [2] become popular for various text classification tasks, utilizing this technique, [28] deployed a self-attentive bidirectional GRU to infer the importance of each tweet token and model the annotation distribution of headlines in the corpus, and achieved S.O.T.A on the Webis 17 dataset so far.

1. Changes since Project Proposal

1.1. Data Set

In our proposal, we mentioned two data sets: Webis Clickbait Corpus 2016 [21] and the data set from Chakraborty's work [7]. The first one preserves contextual information of a post while the second contains only the post titles.

After further exploration, we found Webis Clickbait Corpus 2017 [18], a more sophisticated version for Webis Clickbait Corpus 2016. This new dataset is extracted from 38,517 Twitter posts and provides the linked information of the posts as well. The data in this set is well-formed (much more standard than the 2016 version) and can be loaded with pandas package in python. We finally decided to use this dataset for our training and testing for its larger scale and clarity.

*Our codes are available at https://github.com/PeterQiu0516/CARE-BERT (incomplete at the time of writing yet)

1.2. Method

In the proposal, we proposed to use Support Vector Machine (SVM) as our model. However, prior studies showed that it does not fit this problem setting well where the task is to differentiate a short title and a long text based on prior studies [14, 15].

As a result, we conducted more in-depth research about the cutting-edge method in this domain. We found that Long Short Term Memory (LSTM [11], has variants like Bi-LSTM [23], GRU [8]) and Transformer [25] model are more advanced and powerful architecture for text similarity measurement, text classification and representation learning than a plain SVM, and has been proven to perform better on clickbait detection task. Both LSTM and Transformer are able to extract context information through Recurrent Neural Network (RNN) and attention-machanism respectively. And we decide to modifies our architecture to make use of these cutting-edge and effective models.

Moreover, pre-training methods which are based on transformer have revolutionized NLP over the past few years. Task-agnostic objectives such as autoregressive (GPT-2 [22], GPT-3 [6]) and masked language modeling (BERT [10]) have scaled across many orders of magnitude in compute, model capacity, and dataset, which leads to steady improvement in capabilities. Flagship systems like GPT-3 and BERT are now competitive across many tasks in NLP with bespoke models while requiring little to no dataset specific training data. As a result, we decide to utilize theses pre-training models for feature extraction from headlines and text, and apply fine-tuning to our downstream clickbait detection task.

1.3. Problem Formulation

We now formulate the problem as a supervised regression problem, that given the input title (and text for baseline model 1), we train our model on the dataset so that it could predict the clickbait score in the range of [0, 1] (to be consistent with labels). Then for binary classification analysis, we predict data with score > 0.5 to be clickbait and those with score ≤ 0.5 to be non-clickbait.

2. Data-preprocessing

2.1. Data Set

We perform experiments on the Webis 17 [18] dataset. [19] crowd-sourced the annotation of 38,517 tweets they had curated into various levels of their clickbait nature. These tweets contained the title and text of the article and also included supplementary information such as target description, target keywords and linked images. The data set is already split into train set (19,538 posts with 4761 being clickbaits and 14,777 non-clickbaits) and test set (18,979 posts). More detailed information about the dataset can be found in this website: Webis Clickbait Detection Challenge. Here we also included the structure of the dataset in the Table 1.

2.2. Data Selection

We focus on "postText", "targetParagraphs" and "truthMean" for each data. We excluded the media information since some of the posts do not have corresponding media annotations and thus it could hinder us from developing a model that can generalize well. For other information like post-id, target-keywords and post-timestamp, we think they are not highly related with whether a post is clickbait or not and we dispose them.

2.3. Data Cleaning

In our data cleaning process, we also decided it would be beneficial to exclude data that come with annotation (or label) of low confidence. The raw clickbait scores ('truthMean' label) are in a range of [0,1], as an average of 5 scores coming from 5 annotators, and we decided to exclude tweets with a mean clickbait score that deviate no more than 0.2 from 0.5, or in the range of [0.3, 0.7]. These annotations shows little confidence to be distinguished as either clickbait or not, and might confuse our model. After the cleaning process, we obtained a total of 12963 valid examples from training set.

2.4. Train/Validation Split

We divide the dataset in to a Training set of 11663 tweets and a Validation set of 1300 tweets. We use the validation set for preliminary performance evaluation and model selection and we report our results on the final Test set. The statistics of the dataset can be found in Table 2.

3. Methodology, Training & Evaluation

3.1. Word Embedding

Word embedding is a text feature extraction technique that encodes the meaning of the word into a vector, thus obtaining the representation of words for text analysis and enables computers to measure distances between words and measure similarity [12].

In this article, we used BERT [10] embedding, which provides provides bidirectional representations for text extracted from transformer. We choose this embedding because it has shown significant capacity of text feature extraction and achieved S.O.T.A. performance on many NLP downstream tasks. We will try other embedding method like GloVe [17] and Word2Vec [16] in the future if time permits.

Table 1. Webis 17 dataset structure [18]

Variable	Corresponding Data & Format
id	instance id
postTimestamp	⟨weekday⟩⟨month⟩⟨day⟩⟨hour⟩ : ⟨minute⟩ : ⟨second⟩⟨time offset⟩⟨year⟩
postText	⟨text of the post with links removed⟩
postMedia	⟨path to a file in the media archive⟩
targetTitle	⟨title of target article⟩
targetDescription	⟨description tag of target article⟩
targetKeywords	⟨keywords tag of target article⟩
targetParagraphs	⟨text of the ith paragraph in the target article⟩
targetCaptions	⟨caption of the ith image in the target article⟩
truthJudgments	clickbait scores provided by 5 annotators, fall in to 4 categories: 0 - "not clickbaiting", 0.33 - "slightly clickbaiting", 0.66 - "considerably clickbaiting", 1 - "heavily clickbaiting"
truthMean	mean of the truthJudgements
truthMedian	median of the truthJudgments
truthMode	if truthMean $>$ 0.5, 0 - non-clickbait; else, 1 - clickbait
truthClass	"non-clickbait" or "clickbait"

Table 2. Dataset statistics.

	#tweets	#click-baits	#non-clickbaits
Training	11663	2027	9636
Validation	1300	203	1097

3.2. Feature Engineering

We have built up a testing frame for feature engineering. As discussed in our proposal, we will explore features mentioned in Chakraborty's paper [7] and other features like word sentiment. In addition, experiment results in [26] suggests that "Character Sum", "Number of Dots", "Number of Easy Words", word "this", characters "?" and "@" could also have a powerful impact on the classification. Currently, we have finished the program frame for testing the effectiveness of each feature. We also tried testing features like article length for sanity check. And we are still working on the choice of features to extract and their relative influence on the clickbait classification problem.

Furthermore, if time permits, we will consider combining feature engineering and our full model together by providing both word embedding and extracted features to a full network and compare the performance with the raw model with ablation study.

3.3. Model architecture

We have constructed 2 baseline models at the time of writing:

1. Naive Cosine Similarity Checker with BERT: use BERT embedding to encode both the headline and text $x_{headline}, x_{text}$, use elementwise inverse exponential mapping $x' = e^{-x}$ to each test embedding so that each element in the embeding is strictly positive, and then perform a naive cosine similarity check and output the result $y = \frac{x'_{headline} \cdot x'_{text}}{||x'_{headline}||_2 \cdot ||x'_{text}||_2}$ (cosine similarity falls in range [0,1]).
2. headline Bi-LSTM with BERT: use BERT attention embedding to encode only the headline and feed into a Bi-LSTM followed by a fully connected layer with 1 output node, and then train the network supervisedly for regression (use sigmoid activation at last node so that output falls in range [0,1]).

We construct our first baseline model to be a naive cosine similarity checker. We directly calculate the cosine similarity of the embedding of the title and the text. The intuition behind the first model is that a clickback tweet should have lower correlation between the headline and the actual content than a non-clickback tweet.

Regarding our second baseline model, due to the limitations of computational (GPU) resources and the immense size of the BERT model, we have not finished fine-tuning and testing our target model, which is a fine-tuned BERT-based model with output layer to be a self-attentive network. As a result, we first analyze the representative capacity of BERT by feeding the embedding directly to a simple bidirectional LSTM (or Bi-LSTM) without fine-tuning on the whole BERT model for preliminary analysis.

3.4. Training

The first baseline model does not involve any training.

We train the second baseline model for 20 epochs. We use the Adam optimizer [13] with a starting learning rate of

$3e-4$ and decay the learning rate by a factor of 0.25 when the validation performance does not improve for 2 consecutive epochs (with minimum learning rate set to be $1e-6$). After preliminary data analysis we found that all headlines in the training set have an average number of token to be around 17.64, therefore we pad and truncate all headlines to have 20 tokens (with 1 [CLS] token and 1 [SEP token]) to be consistent. And we feed the embedding that corresponds to the [CLS] token which holds sentence-level classification representation into Bi-LSTM. We set the hidden dimension of the Bi-LSTM to be 50, which is of the same scale as number of tokens for each headline. We use Xavier initialization for weights in the final fully-connected layers. The model is trained for 12 hours on one Tesla T4 GPU hosted on Google Colab.

3.5. Evaluation

We are evaluating the performance of our model on 2 tasks: binary classficiation task and regression task, the evaluation metrics include:

- For the binary classification task, we used accuracy and F1 score as our evaluation metric, which is consistent with prior works ([28, 20, 24]) on this dataset for comparison.
- For the regression model, we used Mean Squared Error (MSE) as our metric, which is consistent with prior works ([28, 20, 24]) on this dataset for comparison.

4. Experiments

We perform the experiments on the pre-processed dataset. The results are shown in Table 3 and 4.

Our first baseline model, or Naive Cosine Text Similarity with BERT, involves no explicit training process. One peculiar observation is that the output of this baseline model is almost always around 0.5, which indicates little confidence in almost all predictions. As a result, the model only gets 26.9% accuracy on the dataset, even falls below a plain random classifier. The first baseline model demonstrates that:

1. Without fine-tuning BERT on our own corpus, the attention embedding is unfitting to our task.
2. The 26.9% accuracy might have something to do with the imbalanced training data that the raio of clickbait:non-clickbait is approximately 1:5.5.

Surprisingly, our second baseline model beats the previous S.O.T.A on both accuracy and MSE [28] on Webis 17 dataset with a simple Bi-LSTM structure even without fine-tuning on the raw BERT architecture, and this demonstrates that:

1. The BERT embedding is significantly powerful in the problem setting and provides salient representation of headlines.
2. The Bi-LSTM architecture fits this downstream task well, and we might experiment further with CNN, GRU or Transformer for comparison.
3. There might exist further space for improvement if we replace Bi-LSTM with a Transformer.

5. Future Work

Our first baseline model performs poorly, and while our second baseline model achieve S.O.T.A on accuracy and MSE, there are still rooms for improvement.

5.1. Word embedding

We plan to explore other word embedding such as the GloVe [17] embedding, or Word2Vec [16] embedding instead of BERT embedding, as they are pretrained on different corpus and may contain different distribution-specific property that could affect model performance.

5.2. Feature Engineering

We expect to use feature engineering to further extract features like text sentiment score, "Number of Dots", and other features. And finally if time permits we will try to incorporate these features into our full model.

5.3. Model architecture

1. There exists a large gap in length (or number of tokens) between titles and article texts, and this might lead to the failure in the text similarity method, which is inadequate for document pairs that have non-comparable lengths due to the lexical, contextual and the abstraction gaps between a concise title and a long document of rich details. To bridge this gap, we could possibly seek to S.O.T.A. [20] text summary method to preprocess the text so that title and text are of similar scale and then perform the similarity check.
2. While the full self-attention operation in BERT brings its success, it also serves as its bottleneck by limiting BERT's capability of processing long text due to the quadratic scale of resources consumption (computation and memory). As a caveat, BERT is limited to a maximum of 512 tokens, while the tokenized text in the dataset has an average of 791 and a maximum of 43357 tokens. For text that have significantly much more tokens than the threshold of BERT, simply head-truncation or tail-truncation may lead to in-negligible loss of information and other measures (concatenate single BERT representation of each 512 token block

Table 3. Model Performance Comparison (Classifcation)

	Accuracy	F1-score
Naive Cosine Text Similarity with BERT (ours)	26.9%	0.166
Headline Bi-LSTM Regression with BERT (ours)	**85.83%**	0.674
Feature Engineering S.O.T.A. [20]	83.24%	0.550
Concatenated NN Architecture [24]	74.00%	0.390
Headline multi-classification with self-attentive Bi-GRU (prev S.O.T.A) [28]	85.6%	**0.683**

Table 4. Model Performance Comparison (Regression)

	MSE
Naive Cosine Text Similarity with BERT (ours)	0.269
Headline Bi-LSTM Regression with BERT (ours)	**0.031**
Feature Engineering S.O.T.A. [20]	—
Concatenated NN Architecture [24]	0.045
Headline multi-classification with self-attentive Bi-GRU (prev S.O.T.A) [28]	0.033

together, or move to advanced transformer model for long text like Transformer-XL [9], LongFormer [3] and Big Bird [27]) should be considered.

3. Recently we have made use of BERT for word embedding and achieves S.O.T.A result on clickbait classification (accuracy) and regression (MSE) task on Webis 17 dataset, but the performance improvement is insignificant and there are still room for improvement. In future work we could (1) try to use other advanced architecture like GRU or transformer as the downstream regression layer, (2) further fine-tune on the raw BERT model which has proven to be promising in lots of NLP downstream tasks.

5.4. Data Preprocessing

As we can see from Table 2 that the dataset is imbalanced — there are approximately 5.5 times non-clickbaits training data than clickbait ones. We may seek to sampling approaches that help with imbalanced regression like Synthetic Minority Over-Sampling Technique for Regression with Gaussian Noise (SMOGN [5]) and compare with the baseline.

References

[1] Ankesh Anand, Tanmoy Chakraborty, and Noseong Park. We used neural networks to detect clickbaits: You won't believe what happened next! In Joemon M. Jose, Claudia Hauff, Ismail Sengör Altingövde, Dawei Song, Dyaa Albakour, Stuart N. K. Watt, and John Tait, editors, *Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8-13, 2017, Proceedings*, volume 10193 of *Lecture Notes in Computer Science*, pages 541–547, 2017. 1

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 1

[3] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020. 5

[4] Prakhar Biyani, Kostas Tsioutsiouliklis, and John Blackmer. "8 amazing secrets for getting more clicks": Detecting clickbaits in news streams using article informality. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 94–100. AAAI Press, 2016. 1

[5] Paula Branco, Luís Torgo, and Rita P. Ribeiro. SMOGN: a pre-processing approach for imbalanced regression. In *First International Workshop on Learning with Imbalanced Domains: Theory and Applications, LIDTA@PKDD/ECML 2017, 22 September 2017, Skopje, Macedonia*, volume 74 of *Proceedings of Machine Learning Research*, pages 36–50. PMLR, 2017. 5

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 2

[7] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. Stop clickbait: Detecting and preventing clickbaits in online news media. *CoRR*, abs/1610.09786, 2016. 1, 3

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn

encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 2

[9] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics, 2019. 5

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. 2

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. 2

[12] Dan Jurafsky and James H. Martin. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*. Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, 2009. 2

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 3

[14] Yang Liu, Cheng-Jie Sun, Lei Lin, and Xiaolong Wang. yigou: A semantic text similarity computing system based on svm. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 80–84, 2015. 2

[15] Prodromos Malakasiotis and Ion Androutsopoulos. Learning textual entailment using svms and string similarity measures. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 42–47, 2007. 2

[16] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. 2, 4

[17] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. 2, 4

[18] Martin Potthast, Tim Gollub, Matthias Hagen, and Benno Stein. The clickbait challenge 2017: Towards a regression model for clickbait strength. *CoRR*, abs/1812.10847, 2018. 1, 2, 3

[19] Martin Potthast, Tim Gollub, Kristof Komlossy, Sebastian Schuster, Matti Wiegmann, Erika Patricia Garces Fernandez, Matthias Hagen, and Benno Stein. Crowdsourcing a large corpus of clickbait on Twitter. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1498–1507, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics. 1, 2

[20] Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. Clickbait detection. In *European Conference on Information Retrieval*, pages 810–817. Springer, 2016. 1, 4, 5

[21] Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. Clickbait detection. In Nicola Ferro, Fabio Crestani, Marie-Francine Moens, Josiane Mothe, Fabrizio Silvestri, Giorgio Maria Di Nunzio, Claudia Hauff, and Gianmaria Silvello, editors, *Advances in Information Retrieval - 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings*, volume 9626 of *Lecture Notes in Computer Science*, pages 810–817. Springer, 2016. 1

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 2

[23] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45(11):2673–2681, 1997. 1, 2

[24] Philippe Thomas. Clickbait identification using neural networks. *CoRR*, abs/1710.08721, 2017. 1, 4, 5

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. 2

[26] Matti Wiegmann, Michael Völske, Benno Stein, Matthias Hagen, and Martin Potthast. Heuristic feature selection for clickbait detection. *CoRR*, abs/1802.01191, 2018. 3

[27] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 5

[28] Yiwei Zhou. Clickbait detection in tweets using self-attentive network. *CoRR*, abs/1710.05364, 2017. 1, 4, 5