Clickbait; Didn't Read: Clickbait Detection using Parallel Neural Networks

Peter Adelson[1], Sho Arora[2], and Jeff Hara[3]

Abstract— **Clickbaiting is the strategy of using intentionally misleading links, tweets, or social media posts to attract online viewership, and clickbait has been one method of disseminating misinformation on the internet. We propose a novel neural network model for determining how "clickbaity" an article is given its article and associated metadata. We achieve promising results with our best models scoring close to state-of-the-art models.**

I. INTRODUCTION

Clickbaiting is a growing phenomena on the internet, and it is defined as a method of exploiting cognitive biases to attract online viewership, that is, to attract "clicks." The articles behind clickbaits are usually uninformative, and besides contributing to an overall decline in journalistic integrity, clickbaits spread misinformation, often by making shocking implications, only to backtrack on those claims in the article.

Examples of clickbait titles are:

- "10 ways the expat life Is like a continual espresso buzz"
- "5 incredible Italian dishes you haven't tried before"
- "What India's microloan meltdown taught one entrepreneur"

Clickbaiting can come in many forms, like advertisements, videos, and such, but the problem is that websites are highly incentivized to publish clickbait articles because they are cheap to produce and can generate revenue. The main motivation in trying to identify clickbait is to filter out potential sources of misinformation on the internet.

Bauhaus University at Weimar published a dataset of clickbait articles and their associated metadata with a human-evaluated "clickbait-score" and organized a coding challenge to create the best clickbait classifier. We used submissions to this Clickbait Challenge that were publicly available as our baseline models and developed our own neural network model. The input is the first 100 words of the article, the headline, and the post text, which is the text that linked to the article. The dataset was hand-labeled with a clickbait score ranging from 0 to 1. We used linear regression (LR) and an SVM as baseline models, and we experimented with Convolutional Neural Networks (CNNs) and word embeddings (GLoVe).

Finally, we implemented a novel model we call Parallel Neural Networks (PNNs), which use an ensemble of CNNs to learn the relationships between headlines, post texts, and article texts. Our model was based on the observation that the "clickbait-iness" of article is some measure of how similar the post text is to its linked article.

II. RELATED WORK

A. Interdisciplinary Research

The earliest study of clickbait comes from psychology as the cognitive biases that clickbait exploits is known as the "curiosity gap." By using vague language, clickbait headlines withhold information to increase curiosity and increase the likelihood that someone will click on the link [5].

This has also been studied in linguistics. There have been studies on "listicles," or articles that are simply lists of things, that are sensational and "clickbaity" [7]. Additionally, "clickbaity" articles tend to have forward references like "this" or "these" [1]. These findings have identified telltale characteristics of clickbait articles, but a model that extracts only these features would not be robust. The features need to be more nuanced to avoid flagging non-clickbait articles.

B. Machine Learning Approaches

Recently, machine learning approaches to clickbait detection have been proposed. Potthast et al. (2016) published the first clickbait detection model that extracted features from the post text, the raw webpage, and various metadata and used LR, naive Bayes, and Random Forest regression (RF), achieving an Area Under the Curve of Receiver Operating Characteristic (ROC-AUC) of 0.79 [6]. However, these results came from using their top 1000 features, which makes for tedious, hand-written feature extraction. Their features included n-grams, sentiment scores, and grammatical patterns.

Cao et al. (2017) published a list of their top 60 clickbait features and, using an RF model as well, achieved an ROC-AUC of 0.745 and an F1 score of 0.61 [2]. These results are arguably better than Potthast et al.'s because the model is significantly less complex and therefore less prone to overfitting.

Grigorev (2017) presented a high-performance model using an ensemble of linear SVMs [4]. Multiple SVMs were trained on post text, keywords, captions, titles, and the articles themselves where the inputs were a Bag-of-Words word vectors (a summation of one-hot vectors). Then, the trained models were stacked using an RF. This method achieved a MSE of 0.0362. It also found that the most salient feature of clickbait is the post text. This was one of the best performing models in the Clickbait Challenge. We were influenced by the use of ensembling, and we hoped to get better results using word embeddings instead of Bag-of-Words vectors.

[1]`padelson@stanford.edu`
[2]`shoarora@stanford.edu`
[3]`jhara18@stanford.edu`

III. Dataset and Features

We used the dataset provided by http://www.clickbait-challenge.org/, which provides 19538 data points (4761 clickbait, 14777 non-clickbait) annotated by humans. Each annotator gave a score between 0 and 1, and the median of all judgments is taken to be the true value. Each data point had the article's title, content, and the text post accompanying the article online.

For baseline features, we used TF-IDF (term frequency - inverse document frequency) scores, features from Cao et al. [2], and pre-trained GloVe word vectors.

A. TF-IDF

TF-IDF represents the importance of a word in a given document by how many times it appears (term frequency), and regularizes it by how commonly it appears in all documents (inverse of document frequency). Let $s_w^{(d)}$ be the TF-IDF score of word w in document d. $tf_w^{(d)}$ is the frequency of word w in document d, and df_w is number of documents containing word w. For a corpus of D documents,

$$s_w^{(d)} = tf_w^{(d)} \times -log\frac{df_w}{|D|}. \tag{1}$$

We used TF-IDF scores only for words that occurred in at least 100 documents to limit the dimensionality of our features.

B. Top 60

Our other baseline feature set comes from Cao et al. [2]. In their Clickbait Challenge submission, they engineered and identified the sixty top features using feature selection. We used fifty of them in an additional baseline experiment. The part-of-speech tags are written in the shorthand used in the Penn Treebank Project. We used Spacy NLP to extract these features.

Rank	Feature	Rank	Feature
1	Number of NNP[1]	31	Count POS pattern DT[2]
2	Readability of target paragraphs (1)	32	Number of DT
3	**Number of tokens**	33	POS 2-gram NNP IN
4	Word length of post text	34	POS 3-gram IN NNP NNP
5	POS 2-gram NNP NNP	35	Number of POS[3]
6	Whether the post start with number	36	POS 2-gram IN NN
7	Average length of words in post	37	**Match between keywords and post**
8	Number of IN[4]	38	Number of ','
9	POS 2-gram NNP VBZ[5]	39	POS 2-gram NNP NNS[6]
10	POS 2-gram IN NNP	40	POS 2-gram IN JJ[7]
11	Length of the longest word in post text	41	POS 2-gram NNP POS
12	Number of WRB[8]	42	Number of WDT[9]
13	Count POS pattern WRB	43	Count POS pattern WDT
14	Number of NN[10]	44	POS 2-gram NN NN
15	Count POS pattern NN	45	POS 2-gram NN NNP
16	Whether post text start with 5W1H[11]	46	POS 2-gram NNP VBD[12]
17	Whether exist QM[13]	47	**Similarity between post and target paragraphs**
18	**Similarity between post and target title**	48	Count POS pattern RB[14]
19	Count POS pattern this/these NN	49	Number of RB
20	Count POS pattern PRP[15]	50	POS 3-gram NNP NNP NNP
21	Number of PRP	51	POS 3-gram NNP NNP NN
22	Number of VBZ	52	Readability of target paragraphs (2)
23	POS 3-gram NNP NNP VBZ	53	Number of RBS[16]
24	POS 2-gram NN IN	54	Number of VBN[17]
25	POS 3-gram NN IN NNP	55	POS 2-gram VBN IN
26	Ratio of stop words in post text	56	Whether exist NUMBER NP[18] VB[19]
27	POS 2-gram NNP .	57	POS 2-gram JJ NNP
28	POS 2-gram PRP VBP[20]	58	POS 3-gram NNP NN NN
29	Count POS pattern WP[21]	59	POS 2-gram DT NN
30	Number of WP	60	Whether exist EX[22]

Cao et al. suggest that these features perform better than n-grams features did because it reduced runtime significantly and reduced the variance of the model.

C. GloVe Word Vectors

GloVe vectors are vector representations of words created by co-occurrence statistics on a given corpus. Specifically, we used 300-dimensional GloVe 6B trained on Gigaword and Wikipedia for our deep learning models (https://nlp.stanford.edu/projects/glove/).

IV. Methods

For our baseline experiments, we used Logistic Regression and Support Vector Machine models along with TF-IDF and Top 60 features. For our deep learning experiments, we used convolutional neural networks, and a novel architecture based on convolutions and highway connections.

A. Logistic Regression

Logistic Regression is an algorithm that takes a linear combination of weights and features, and then applies a non-linear transformation to them (equation 2) to try to model a function.

$$g(z) = \frac{1}{1+e^{-z}} \tag{2}$$

Given the above definition for our transformation, the hypothesis function is:

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}} \tag{3}$$

B. Support Vector Machines

Support Vector Machines (SVMs) also aim to classify points in a dataset. However, instead of trying to estimate a function, SVMs try to separate the data based on their features, and use this separation to make decisions.

$$h_{w,b}(x) = g(w^T x + b) \tag{4}$$

Here, $g(z) = 1$ if $z \geq 0$ and $g(z) = -1$ otherwise.

C. Convolutional Neural Networks

Briefly, a convolution is a transformation takes a small weight matrix $\theta \in \mathbb{R}^{m\times n}$ and slides it over a larger target matrix X, collapsing the product between the two into an entry in a new matrix. Formally, a new entry a is defined as

$$a = \sum_{i=1}^{m}\sum_{j=1}^{n} \theta_{ij} X_{ij} \tag{5}$$

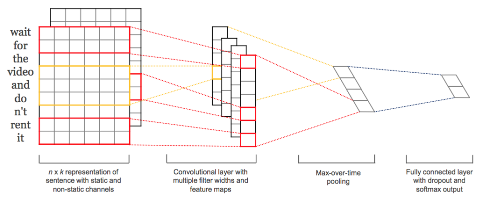
A Convolutional Neural Network (CNN) uses multiple weight matrices and aggregates the result into a new vectorized representation of the input. This new representation is then passed through a fully-connected linear layer for classification or regression.

In particular, we use a max-pooling layer in our experiments. The purpose of this layer is to aggregate the results from convolution by choosing the maximum values from various clusters.

Our final linear layer is defined as

$$y = Wx + b \tag{6}$$

where the weight W and bias b are trained, and the output y is used as a regression score.


wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

D. *Parallel Convolutional Highway Network (PNN)*

We decided to try a novel model on our data, based on the idea that clickbait is often characterized by an intriguing title and disappointing content. We use convolution to extract features from multiple aspects of a data point and combine them into a final regression score.

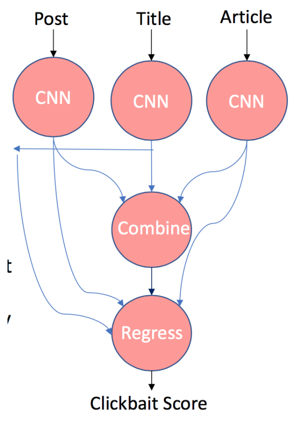
$$h_1 = CNN_1(input_1) \tag{7}$$

$$h_2 = CNN_2(input_2) \tag{8}$$

$$h_3 = tanh(W[h_1, h_2] + b) \tag{9}$$

$$out = W[h_1, h_2, h_3] + b \tag{10}$$

Extracting these features from different texts requires distinct convolutions run in parallel. (We remove the final linear layer from the CNN model we described previously.) We then combine these features into a new representation by concatenating them and running them through a linear layer with activation. Finally, we apply highway connections by generating our regression score from a linear layer applied on our convolutional features and our combined features.



V. Experiments

Asides from running the baselines, we also conducted experiments centered around CNNs and the PNN. Since Recurrent Neural Networks (RNNs) are commonly used with text applications, we considered using them but opted for CNNs because in initial testing, our RNNs were underperforming, possibly due to an insufficient amount of data. Upon viewing results, we also tried a condensed Parallel Convolutional Highway using only two CNNs on title and post, excluding article text. We did this because article text seemed poorly correlated with clickbait score. All CNNs had 4 layers with hidden dimensionality of 256 and word vector input dimensionality of 300. We tested lower dimensionality word embeddings (50 and 100) as well, but we found that they were underfitting.

As this is a regression problem, to judge results, we primary used mean squared error (MSE) using the true median score as the ground truth. However, as this is a regression between two distinct categories, we also report F1 score for the positive truth labels. F1 is the harmonic mean of the model's precision and recall.

For neural networks, we take the model with best dev performance and report its metrics from the test set.

Results are reported in table 1.

TABLE I
Model and Baseline Performance

Model	Train MSE	Train F1	Test MSE	Test F1
TF-IDF (LR)	101.189	0.908	43.008	0.445
Cao (LR)	5.485	0.482	5.382	0.501
Cao (SVM)	0.073	0.522	0.075	0.517
CNN (Text)	0.019	0.909	0.094	0.299
CNN (Title)	0.058	0.530	0.088	0.385
CNN (Post)	0.017	**0.913**	0.062	**0.650**
PNN 2 (Post/Title)	0.039	0.748	**0.061**	0.646
PNN 3	**0.014**	0.896	0.067	0.648

VI. Discussion

Overall, the parallel network architecture outperforms the baselines and approaches peak model performance. However, best performance was seen on a single CNN processing only the post text.

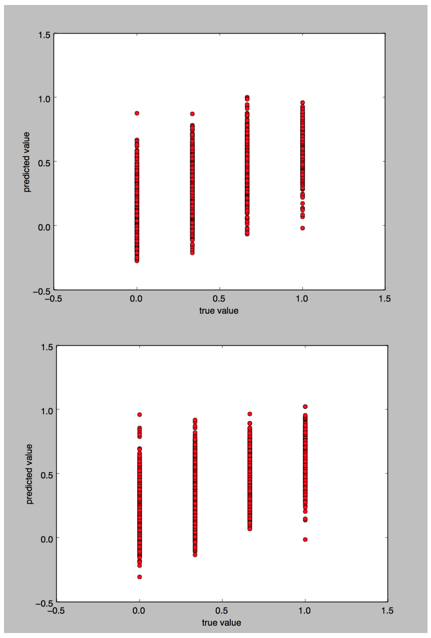
The baselines demonstrate somewhat effective performance, with the top 60 features outperforming TF-IDF. The logistic regression baselines have extremely high MSE. This follows from them making every classification either 0 or 1. Consequently, while F1 is on par with SVM, it has poor regression performance.

CNNs run on the text and title perform worse than the baselines, indicating that article text and title are much less important in labeling an article clickbait. The high-performance of CNN post indicates that the post is the most salient feature for determining clickbait. As the scores were generated by humans, this means that humans primarily look at the context of how the article was shared in determining whether or not it is clickbait. This indicates that clickbait is something of a social-media phenomenon, associated with posts trying to garner attention on a news feed. The article title and text influence clickbait classification much less.

Running on the article text produces a high degree of overfitting. Since article text is extremely lengthy, it provides ample flexibility for a model to fit the training set, but it does not generalize well.

These conclusions are supported by the performance of PNN 2, which used only title and post. It had less overfitting

Fig. 1. Clustering visualizations for PNN 3 (top) and CNN with Post (bottom)



and performed at the same level as PNN 3, indicating the article text is not a generalizable feature for clickbait scoring.

To get a better sense of how PNN compared to the CNN with post, figure 1 visualizes how the predicted scores compare with the true scores. In both cases, we see a high degree of spread in the different clusters but a clear trend that more clickbaity data points are more likely to be scored as such. CNN post has more outliers than PNN 3 on both ends. This follows from its less flexible model that has fewer features. Also some data points have no post text associated with them but different truth labels and hence have nothing to differentiate themselves for the CNN.

The means of the predictions for each of the clusters are given in table 2. Overall, they indicate a clear trend of effective prediction. As evidenced by the means, PNN 3 has more of tendency to under-predict the correct score. We believe this is due to the aforementioned overfitting caused by including article text, and to a less extent, article title.

Table 3 demonstrates a posts, their true clickbait score, and their predicted clickbait score. The PNN picks up on some structures that correlate with clickbait, such as questions and an invitation to learn more. However, the question in the informational headline about make the model overpredict its clickbait score. Less clickbaity posts present highly informational and specific headlines, as demonstrated with the Purple Rain post.

All neural networks exhibit some degree of overfitting. We believe that the relatively limited dataset of 20,000 points plays a large role in this, since this is a small dataset for typical deep learning applications.

Our best results are on par with, but fall short of, the top 5 results of the clickbait challenge, which range from 0.0033 to 0.0413 [4].

TABLE II
PREDICTION MEANS BY TRUTH MEDIAN CLUSTER

Model	0.00	0.333	0.667	1.000
CNN (Post)	0.181	0.327	0.478	0.623
PNet 3	0.111	0.271	0.433	0.589

TABLE III
SAMPLE POSTS AND PREDICTIONS

Post Text	True Score	PNet 3 Predicted Score
'Everything you need to know about the executive order on sanctuary cities:'	1.0	0.564
'Our complete guide to completing your taxes:'	0.666	0.633
'How much sugar is OK? A new paper adds to debate'	0.333	0.605
'Prince's Purple Rain Expanded Edition coming June 23 with unreleased tracks'	0.0	0.092

VII. CONCLUSION

We proposed a novel neural network model for determining how "clickbaity" an article is given its article and associated metadata. Our results were comparable to the top performers in the recent Clickbait Challenge, and we believe that some more tuning of our model can improve our performance easily. Our best model (lowest MSE) was our PNet that ensembled information about the post text and the article headline, and it achieved a test MSE of 0.061. For future work, the dataset came with images associated with the articles, and we believe that there may be salient information that could be extracted from the image. Some other papers have tried to use image-to-text algorithms, so there are still many interesting approaches to explore [3].

VIII. CONTRIBUTIONS

Peter Adelson

- Feature Extraction
- TFIDF Extraction
- Experiments
- Poster
- Paper

Sho Arora

- Harness Construction
- Neural Network Models
- Experiments
- Paper

Jeff Hara

- Baseline models and testing
- Feature extraction
- Word embedding
- Data Analysis
- Paper

REFERENCES

[1] Jonas Nygaard Blom and Kenneth Reinecke Hansen. "Click bait: Forward-reference as lure in online news headlines". In: *Journal of Pragmatics* 76 (2015), pp. 87–100.

[2] Xinyue Cao, Thai Le, and Jason Zhang. "Machine Learning Based Detection of Clickbait Posts in Social Media". In: *CoRR* abs/1710.01977 (2017). arXiv: `1710.01977`. URL: `http://arxiv.org/abs/1710.01977`.

[3] Aviad Elyashar, Jorge Bendahan, and Rami Puzis. "Detecting Clickbait in Online Social Media: You Won't Believe How We Did It". In: *CoRR* abs/1710.06699 (2017). arXiv: `1710.06699`. URL: `http://arxiv.org/abs/1710.06699`.

[4] Alexey Grigorev. "Identifying Clickbait Posts on Social Media with an Ensemble of Linear Models". In: *CoRR* abs/1710.00399 (2017). arXiv: `1710.00399`. URL: `http://arxiv.org/abs/1710.00399`.

[5] George Loewenstein. "The psychology of curiosity: A review and reinterpretation." In: *Psychological bulletin* 116.1 (1994), p. 75.

[6] Martin Potthast et al. "Clickbait Detection". In: *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings*. Ed. by Nicola Ferro et al. Cham: Springer International Publishing, 2016, pp. 810–817. ISBN: 978-3-319-30671-1. DOI: `10.1007/978-3-319-30671-1\_72`. URL: `https://doi.org/10.1007/978-3-319-30671-1\_72`.

[7] Bram Vijgen. "THE LISTICLE: AN EXPLORING RESEARCH ON AN INTERESTING SHAREABLE NEW MEDIA PHENOMENON." In: *Studia Universitatis Babes-Bolyai, Ephemerides* 59.1 (2014).