DeClickbaiting the News

Stanford CS224N Custom Project

Michael Elabd
Stanford University
mhelabd@cs.stanford.edu

Ronak Malde
Stanford University
rmalde@stanford.edu

Arjun Karanam
Stanford University
akaranam@stanford.edu

Abstract

News headlines have long been criticized for being “clickbait”, where the headline misleads the reader to draw their attention. While there have been various solutions to classify clickbait headlines [1, 2] and generating a new headline from scratch [3, 4], little has been done to both preserve the author’s intent while ensuring clarity of information. In this project, we used a pretrained encoder-decoder model, T5, and proposed two novel methods to achieve these goals. First, a custom loss function based on a pretrained BART summarizer, and second a loss penalty based on a BERT-based finteuned clickbait classifier. We found that while each model performs well on some success metris, using the BERT-based clickbait classifier with the finetuned T5 encoder-decoder model performed best.

1 Key Information

- Mentor: Kaili Huang

2 Introduction

News headlines have long been criticized for being “clickbait”, where the headline misleads the reader in order to draw their attention get them to, as the name implies, "click" on the article. More and more, we’re presented with examples of headlines that, through word choice, are intended to deceive, often employing methods such as: implying that something that’s false is actually true (or vice versa), promising an emotional payoff that never comes, and so on. These are problematic, especially when paired with people’s tendency to sometimes read headlines instead of reading through associated articles.

In response to this, many have tried to detect the presence of such clickbait headlines[2] and have been pretty successful. However, this only partially solves the problem. While it helps readers detect which headlines are clickbait and which ones are not, these classifiers don’t result in the creation of better headlines.

In this project, we seek to create a headline generator that takes in an article and its current title, and outputs a "good" headline. For the purposes of this project, we define a "good" headline as follows: a) a headline that accurately portrays the body of the text, b) a headline that is now "clickbait", and c) a headline that preserves the author’s intent. In order to do this, we aim to develop novel loss functions and penalties to emphasize the aforementioned attributes while training a generator model.

3 Related Work

There are have been two major research focuses within the area of news headlines - classification of clickbait, and generation of news headlines from scratch. We will also discuss some general natural language generation strategies that set the foundation for our approaches.

A "clickbait" headline can take many forms, as there are many strategies to mislead or entice the reader to read an article. Thus, there are a variety of strategies used in previous works to detect and

classify such a phenomenon in headlines. One paper, [5], focuses on detecting specific biased or subjective words that can appear in text or headlines. They attempt to accomplish this task using specific grammar rules and context with logistic regression, rather than deep learning. Other papers have focused on the overall meaning of the headline to assess how clickbait it is rather than individual words [1, 6], which is similar to our approach for addressing clickbait. These papers used pretrained embeddings to gain semantic understand, and further ran probabilstic models and deep learning models to achieve results in clickbait detection. More recently, research has found large success in using a large pretrained language model with transformers, such as BERT [7], and finetuning it for the specific task of clickbait detection [8]. We incorporate a similar approach for several of our novel additions, by finetuning BERT and other large pretrained language models.

There have also been several works for generating headlines from scratch in order to get an unbiased headline. One early example of specifically generating headlines was accomplished in 2015, with an LSTM encoder-decoder model with a simple attention mechanism [3]. For recently, research has focused on the fact that this task is somewhat similar to a summarization task, which has been well explored my large language models such as BART [9]. Thus, there are many papers that finetune on an existing summarization model to apply it to headline generation [10]. While we don't use a pretrained summarization model directly in our model architecture, it is an integral part of our novel loss function in this project.

The main focus of this project is to create a non-clickbait headline while also preserving the authors original wordsd as much as possible. We investigated various natural language processing strategies from otehr domains that could be useful for our project. One paper sought to neutralize words in a headline that might biased or subjective, by editing the words one by one [11]. This is slightly different than the task of clickbait, however, where individual words play less of a factor in the over clickbait of a headline.

4 Approach

This section details your approach(es) to the problem. For example, this is where you describe the architecture of your neural network(s), and any other key methods or algorithms.

Our task is to generate new headlines for an article. However, instead of just generating a headline, we wanted to find an optimization between keeping the author's intent, summarizing the article correctly, and avoiding clickbait. Thus, we created multiple pipelines to achieve each of these goals with the end goal of merging them into a unified pipeline. The goal of said pipeline is to achieve all three success metrics.

4.1 Pipelines

Retain Author Intent Most headline generators today [3, 4] look at the article body, and generate a headline. The assumption is that article headlines created by humans are the "correct" way of creating headlines. In other words, the headline is the "correct" headline for the article body. Thus, the model learns to mimic how humans create headlines. We see value in retaining the author's original intent, as it may include information or a style that can't be inferred from just the body. To accomplish this, we take both the article body and article title and use a fine-tuned T5 model to generate a title. Then we compute loss between this computed title and the article's original title using cross-entropy loss. Mathematically, we write

$$\mathcal{L}_t = -\sum_{v=1}^{V} y_{i,v} \log(p_{i,v})$$

where v is the index of the word in the vocab and y and p are matrices containing the article title labels and the model's probability distribution output respectively.

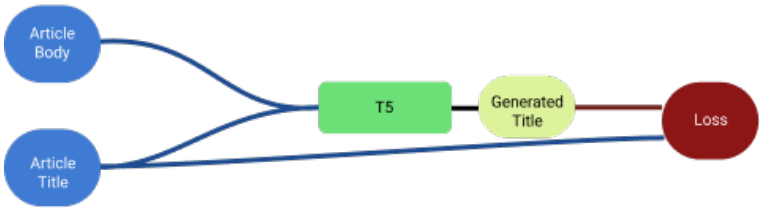

Figure 1: Finetuing T5 on headline Generation

Summarized Article Furthermore, a good headline is one that captures the article's content in a few word. Thus, a short article summary can also serve as a reference while generating an article headline. Especially when the article headline is clickbaity. To achieve this, we first create a summarizer pipeline to summarize our articles. We use a BART summarizer to generate an article summer given the article body. The goal of this pipeline is to get an objective short summary of the article. This short summary could be thought of as a secondary reference headline, and will be used as such.

Thus, we design two sub loss functions. The first compares the generated titles to the original article title and the second compares the generated title to the article summary. The idea behind having both losses is to ensure that we are not only generating headlines like humans but also that we generate headlines that effectively summarize the article. Moreover, this new loss function allows the model to deviate from the original title if it is clickbaity without accruing too much loss. This is because a clickbaity title would in theory be far from the article summary. Mathematically, we write

$$\mathcal{L}_{sl} = -\sum_{v=1}^{V} y_{i,v} \log(p_{i,v}) \tag{1}$$

where v is the index of the word in the vocab and y and p are matrices containing the summary labels and the model's probability distribution output respectively.With the T5 finetuning loss function being

$$\mathcal{L}_s = \mathcal{L}_{sl} + \mathcal{L}_t \tag{2}$$

if we are only concerned about retaining the author's intent and summarizing the article in our headline.

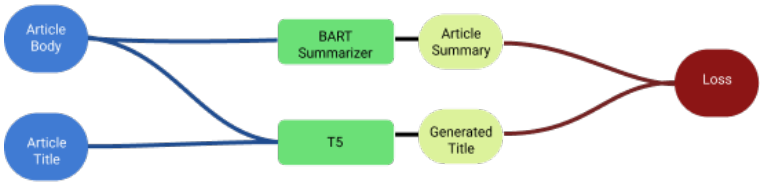

Figure 2: Summarization Loss on headline generation

Avoid Clickbait Another problem however, is still clickbait. Even if the generated title preserves author intent and summarizes the article. It could still be labelled as clickbaity. Thus, we would like to add a penality to the model's loss whenever its output article is indeed clickbaity.

However, we only have the labels of the actual article titles. We want to classify the generated output during training. So, we finetuned a BERT model with a linear layer and sigmoid function to classify clickbait on our training dataset. Table 1 shows the results.

We then used the classifier to check how clickbaity the generated article headline is. We used the classification probability along with a constant multiplier as a loss penalty for our model. Mathematically

Train	Dev	Test
86.60%	85.45%	85.32%

Table 1: Clickbait Classification on BERT

we write,

$$\mathcal{L}_{cl} = -p_c \cdot k$$

where p_c is the probability (as calculated by our classifier) that the generated title is clickbaity and k is a hyperparameter-tuned constant. With the T5 finetuning loss function being

$$\mathcal{L}_c = \mathcal{L}_{cl} + \mathcal{L}_{at}$$

if we are only concerned about retaining the author's intent and avoiding clickbait in our headline.

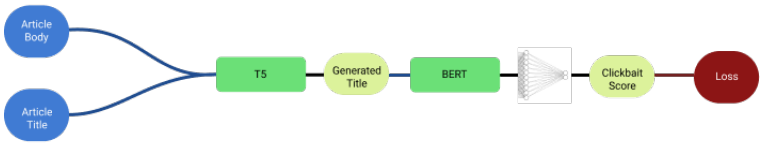

Figure 3: Clickbait penalty Loss on headline generation

4.2 Model

We combine all three pipelines to form the final model. Table 4 shows how loss propagates through out the model to fine tune the T5 headline generator network, where the final loss is the summation of summarization loss, author intent loss, and the clickbait penalty. In the experiment section below, we will investigate how adding each loss affects the model's finetuning and its output.

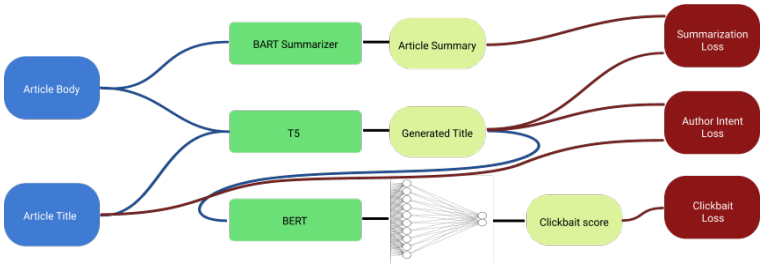

Figure 4: Model Design

5 Experiments

5.1 Data

All models were trained using the Webis-Clickbait-17 dataset [12]. The dataset contains a total of 40976 labelled articles along with 80013 unlabelled articles. These articles are published between November 2015 and June 2017 and are only from United States news outlets. Each article is bucketed into 4 categories of the headline's "clickbait score", judged by human reviewers. Webis defined clickbait as a headline that is designed to entice its readers into clicking an accompanying link i.e. something unnamed is referred to, some emotional reaction is promised, some lack of knowledge is ascribed, some authority is claimed, etc. We split the dataset 80/10/10 into train, dev, and test respectively.

We used the train dataset in two ways. First, we used all the articles and corresponding headlines that were not clickbait, in order to train the T5 model. Second, we used the train dataset to finetune the

BERT classifier, with the input being a merged sequence of body and title, and the labels being the clickbait score.

We used the dev dataset in order to tune hyperparameters such as learning rate, AdamW betas, and warmup steps. We used the test dataset to obtain metrics for model performance.

All inputs for the models were tokenized by the pretrained tokenizers for the respective models (T5, BERT, and BART), and were truncated and padded as needed for the model sizes. For the BERT classifier, we constructed the inputs as a single sequence in the form Article Body, <Seperator Token>, Article Title, <Padding Token> (until block size).

5.2 Evaluation method

BLEU Score Since we want to preserve the author's intent and restrict modifications on the headline, we examine the BLEU Score between the generated title and the real title [13]. Ideally, our model would generate titles that are close in proximity, thus receiving a high BLEU score.

ROUGE Score Moreover, our model's success is defined by how well it summarizes the body of the article [14]. Thus, we compute the ROUGE score between the generated title and the article body. Moreover, we compute the ROUGE score between the real title and article body as a baseline.

Cosine Similarity Finally, we compute Cosine similarity between the embeddings of the generated headline and the original headline. This shows how well the generated headline retains the meaning of the original headline. We evaluate the model on both SpaCy and BERT embeddings of the headlines [15].

Our goal is to have our model get both a high ROUGE score with the text body and high embedding cosine similarity with the original headline, so that the headline is more representative of the document (not clickbait), but remains similar to the author's initial headline.

5.3 Experimental details

As mentioned in our approach, for our project we trained 4 different models: Headline generation using T5 fine-tuned on the Webis Clickbait dataset, Headline generation using a finetuned T5 with our summarization loss, Headline generation using a finetuned T5 with our classifier penalty, and Headline generation using a finetuned T5 with both our summarization loss and our classifier penalty. We ran each of these models for 5 epochs, as this is when the loss curve begin to stagnate and wouldn't descend any further. While the models were all run on the same amount of Train data and for the same length, the actual training time vastly differed. This is namely because the the summarization + classifier model had to compute these additional losses on every batch. The models that used the classifier loss took especially long, mainly because each example needed to be fed through our BERT classifier. On the lower end, our finetuned T5 model took 30 minutes to train, while our summarization loss + classifier penalty model took upwards of 3 hours to train.

Since we were acting on an already finetuned model, we were fairly aggressive with our learning rate, using a a rate of 0.01. Additionally, we used an Adamw optimizer (which is much like the Adam optimizer, but with an improved weight decay) with betas of 0.7 and 0.8, meaning that the models go through weight decay faster as well. We used a weight decay of 0.0001 for finetuning.

5.4 Results

Here, we show how our model performed on both the train and test datasets, according to 5 metrics - a BLEU score, Cosine Similarity (Embedded and calculated by the Spacy library), another Cosine Similarity (Embedded and calculated by the SBert), Rogue 1, and Rogue 2.

5.4.1 BLEU Analysis

For the BLEU metric, we see that that the normal T5 model performs worst. Moreover, we see that all models perform about the same except for the model with classification penalty loss. This is quiet surprising as the model that only optimizes for mimicking the original headline performs worse. The

Method	BLEU	Cosine Sim (Spacy)	Cosine Sim (SBERT)	Rouge 1	Rouge 2
T5: Author Intent Loss	0.026	0.57	0.27	0.033	0.0060
T5: Author Intent Loss + Summarization Loss	0.027	0.62	0.28	0.038	0.0067
T5: Author Intent Loss + Classification Penalty	**0.029**	**0.63**	**0.31**	**0.044**	**0.0084**
T5: Author Intent Loss + Summarization Loss + Classification Penalty	0.025	0.62	0.28	0.037	0.0058

Figure 5: Train Metrics on different Loss functions

Method	BLEU	Cosine Sim (Spacy)	Cosine Sim (SBERT)	Rouge 1	Rouge 2
T5: Author Intent Loss	0.023	0.575	0.282	0.035	0.0077
T5: Author Intent Loss + Summarization Loss	0.025	**0.647**	0.297	0.040	0.0068
T5: Author Intent Loss + Classification Penality	**0.034**	0.642	**0.306**	**0.041**	**0.008**
T5: Author Intent Loss + Summarization Loss + Classification Penality	0.024	0.641	0.243	.0267	0.004

Figure 6: Test Metrics on different Loss functions

reason the classification penalty performs better could be because it helps the model converge on a solution space with less clickbaity titles.

5.4.2 Cosine Similarity Analysis

The cosine similarity scores followed a trend where both the Summarization Loss Model and the Classifier Penalty Loss Model outperform the basic Author Intent model. But, the combination of the two (the Summarization Loss + Classifier Penalty Loss), does worse than either the Summarization Loss or Classifier Penalty Loss models on their own. On unexpected finding is the fact that one model (the Summarization Loss Model) performs better on one cosine similarity metric (Spacy Cosine Similarity), while another model (the Classifier Penalty Model) performs better on the other cosine similarity metric. Since they both perform the the cosine similairty itself in the same way, and just differ in their embeddings. The exact reason for this is unclear. Upon further investigation, it seems as though the SBERT Cosine Similarity is more accurate than Spacy's, though the the overall phenomena surrounding this is unclear.

5.4.3 ROGUE Similarity Analysis

Similar to the cosine similiarty trends, we see that both the Summarization Loss Model and the Classifier Penalty Loss Model outperform the basic Author Intent model on the ROUGE score. Moreover, we see that that the classification Penalty Loss Model performs the best. Moreover, again like the cosine similarity, the combination of both (the Summarization Loss + Classifier Penalty Loss) performs very badly. A possible reason for this could be because the classification penalty encourages the model to search in a more confined subspace with less clickbait. We also think that the combination of both models does not work well because the model has a hard time converging.

5.4.4 Overall

Overall, based on the models we trained and the metrics we evaluated by, we can say that the Classifier Loss Model performs the best. Then comes the Summarization Loss Model, followed by, surprisingly, the combined Summarization Loss Model and Classifier Penalty Model, and finally, the Author Intent Loss Model. The fact that the Author Intent Loss Model performed the worst makes sense - while preserving author intent is nice, there is no guarantee that when writing the headline, the authors optimized for factors (such as accuracy to the article body) that the above metrics evaluate. What is surprising is that our ensemble Summarization and Classifier Loss Model did worse than its component parts. Further testing is required to find out exactly why this is, but our current hypothesis is that these two loss metrics might be acting in an adversarial manner. For example, a generated headline might be classified as having high cosine similarity (it's highly related to the article body), but is classified as clickbait. There are various reasons this might happen (the generated title uses

phrases like "You'll never guess", or emotion words, etc). A takeaway here might be that having too many loss components at once might confuse the model more than it actually helps it.

6 Analysis

Overall, the models performed well in reducing the amount of clickbait in headlines, and instead making a title that accurately describes the article. Both the summarization loss and the classification penalty resulted in headlines that better summarized the original document. There were some interesting behaviors we found when analyzing the generated data. One was that the model would deem some aspect of the headline "clickbait" when they might necessarily be. For example there was an article titled, "Boat Race: unexploded bomb found near starting line". The generated title was "police were called after a member of the public spotted second world war bomb near Putney". The generated title is less sensational, but it also removed any mention of the boat race, perhaps attributing the boat race to be sensational not part of the story. However, the story itself was largely focused on the boat race so it probably should have been included in the story.

Another interesting behaviour that we saw across different models was providing statistics in the title. For instance, in an article talking about french jobless claims declining, the generated title by the Summarizer Loss Model is "the number of people actively looking for work fell 0.9 percent, or 31,800, last month". We see this behaviour of citing statistics across all 4 models. In the Classification Loss Model we see the model generating the headline "60,000 people across eight time zones took to the streets to back navalny's campaign" for an article about Russian elections. We can see that the model often identifies the statistics as good summarizers or good headlines for the article. Moreover, We think that statistics convey information about the content of the article pretty well.

However, we see that the model misses some important information. In the first example, we see that it cites the statistic without enough information to know who the statistic pertains to.

Additionally, some of our outputs were either worded strangely, were gramatically incorrect, or contained an unfinished sentence. For example, one generated title by the model with summary loss and classification penalty was, "the government's crackdown on meat shops in the northern state of Uttar Pradesh has left many". Because the title might be encouraged to be shorter on general, it might prevent it from conveying all the information it needs to. For future work, we can experiment with our encoding process or model parameters, or enforce grammatical correctness for the title.

7 Conclusion

Our results show promising approaches towards generating non-clickbait headlines that still preserve the author's original intent.

There is still a lot of room to improve our existing architectures. We can further tune our hyperparameters like learning rate, AdamW optimizer parameters, and weight decay, using parameter grid search. Additionally, we would like to make our metrics more robust to assess how well the headline achieves our goals. ROUGE scores seem inadequate because they only calculate n-grams and not some higher-level meaning, and it might be useful to create a large-scale human scoring system, similar to how the dataset was constructed. This could also help identify specific edge cases with outputs. One future area for these methods could be style transfer of generated headlines, as an abstraction of our current approach. This could be used to not only unclickbait a headline, but to make a headline more funny, or more academic. These findings are a great first step, and show some promising results to be used for declickbaiting headlines.

References

[1] Sawinder Kaur, Parteek Kumar, and Ponnurangam Kumaraguru. Detecting clickbaits using two-phase hybrid CNN-LSTM biterm model. *Expert Systems with Applications*, 151:113350, August 2020.

[2] Ankesh Anand, Tanmoy Chakraborty, and Noseong Park. We used neural networks to detect clickbaits: You won't believe what happened next!

[3] Konstantin Lopyrev. Generating News Headlines with Recurrent Neural Networks. *arXiv:1512.01712 [cs]*, December 2015. arXiv: 1512.01712.

[4] Daniil Gavrilov, Pavel Kalaidin, and Valentin Malykh. Self-Attentive Model for Headline Generation. *arXiv:1901.07786 [cs]*, January 2019. arXiv: 1901.07786.

[5] Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. Linguistic models for analyzing and detecting biased language. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1659, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[6] Abinash Pujahari and Dilip Singh Sisodia. Clickbait detection using multiple categorisation techniques. *Journal of Information Science*, 47(1):118–128, February 2021.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[8] Muhammad N. Fakhruzzaman, Saidah Z. Jannah, Ratih A. Ningrum, and Indah Fahmiyah. Clickbait headline detection in indonesian news sites using multilingual bidirectional encoder representations from transformers (M-BERT). *CoRR*, abs/2102.01497, 2021.

[9] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[10] Yisong Chen and Qing Song. News text summarization method based on bart-textrank model. In *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, volume 5, pages 2005–2010, 2021.

[11] Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. Automatically Neutralizing Subjective Bias in Text. *arXiv:1911.09709 [cs]*, December 2019. arXiv: 1911.09709.

[12] Potthast, Martin, Gollub, Tim, Wiegmann, Matti, Stein, Benno, Hagen, Matthias, Komlossy, Kristof, Schuster, Sebstian, and Fernandez, Erika P. Garces. Webis Clickbait Corpus 2017 (Webis-Clickbait-17), June 2018. Type: dataset.

[13] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, USA, July 2002. Association for Computational Linguistics.

[14] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. pages 74–81, July 2004.

[15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]*, September 2013. arXiv: 1301.3781.