Can these simple methods solve your clickbait issues?: comparing approaches to handle long context in question answering models for clickbait spoiler generation

Darija Filipović

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
darija.filipovic@fer.hr

Abstract

Everything is trying to catch our attention, including titles of posts and articles. The purpose of this paper is to examine how well some existing question answering models, either in their original form or fine-tuned on the Webis Clickbait Spoiling Corpus 2022, are able to spoil clickbait based on the content of the article and the article title. Since the full articles are longer than the maximum length that the models can process, the author compares four different approaches to handling the long context. The filtered aggregation approach performs best across the tested models.

1. Introduction

Clicks have become a digital currency - the livelihood of news portals depends on it. Although it has been associated with lower-quality news outlets, even reputable ones utilise the technique nowadays while fighting for our attention. Clickbait creates an information need the reader didn't know they had and the only way to satisfy it used to be reading the article. With the rise of clickbait usage (Hurst2016), digital media vigilantes spoiling clickbait articles, such as SavedYouAClick started appearing, but as the global amount of content grows, so does the amount of clickbait, and a few real people can't solve the problem. Because of that, the need for an automated solution rises.

Automated clickbait detection systems exist to curb the amount of clickbait on social media, but don't satisfy the reader's information need created by the article title or post text. Browsers such as Arc (Good Marketing Club2023) started implementing summaries on hover while in the browser, but not while viewing websites. One approach to the solution is the SemEval 2023 competition which aims to solve two tasks: classify which type of answer the clickbait article has and identify the spoiler itself for the clickbait article.

The goal of this paper is to see how some approaches to question answering do at the spoiler generation task. First I'll go through related work and existing approaches. Then I'll describe the dataset and models used in the experiments, how the datasets were pre-processed and then discuss the results in the analysis.

2. Related work

It's still not clear what kind of existing NLP task clickbait spoiler generation falls into since clickbait can appear as a question as well as full sentences. Question answering is chosen in this paper because of the relatively small size of the models as well as easy data preprocessing. Other more robust approaches include generating spoilers with an ensemble of LLM-s (Woźny and Lango2023) which would require more resources. The articles are quite long so it's important to think of ways of feeding the whole article to the model or finding the part where the answer is. One way to do it is by first summarizing the article (Yeh et al.2006), but we risk losing important information. Another way is to first do passage retrieval (Liello et al.2022), but the model still needs to be able to process a lengthy article. Other approaches include Chain of Agents (Zhang et al.2024).

3. Dataset and models

3.1. Dataset

The dataset used for this project is the Webis Clickbait Spoiling Corpus 2022 (Hagen et al.2023). It contains 5,000 spoiled clickbait posts in English crawled from Facebook, Reddit, and Twitter from creators/accounts such as SavedYouAClick, HuffPoSpoilers, Stop Clickbait - Lifestyle divided into a train set with 3,200 examples, a validation set with 800 examples and a test set with 1000 examples. The dataset includes selected target paragraphs of the original article that serve as a context, post text, article title, the spoiler extracted from the text, the position of the spoiler(s) within the context and a tag for each example. The tag describes whether the spoiler is a single sentence, a passage within the context or a multi-part text within the provided context. Some examples have a human generated spoiler as well, but since they are incomplete, they won't be used in this paper.

Since finding multi-part spoilers would require a vastly different approach to processing the examples and training the models, especially for the question answering models, all the examples with a "multi-part" tag get filtered out.

3.2. Models

Three question answering models are chosen for the experiments. The models were chosen based on pre-training in English, the number of downloads and trending status within the question answering category on the Hugging Face Models website (hug2023).

deepset/roberta-base-squad2(deepset2020a) is the roBERTa base model pretrained on the SQuAD2.0 dataset. It was trained on question answer pairs including unanswerable questions.

deepset/tinyroberta-squad2(deepset2020b) is a distilled version of deepset/roberta-base-squad2, chosen as it might be less computationally intense.

timpal0l/mdeberta-v3-base-squad2 is based on deBERTa V3 and has also been pretrained on SQuAD2.0 for question answering.

4. Experiment setup

Three different pre-trained models were set up for the extractive question answering task and evaluated their performance as-is and after fine-tuning using the truncation, remove-long-entries and aggregation approaches. Each version was evaluated on 4 metrics: precision, recall, F1 score and exact match.

4.1. Dataset preprocessing

First, the examples with the multi-part tag were filtered out and only the id, targetTitle i.e. question, targetParagraphs i.e. context and spoiler i.e. answers columns were kept. The dataset came with entries about the position of the answer within the context, but since the context was divided into strings which represented paragraphs of the original text, and models need the position of the answer start and end tokens, new start and end positions were calculated based on the offset mapping obtained after tokenizing the questions and contexts.

4.2. Approaches to handling long context

Since the combined article paragraphs for one article were usually longer than a model could process, four different approaches to preparing the data were chosen to fit the models' maximum lengths.

In the truncated approach, question context pairs were fed to the model as a string and truncated to the maximum length a model allowed. This way, a lot of the context was lost and with it the position of the answer.

In the remove-long-entries approach, all entries in the datasets which were too long for the models were removed before tokenization and truncated them further after tokenizatin until they fit the desired maximum length.

In the aggregation approach, the model makes a prediction for every paragraph in the dataset. The dataset was flattened so there were multiple entries with the same id, question and answer, but a different context, where the context was one paragraph from the original list of paragraphs for each article. After finding the most likely prediction for a start and end pair in each paragraph, predictions for the same initial article were aggregated based on the highest likelihood. Inside this approach, two versions of the datasets were created - one with all the paragraphs and one without the paragraphs where the answer isn't mentioned so the dataset could be more balanced. This is important for the train set so the model doesn't predict the first index as the start and end position but actually tries to find the answer in the context.

5. Analysis

5.1. Cutoff approach

Because of the lack of full context, the models in the cutoff variant couldn't find the answer as it wasn't present in the context. We still see an improvement when trained. All the models in the cutoff variant performed poorly with a maximum exact match metric of 0.1212.

5.2. Remove long entries approach

In the remove long variants, the models found the exact solution on their own in less than a quarter of examples. After fine-tuning, there was an increase in performance for the roBERTa models, but not for deBERTa.

5.3. Aggregated approach

Results for a fine-tuned deBERTa are omitted because it required too many resources to train. In the aggregated approach, a significant difference is present both between the unfiltered and filtered approach and between pre-trained and fine-tuned versions. The unfiltered approach sees a great improvement after fine-tuning and although the metrics on the test set seem promising, they occured because of an imbalance in the dataset. The dataset was divided into paragraphs and the answer had to be predicted on each paragraph. Hence, a large portion of entries in the test set have a default start and end position as the first index. The train set was divided in the same way as the test set, and the model learned that predicting the first index is desirable. The filtered approach where paragraphs with the default start and end label were remove from all datasets bore more reasonable results in respect to comparing the results before and after fine-tuning. It still performs too poorly to be considered in real life use.

No statistical testing was done because of a visible change in performance between the model versions as well as because of time constraints.

6. Conclusion

Clickbait won't stop any time soon. One way to keep our concentration on digital content we actually want to consume is automatized spoiler generation. Question answering models don't fit the task of spoiler generation perfectly, but can certainly be used with limited capability. None of the model variations perform well enough for it to make sense to use them on their own. Future work includes improving the capability of the tested models with a sliding window approach or by including them as one component within a larger model which divides specific articles and titles into different types suited for different NLP tasks.

Acknowledgements

I would like to thank Ana Barić and Jan Šnajder for being understanging and alleviating pressure during the final exam season and Ivan for the emotional support. I thank Pipore for selling 1kg bags of mate tea and Overleaf for hiding the results table below.

References

deepset. 2020a. deepset/roberta-base-squad2.

deepset. 2020b. deepset/tinyroberta-squad2.

Good Marketing Club. 2023. Ai workflow - improve your research efficiency ft. arc browser, May.

Matthias Hagen, Maik Fröbe, Artur Jurk, and Martin Potthast. 2023. Webis clickbait spoiling corpus 2022, July.

Model Name	Fine-Tuned	Approach	Precision	Recall	F1	Exact Match
deepset/roberta-base-squad2	No	Truncation	0.0632	0.0419	0.0504	0.1212
deepset/tinyroberta-squad2	No	Truncation	0.0968	0.0144	0.0250	0.0994
timpal0l/mdeberta-v3-base-squad2	No	Truncation	0.0241	0.0551	0.0336	0.0667
deepset/roberta-base-squad2	Yes	Truncation	0.1495	0.1876	0.1664	0.2461
deepset/tinyroberta-squad2	Yes	Truncation	0.1495	0.1876	0.1664	0.2461
timpal0l/mdeberta-v3-base-squad2	Yes	Truncation	0.1182	0.1986	0.1482	0.2121
deepset/roberta-base-squad2	No	Remove-Long	0.1930	0.0552	0.0858	0.2083
deepset/tinyroberta-squad2	No	Remove-Long	0.1163	0.0251	0.0413	0.1667
timpal0l/mdeberta-v3-base-squad2	No	Remove-Long	0.2097	0.2542	0.2298	0.2494
deepset/roberta-base-squad2	Yes	Remove-Long	0.3949	0.3645	0.3791	0.3333
deepset/tinyroberta-squad2	Yes	Remove-Long	0.1106	0.1505	0.1275	0.2083
timpal0l/mdeberta-v3-base-squad2	Yes	Remove-Long	0.1863	0.2355	0.2080	0.2425
deepset/roberta-base-squad2	No	Unfiltered Aggr	0.2405	0.2286	0.2344	0.6194
deepset/tinyroberta-squad2	No	Unfiltered Aggr	0.1150	0.2468	0.1569	0.1540
timpal0l/mdeberta-v3-base-squad2	No	Unfiltered Aggr	0.2674	0.2749	0.2711	0.5806
deepset/roberta-base-squad2	Yes	Unfiltered Aggr	0.6082	0.5141	0.5572	0.8133
deepset/tinyroberta-squad2	Yes	Unfiltered Aggr	0.7064	0.3254	0.4456	0.8012
deepset/roberta-base-squad2	No	Filtered Aggr	0.1683	0.0519	0.0793	0.1009
deepset/tinyroberta-squad2	No	Filtered Aggr	0.1712	0.0322	0.0542	0.0523
timpal0l/mdeberta-v3-base-squad2	No	Filtered Aggr	0.1855	0.0858	0.1174	0.1245
deepset/roberta-base-squad2	Yes	Filtered Aggr	0.3607	0.3314	0.3455	0.3375
deepset/tinyroberta-squad2	Yes	Filtered Aggr	0.3282	0.3073	0.3174	0.2802

Table 1: Model Performance on Test Dataset

2023. Models - hugging face. Accessed: [30.6.2024.].

Nathan Hurst. 2016. To clickbait or not to clickbait? : an examination of clickbait headline effects on source credibility. Master's thesis, University of Missouri-Columbia. Creative Commons Attribution-NonCommercial-NoDerivs 3.0 License.

Luca Di Liello, Siddhant Garg, Luca Soldaini, and Alessandro Moschitti. 2022. Pre-training transformer models with sentence-level objectives for answer sentence selection.

Mateusz Woźny and Mateusz Lango. 2023. Generating clickbait spoilers with an ensemble of large language models. In *Proceedings of the 16th International Natural Language Generation Conference*. Association for Computational Linguistics.

Jen-Yuan Yeh, Hao-Ren Ke, and Wei-Pang Yang. 2006. Summarizing relevant information for question-answering using hybrid relevance analysis and surface feature salience. 3, 12.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Ö. Arik. 2024. Chain of agents: Large language models collaborating on long-context tasks.