Clickbait Spoiler Generation
Final Project — LIGN-167

Rajasvi Vinayak Sharma
(PID: A59012988)
University of California - San Diego
rvsharma@ucsd.edu

December 5th 2022

1 Introduction

Clickbait spoiling aims at generating short texts that satisfy the curiosity induced by clickbait posts generally posted on Reddit, Twitter, and Facebook. This project is a derivative of the Clickbait Challenge [Maik Frobe(2022)] at SemEval 2023 and aims to solve two subtasks from the challenge i.e. spoiler type classification (phrase, passage,multi-line) followed by spoiler generation.

Clickbaits often don't contain enough relevant information and meant to have attractive titles to draw user's attention. They often are used as source to show advertisement or convey something obvious. Generally, clickbait spoilers can be of 3 types: phrase, passage, and multi-line based. As seen from example, all 3 spoiler types have different text structure, length etc. which demands that we have different methods for generating them.

Figure 1: Twitter Clickbait examples

In this project I try to replicate the techniques mentioned in research paper (by organizers of shared-task challenge [Hagen and Fröbe(2022)]) by using Question Answering and Passage Retrievel ranking models to generate spoilers. There exists a code [Fröbe(2022)] for the original paper but it's only used a reference as I tried writing my own code for clarity, understanding and experimentation purposes. Please note that there doesn't exist any starter code for the Clickbait challenge though unlike usual Kaggle competitions. Similar to original paper, I have not yet found method for generating multi-line spoiler type and current work only covers phrase and passage spoiler generation.

I propose a 2 stage approach where firstly a classification model is used to identify spoiler type then we use appropriate spoiler generating model. Main point being that we use separate models for different spoiler types i.e. one for phrase and another for passage type after classification stage. Primarily use Question Answering and Passage Retrieval/Ranking models for spoiler generation. I fine-tuned QA models like deberta, roberta, & distilbert from Hugging Face and did a comparative evaluation study in the end. For evaluation, I have used BLEU-4, exact match and F-1 score for QA models and accuracy for classification models.

2 Dataset

The dataset [Hagen et al.(2022)Hagen, Fröbe, Jurk, and Pott contains the clickbait posts and manually cleaned versions of the linked documents, and

extracted spoilers for each clickbait post (the dataset was constructed and published in the corresponding paper). Additionally, the spoilers are categorized into three types: short phrase spoilers, longer passage spoilers, and multiple non-consecutive pieces of text.

Dataset contains 3,200 posts for training and 800 posts for validation. As a part of competition, they have also provided test set without any spoiler type labels or generated spoilers. For subtask 1, I used the entire dataset, but for subtask 2 I only used phrase and passage training subset.

The data comes in JSON Lines format (.jsonl) where each line contains a clickbait post and the manually cleaned version of the linked document. As mentioned the task is to classify the spoiler type needed (sub task 1), and generate the spoiler (sub task 2).

For each datapoint in the training and validation dataset, the following fields are available:

- uuid: The uuid of the dataset entry.
- postText: The text of the clickbait post which is to be spoiled.
- targetParagraphs: The main content of the linked web page to classify the spoiler type (task 1) and to generate the spoiler (task 2). Consists of the paragraphs of manually extracted main content
- targetTitle: The title of the linked web page to classify the spoiler type (task 1) and to generate the spoiler (task 2).
- targetUrl: The URL of the linked web page. humanSpoiler: The human generated spoiler (abstractive) for the clickbait post from the linked web page. This field is only available in the training and validation dataset (not during test).
- spoiler: The human extracted spoiler for the clickbait post from the linked web page. This field is only available in the training and validation dataset (not during test).
- spoilerPositions: The position of the human extracted spoiler for the clickbait post from the linked web page. This field is only available in the training and validation dataset (not during test).
- tags: The spoiler type (might be "phrase", "passage", or "multi") that is to be classified in task 1 (spoiler type classification). For task 1, this field is only available in the training and validation dataset (not during test). For task 2, this field is always available and can be used.

3 Model

As mentioned in the introduction, I have used a 2 stage approach for 2 subtasks i.e. spoiler type classification then spoiler generation (Fig 2). Here's the details about models for individual subtasks:

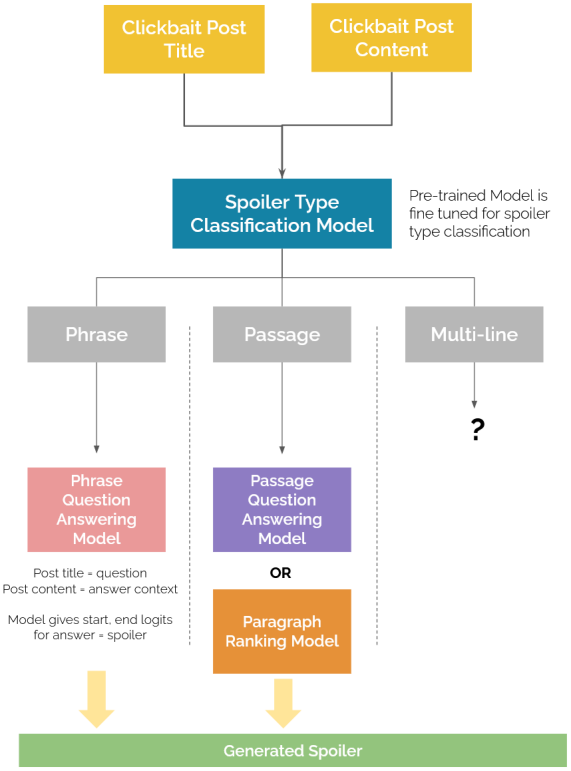

Figure 2: Model stages for spoiler type classification followed by spoiler generation.

3.1 Subtask 1: Spoiler Type Classification

As seen in Fig 2, for this task, input is clickbait post title and linked document which are fed into a sequence classifier model to classifiy into one of the three spoiler types: phrase, passage, multi-line spoiler. As a baseline I considered the bert-base-uncased model from hugging face and fine-tuned the classification head with 3 classes defined for each spoiler type. Both post title and content are tokenized together with token type defined for different parts, but it was truncated to max length of 512 given the limitation of bert model. Furthermore, I also appended post heading inside document link with content to be a part of post content. Also I am only truncating the second longer part i.e. post content.

Primary part in pre-processing of input text consists of 2 sentences as input to tokenizer i.e post title and post content. I tried other models as well like Roberta and Deberta for which I tried the same pre-processing steps. As seen in the Fig 3, BERT or any other text classification model is fine-tuned with classification head attached to CLS token to predict 3 probabilities for each spoiler type.

For task 1 model accuracy is the evaluation metric for comparing different models.

3.2 Subtask 2: Spoiler Generation

Based on spoiler type, I used different Question Answering models trained on specific tag i.e. phrase or passage to generate spoilers. Since majority QA models on hugging face are trained on SQUAD before, I preprocessed the dataset to in the squad format before fine tuning the models. Going into details of QA model, the post title is considered as question and post content as answer context from which spoiler is selected by predicting the most probable start and end indices.

In QA model preprocessing, after converting into SQUAD format, I took reference of Hugging face QA tutorial [Hugging Face(2022)] where they split the training examples into multiple features

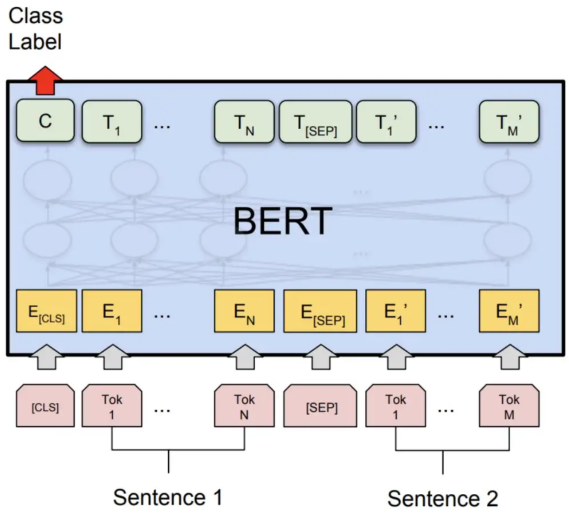

Figure 3: Fine tuning BERT for text classification using sentence 1 = post title, sentence 2 = post linked document with content.

because of longer content length instead of truncating it. In this manner we can create more data points with same post title and different part of post content. As seen in fig 4, the QA model predicts the probability of a token to be start and end index, so as output we get start and end logit probabilities for each token in answer content. Therefore we need a validation post processing as well to get 1 single best answer out of list of start, end logits. In order to find the best answer/spoiler start end pair, I find all possible combination of highest start/end logit indices, and if they are legit combination (i.e. start¡end index) then I store the score along with the best generated spoiler from post content. For both training and validation processing, I modified code from QA Hugging face tutorial [Hugging Face(2022)]. I also saw beam-search based approach for inference stage but instead relied on basic approach due to compute resource limitation.

Now for both phrase and passage training examples I trained separate models but the underlying QA model approach remained the same as mentioned above.

For task 2, I had decided to use BLEU-4 score, exact match and F-1 score as evaluation

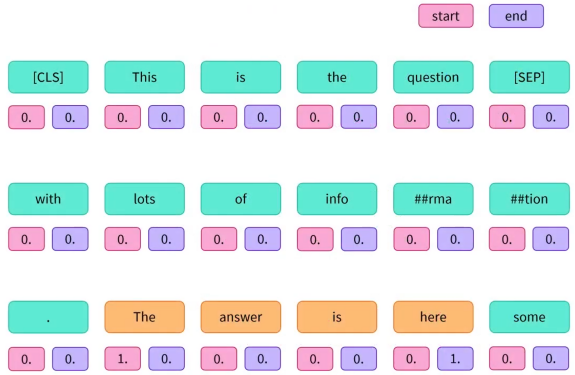

Figure 4: QA Model outputs probabilities for each token being start and end index for the potential answer/spoiler.

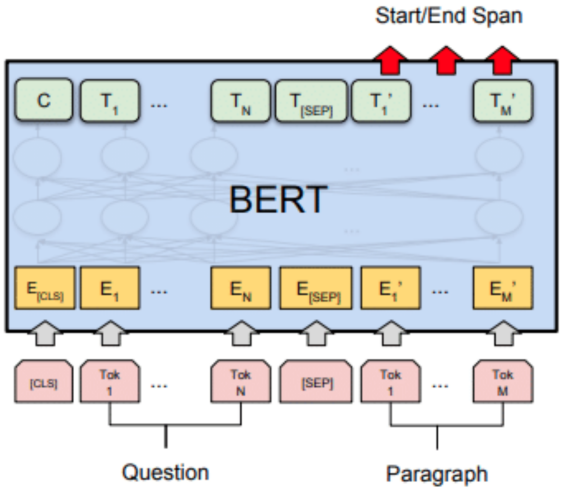

Figure 5: QA Model fine-tuning method

metric for comparing different models which is also the one used in the original paper by the authors of the competition.

Additional Passage/Sentence Ranking Model: For passage type spoiler it's difficult for QA models to generate spoilers because passage spoilers might be long and QA models might not be accurate. Therefore, I tried an approach derived from the original paper [Hagen and Fröbe(2022)] to use query as post title and compare each paragraph with query to rank them according to what's the most relevant paragraph to post title. In this manner we can just use a trained model and use it in zero-shot manner without any fine tuning.

For this approach I compared all paragraphs per post with clickbait post title to rank them according to their relevance to post. Then I selected top 5 paragraphs and split into sentences to compare each sentence with post title. In this manner I can select the sentence which is most relevant to the clickbait post title. Finally I evaluate the BLEU-4, exact match and F1-score. This approach yielded low evaluation metric results but can be expanded to larger spoiler types.

4 Results

For Subtask 1, I tried BERT, RoBerta, DeBERTa models from Hugging faces with batch=8 and trained for 10 epochs to get the best accuracy for spoiler type classification. As seen in Table 1. it can be seen that BERT performs the worst with 66.8% accuracy which is close to the results drawn in original paper [Hagen and Fröbe(2022)]. Also it can be seen that the roberta and deberta [deb()] models perform better roberta achieving the highest accuracy of 70%.

For Subtask 2, I tried different QA models, with different hyperparameteres for phrase and passage spoiler type generation like minilm [min()], roberta [rob()], and bert-base-uncased. As seen in Table 2. for phrase generation roberta model is performing the best with 36.1 BLEU-4 score. Similarly as seen in Table 3, for passage generation as well roberta outperforms other models by a margin with 37.1 BLEU-4 score. In both spoiler type, bert-base-uncased is the worst model achieving mere 17.8 and 29.1 BLEU for phrase and passage generation respectively.

Additional Passage ranking model results: For passage spoiler type I also tried two passage ranking models: microsoft's cross-encoder/ms-marco-TinyBERT-L-2 [msm()] and cross-encoder/stsb-TinyBERT-L-4 [sts()] which predict a score for each passage with respect to query i.e. clickbait post title. In this manner I get top 5 paragraphs per clickbait post then I rank all sentences from 5 paragraphs per post to select the best most relevant one to clickbait post using same

ranking model. For more details please refer to code. As seen in Table 3, Passage ranking model don't perform well and have the worst BLEU score of 13.4 at best which is lower than QA models (even worst is bert with 29.1 BLEU). But it is worth noting that these ranking models are used in a zero-shot manner without any finetuning. Therefore, I believe there is a chance that after more tweaks and fine tuning, it can be used for passage spoiler generation.

Model	Validation Accuracy
bert-base-uncased	66.8
microsoft/deberta-base-mnli	71.5
roberta-base	75

Table 1: Accuracy score for different models on SubTask 1.

Model	BLEU-4	F-1	Exact match
minilm-uncased-squad2	36.3	68.4	56.7
roberta-base-squad2	36.1	70.1	59.0
bert-base-uncased	17.8	56.2	45.9

Table 2: Evaluation comparison between best QA models for phrase spoiler generation.

Model	BLEU-4	F-1	Exact match
Question Answering Models			
minilm-uncased-squad2	31.3	41.7	14.3
roberta-base-squad2	37.1	45.8	18.6
bert-base-uncased	29.1	36.4	12.4
Passage Ranking Models			
ms-marco-TinyBERT	13.4	20.6	4.6
stsb-TinyBERT-L-4	12.1	19.9	3.7

Table 3: Evaluation comparison between best QA and Passage Ranking models for passage spoiler generation.

For Subask 2, I tried different types of QA models for both phrase and passage spoiler types. As seen in table 2, it can be observed that for phrase spoiler generation, the minilm-uncased-squad2 trained on SQUAD 2.0 dataset performs the best after fine tuning reaching an BLEU score of 36.3, exact match of 56.7 and 68.4 F-1 score.

5 Conclusion

In this project, I proposed a 2 stage approach for clickbait spoiler generation where I explored different Text classification, Question Answering and Ranking models which can be used to solve two subtasks. Based on the evaluation metric as BLEU score I was able to evaluate how QA models performed well in spoiler generation especially for phrase and passage-type spoilers. Based on my comparison between different models, it can be seen minilm and Roberta perform the best as compared to the standard vanilla bert model for phrase generation as well as passage generation. Also I explored passage ranking based approach for passage generation but it has major disadvantage over QA models with lowest BLEU score but it works in a zero shot manner. Therefore I believe as a future work, that approach can be explored more.

Also, one of the limitations of my work was how the QA model-based approach can't be used for Multi-line which requires a method that can incorporate spoiler generation as non-consecutive blocks of text from answer context. Hence, I believe in future I will try to think of way to tackle this type of spoiler generation.

References

[deb()] deberta-base-mnli. https://huggingface.co/microsoft/deberta-base-mnli. Accessed: 2022-12-05.

[min()] minilm-uncased-squad2. https://huggingface.co/deepset/minilm-uncased-squad2. Accessed: 2022-12-05.

[msm()] ms-marco-tinybert-l-2. https://huggingface.co/cross-encoder/ms-marco-TinyBERT-L-2. Accessed: 2022-12-05.

[rob()] roberta-base-squad2. https://huggingface.co/deepset/roberta-base-squad2. Accessed: 2022-12-05.

[sts()] stsb-tinybert-l-4. https://huggingface.co/cross-encoder/stsb-TinyBERT-L-4. Accessed: 2022-12-05.

[Fröbe(2022)] Maik Fröbe. 2022. Clickbait spoiling via question answering and

passage retrieval. https://github.com/webis-de/acl22-clickbait-spoiling.

[Hagen and Fröbe(2022)] Matthias Hagen and Fröbe. 2022. Clickbait spoiling via question answering and passage retrieval.

[Hagen et al.(2022)Hagen, Fröbe, Jurk, and Potthast] Matthias Hagen, Maik Fröbe, Artur Jurk, and Martin Potthast. 2022. Webis clickbait spoiling corpus 2022.

[Hugging Face(2022)] Hugging Face. 2022. Question answering hugging face tutorial. https://huggingface.co/course/chapter7/7?fw=pt#question-answering, Last accessed on 2022-12-04.

[Maik Frobe(2022)] Maik Frobe. 2022. Clickbait challenge at semeval 2023 - clickbait spoiling. https://pan.webis.de/semeval23/pan23-web/clickbait-challenge.html, Last accessed on 2022-12-04.