Billy-Batson at SemEval-2023 Task 5: An Information Condensation based System for Clickbait Spoiling

Anubhav Sharma*, Sagar Joshi*, Tushar Abhishek, Radhika Mamidi, Vasudeva Varma
IIIT Hyderabad, India
{anubhav.sharma, sagar.joshi, tushar.abhishek}@research.iiit.ac.in
{radhika.mamidi, vv}@iiit.ac.in

Abstract

The Clickbait Challenge targets spoiling the clickbaits using short pieces of information known as spoilers to satisfy the curiosity induced by a clickbait post. The large context of the article associated with the clickbait and differences in the spoiler forms, make the task challenging. Hence, to tackle the large context, we propose an Information Condensation-based approach, which prunes down the unnecessary context. Given an article, our filtering module optimised with a contrastive learning objective first selects the parapraphs that are the most relevant to the corresponding clickbait. The resulting condensed article is then fed to the two downstream tasks of spoiler type classification and spoiler generation. We demonstrate and analyze the gains from this approach on both the tasks. Overall, we win the task of spoiler type classification and achieve competitive results on spoiler generation.

1 Introduction

Clickbait is an umbrella word for various strategies used to capture attention and stimulate people's interest (Mormol, 2019). Instead of offering comprehensive summaries, clickbait postings link to web pages and sell their content by raising curiosity, making them a potential tool for spreading adversities like fake news (Chen et al., 2015). The Clickbait Spoiling shared task, which is based on the English Language, attempts to categorize and generate short paragraphs that fulfill the interest sparked by a clickbait (Fröbe et al., 2023a) through a spoiler. The task is divided into two subtasks - **(i)** Spoiler Type Classification, in which we need to "identify the type" of spoiler that would be needed for the corresponding clickbait and the article, and **(ii)** Spoiler Generation, which seeks to "generate the spoiler" for a given clickbait. The second task also uses the spoiler type information along with the post and article for generating the spoiler, considering the large difference in the nature of spoilers per type. Although we look at both the tasks individually in this work, the overarching goal lies in having a pipeline that will first predict the type of spoiler before proceeding for spoiler generation. The task deals with three spoiler types: phrase, passage and multipart, the differences in which will be discussed in Section 3. Figure 1 illustrates the two tasks where the clickbait and the article warrants a spoiler of the type "passage". The dataset for both the tasks is a collection of manually spoiled clickbaits extracted from various social media platforms like Facebook, Reddit, and Twitter (Hagen et al., 2022). Each data point consists of clickbait, multiple paragraphs in the article (which hold the content regarding the clickbait), and spoiler information.

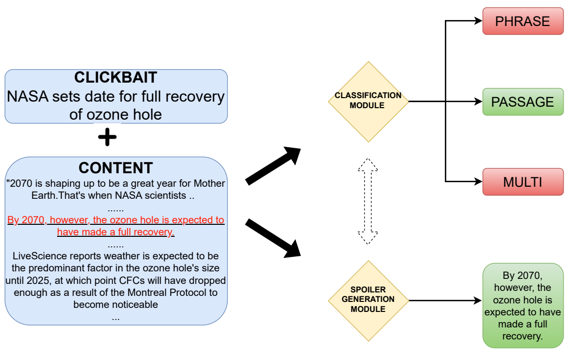
CLICKBAIT
NASA sets date for full recovery of ozone hole
+
CONTENT
"2070 is shaping up to be a great year for Mother Earth.That's when NASA scientists ..
......
By 2070, however, the ozone hole is expected to have made a full recovery.
......
LiveScience reports weather is expected to be the predominant factor in the ozone hole's size until 2025, at which point CFCs will have dropped enough as a result of the Montreal Protocol to become noticeable
...
CLASSIFICATION MODULE
PHRASE
PASSAGE
MULTI
SPOILER GENERATION MODULE
By 2070, however, the ozone hole is expected to have made a full recovery.

Figure 1: In the specified example, the text in red is the corresponding spoiler and is of the category "passage". The connection in the dotted arrow symbolizes dependency between the two tasks as a motivation for a unified pipeline for spoiling a clickbait.

In our work, we propose and study an **I**nformation **C**ondensation-based (IC) framework for the problem. By information condensation, we refer to distilling out unnecessary paragraphs from the article to supply precise information for downstream tasks. The motivation for this approach stems from the large size of the article containing a very diluted nature of the context available

*Both the authors have equal contribution to this work.

in contrast to the very specific information in the spoiler we're looking at. To tackle this, we implement various precursor techniques before the final downstream tasks. We try paragraph-wise classification with and without pretraining on datasets like Answer Sentence Natural Questions (ASNQ) for the task of Answer Sentence Selection (Garg et al., 2019), and implement the **contrastive learning** module in order to extract paragraphs from the article that are more likely to satisfy the clickbait curiosity and analyze their performance to achieve the Information Condensation needed. We probed over two different data feeding strategies to our models - **SIMPL** and **CONCT** (more on it in Section 4.2.2), and analyzed their performances.

For the downstream tasks, we model using the multiclass classification paradigm for the first task and extractive question answering paradigm for the second task, where we train individual models for each spoiler type (Fröbe et al., 2023a). For modeling these approaches, we heavily rely on pretrained transformer architectures like RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2020).

The major findings and insights we present from our experimentations are summarized below:

1. The task of Paragraph-wise classification to find out if a given paragraph contains a spoiler, is challenging: a binary classification setup is not adequate for the task.
2. Considering that we need the intermediate task for reducing the information for downstream tasks while maintaining a high probability of spoiler being present in the input, a high recall system is required from the intermediate stage.
3. A contrastive learning-based solution works the best to provide the high recall needed with good precision in selected passages. We measure the same using ranking metrics.
4. Out of the two data feeding techniques, SIMPL outperforms CONCT because of better information exchange between the two inputs.
5. While filtering strategy works well for the spoiler type classification task, the disturbances to textual coherence caused and the possible loss of spoiler content adversely affect models already trained on dealing with coherent textual contexts for the spoiler generation task.

For the first subtask, our submission was ranked highest with the best performing model having a 1.71% improvement over the task baselines. For the second subtask, we achieve a BLEU score of 40.14 for phrase spoiler extraction and 36.90 on passage spoiler extraction. Our best performing approach was evaluated on test set using the TIRA platform (Fröbe et al., 2023b). Our code is open sourced for replication and further analysis[1].

2 Related Work

The task of spoiler detection and generation was introduced in Fröbe et al. (2023a). Research studies related to clickbait have been categorised into clickbait detection and clickbait generation. The domain of automated clickbait detection, which was started by Rubin et al. (2015), and further independently explored by Potthast et al. (2016) and Chakraborty et al. (2016) has been heavily dominated by the transformer models. Some other progress in terms of clickbait generation was done by Xu et al. (2019). The task of spoiling a clickbait can be analogous to a question answering paradigm (Rajpurkar et al., 2016), where given a clickbait as a question, the answer would be a spoiler.

3 Data Description

For majority of our experiments, we have used the dual set up of the original Webis Clickbait Spoiling (WCS) Corpus 2022 (Hagen et al., 2022) and the Paragraph-wise (Pw) Dataset which we derive from the WCS Corpus. The WCS Corpus has 14 fields giving information about the clickbait, article title, article content, spoiler category type and spoiler positions. The three types of spoilers vary on the basis of their length and structure: (1) phrase is a short span, spanning an average of 2.8 words, (2) passage is a longer span, with an average of 24.1 words and (3) multipart is a bullet-list style structure consisting of individual phrase and passage-type spoilers.

Dataset Name	Train	Dev
WCS Corpus	3200	800
Pw Dataset	48,626	12,410

Table 1: Statistics of the given WCS Corpus for the main task and the derived Pw Dataset.

The training and validation split stats for the WCS-Corpus and the Pw Dataset can be found in Table 1. More details on the creation of the derived

[1]https://github.com/anubhav-sharma13/ic-clickbait-spoiling

Pw Dataset along with the other data sources used can be found in Appendix A.

4 Methodology

In this section, we discuss our proposed approach consisting of paragraph-wise filtering for information condensation followed by finetuning on downstream tasks. Given a clickbaity post c and an article a consisting of the page title p_0 followed by n paragraphs $p_1, p_2, ..., p_n$, we need to (1) identify the spoiler type t needed to spoil the clickbait, and (2) generate a spoiler s, using the additional information of spoiler type t.

4.1 Oracle Experiments

To demonstrate the advantage in passing focused information to the model and to establish an upper limit of performance in this regard, we conduct oracle experiments for both the downstream tasks. For these experiments, we assume the existence of an oracle that selects only those paragraphs p_i from a which contain s. The extremely shortened article is then fed to a classification model C_{oracle} for the first task and to the extractive question-answering models $G_{oracle-phrase}$ and $G_{oracle-passage}$ for the phrase and passage spoiler extraction in the second task.

Encouraging results from the oracle experiments motivated our approach for optimizing on the intermediate task of paragraph filtering as explained in Section 4.2 to reach closer to the limits established by the oracle experiments.

4.2 Paragraph-wise Filtering

The goal of paragraph-wise filtering stage is to achieve the information condensation from the content of the article. This is not only necessary considering the large textual length of a as compared to the input limits of pretrained transformer models, but also because of the advantage gained in feeding more focused information to the model (which our oracle experiments validate). Here, we need to identify the top k paragraphs from a, making use of c to guide the selection. This creates the condensed article, a_c. The value of k needs to be chosen to maximize the recall, which indicates that the trained model for filtering should be able to generally rank the paragraphs containing the spoiler at very high ranks.

4.2.1 Classification Paradigm

Vanilla classification. Our first attempt on paragraph-wise filtering was modeled as a binary classification task. Here, we need to finetune a classification model $PARA_{CLF}(\theta_{CLF})$ to classify if a passage p_i contains an overlap with the spoiler s for the given clickbait c. This was an imbalanced classification problem as discussed in Section 3. We experimented with Cross Entropy, Weighted Cross Entropy and Focal Loss (Lin et al., 2017) as the objectives for directly finetuning models on the task. Once the model is trained, we infer the paragraph score $sc_i = P(p_i \cap s \neq \phi \mid c, p_i; \theta_{CLF})$ which is used in the filtering stage as discussed in Section 4.2.3.

ASNQ Pretraining. Considering the difficulty of the task, we also tried out prior conditioning of the model on the task of Answer Sentence Selection (AS2) (Yang et al., 2015). The task of AS2 aims to identify the sentences which contain an answer to the given question from a pool of candidates, and is very similar to our setting for paragraph filtering. The Answer Sentence Natural Questions (ASNQ) (Garg et al., 2019) dataset was introduced as an effective dataset for transferring a general pretrained model on the AS2 task. Our paragraph classification model $PARA_{CLF}$ was adapted on the Pw Dataset following the transfer step on ASNQ (using the Focal loss objective), following Garg et al. (2019).

4.2.2 Contrastive Learning Paradigm

Modeling. As opposed to classification, here, we adopt a ranking-based paradigm to optimize the model to rank the positive paragraphs higher among all the paragraphs in the article. For training, we implement a pairwise contrastive approach to train a model $PARA_{CONT}(\theta_{CONT})$. Given two clickbait-paragraph pairs, out of which one pair has a positive paragraph (c, p_{pos}) and the other has a negative paragraph (c, p_{neg}), the model is trained to give a higher score sc_{pos} to the positive pair than to the negative pair sc_{neg} by at least a threshold value $threshold_{CONT}$. Thus, the objective can be given as $l_{CONT} = max(0, sc_{neg} - sc_{pos} + threshold_{CONT})$.

At inference time, we pass a single paragraph p_i to the model in one pass to infer its corresponding score sc_i for filtering as elaborated in Section 4.2.3.

Data Feeding Techniques. In designing the model architecture for contrastive learning, we ex-

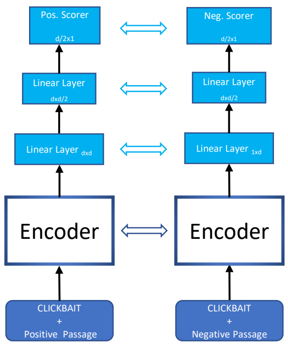

Figure 2: Model architecture for contrastive learning from clickbait-paragraph text sequence.

perimented with two settings for feeding the inputs to the model: (1) concatenating the embeddings for the clickbait and paragraph separately by Siamese modeling, which we refer to as **CONCT**, and (2) passing the concatenated clickbait-paragraph text sequence as a unified input to the model, which we refer to as **SIMPL**. The SIMPL setting performed turned out to be the better architecture as we show and discuss in Section 6.1. We have illustrated the SIMPL in Figure 2.

4.2.3 Inferencing for Paragraph Filtering

The classification model $PARA_{CLF}$ or the contrastive model $PARA_{CONT}$ is trained to give a paragraph-level score sc_i to each paragraph p_i of the given article a. We use this score to first rank the paragraphs in a followed by the selection of the top k paragraphs for an empirically suitable value of k. The value of k is fixed so as to maximize the recall of the spoiler-containing paragraphs while achieving an effective condensation of the article. The selected paragraphs are then sequentially sorted to form the condensed article a_c which can be passed to the downstream tasks.

4.3 Spoiler Type Classification

This is one of the two downstream tasks that are to be solved as a part of the complete problem. Here, we model a classifier model C to classify which spoiler type does a given pair of clickbait c and article a warrant. We tried out the SIMPL and CONCT strategies explained in Section 4.2.2 for passing the inputs a and c to the model. The CONCT strategy was tried because of the unfiltered a being particularly long, thus passing it separately would enable the model to capture more content in its input window. If paragraph-wise filtering is used, the article is passed in its condensed form a_c to the model, with which we use the SIMPL strategy. The classifier C was optimized on the cross entropy objective in a typical 3-way classification problem.

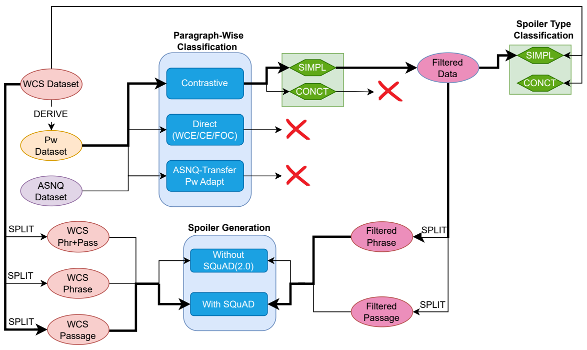

Figure 3: Experiment flow for our entire system. The red cross represents the termination of an approach due to inferior performance compared to the best-performing one, whose flow is marked with bold arrows.

4.4 Spoiler Generation

In the second downstream task, we need to extract the spoiler of the given type t for an input (c, a) pair. Considering the nature of the phrase and passage spoiler generation tasks as being the extraction of single span from the article, we model the tasks in the extractive question-answering (QA) paradigm. Here, c represents the question and the article a represents the context from which we need to extract the answer s of the given t. We initially experimented with the extraction of phrase and passage spoilers using a unified QA model while not making use of the spoiler type information at the input, modeling a single extractive QA model $G_{phrase-passage}$. Considering the differences in the nature of the spoilers per category, we used the spoiler type information t to model a separate QA model per category, G_{phrase} and $G_{passage}$. The models used were also pretrained on the SQuAD 2.0 dataset (Rajpurkar et al., 2018) to benefit from task-specific alignment before finetuning on the small spoiler type-specific subsets of the WCS Corpus.

5 Experimental Setup

5.1 Training

We explain the settings adopted for the experiments based on classification, contrastive learning and extractive QA-based experiments. The flow of experiments has been visualised in Figure 3.

Classification-based. For the experiments specific to paragraph-wise filtering, we used the Pw Dataset, while we use the WCS Corpus (with and without filtering) for the spoiler type classification experiments. The ASNQ dataset was used for the initial transfer step following the classification on Pw Dataset (Pw adapt step) (as described in Section 3).

For modeling, we used transformer-based models like RoBERTa-*base* (rb-b), RoBERTa-*large* (rb-l), and DeBERTa-*base* (db-b) as the encoders. Besides the Cross Entropy (CE) objective which we commonly applied for all our experiments, we also tried out with Weighted Cross Entropy (WCE) and Focal Loss (FOC) objectives on the Pw classification task owing (Vanilla) to its imbalance in classes. We discuss the architectural specifics of the model in Appendix B.1. The learning rate is selected from {1e-5, 2e-5 and 1e-4} in combination with the number of epochs count chosen from a set of {10, 15 and 25}. The model parameters are optimized using the AdamW optimizer with default parameters. The learning rate is increased till 10% of the total training steps, followed by linear decay.

Contrastive learning-based. In our pairwise contrastive learning setup, at each training step, we randomly sample a p_{neg} from all the negative paragraphs from the corresponding article for each given positive pair (c, p_{pos}). We used 0.5 as the threshold $threshold_{CONT}$ in the margin ranking ranking objective l_{CONT}. Architectural specifics for the models are discussed in Appendix B.2.

Extractive QA-based. We employ the extractive QA paradigm for phrase and passage spoiler extraction in which we apply a linear layer on the encoder output embeddings for classifying the spoiler start and end tokens. We use DeBERTa-*base* (db-b) and RoBERTa-*large* (rb-l) as the encoder models. We also use a checkpoint of the RoBERTa-large (rb-l) finetuned on the SQuAD 2.0 dataset (rb-l-squad) to benefit from task-specific learning. As the context, we experiment with using the unfiltered a from the WCS Corpus as well as the condensed article a_c after paragraph filtering. All the models were trained for 5 epochs, with the rest of the hyperparameters being the same as used for the classification-based experiments. Appendix C provides the specifics on spoiler position mapping for extractive QA modeling.

Filtering. The best performing RoBERTa-*large* model trained with contrastive learning objective was used for condensing the article. The value of k was chosen to be 5 to maximize the recall of positive paragraphs while optimally condensing the article. Section 7.2 provides more insights on the same.

5.2 Evaluation

Classification. The tasks modeled as classification were evaluated using the usual metrics of Accuracy, Precision, Recall, F1 and Matthews Correlation Coefficient (MCC). Precision, Recall and F1 for multiclass classification were calculated using the macro strategy. It should be noted that macro-Recall is the same as Balanced Accuracy (the primary metric for evaluating the task of spoiler type classification), and we thus refer to the same as Balanced Accuracy in the case of multiclass classification experiments.

Ranking. We used ranking-based metrics of Mean Reciprocal Rank (MRR) and Mean Rank to evaluate the trained models for paragraph-based filtering, along with the classification-based metrics. Ranks given to all the positive paragraphs in an article were considered in the computation of these metrics.

Generation. Generation-based metrics of BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005) and BERTScore (Zhang et al., 2019) were used along with the Exact Match metric to get an idea of the direct matches between the extracted spoilers versus the expected ones. Following the organizers, BLEU was calculated as an average of the Sentence-BLEU scores [2].

6 Results

We discuss the quantitative results on all our primary experiments in this section.

6.1 Paragraph-wise Filtering

The models trained on classification and contrastive learning-based metrics are evaluated using both - classification and ranking-based metrics to establish a common ground for best model selection. To evaluate the models using the ranking metrics, we use the paragraph score sc_i as discussed in Section 4.2.3. To evaluate the contrastive model using classification-based metrics, we use a threshold of 0.5 to classify the prediction of the model as a positive or a negative paragraph. Table 2 shows the

[2]https://www.nltk.org/_modules/nltk/translate/bleu_score.html

Experiment	Model	F1	Recall	Precision	MCC	Accuracy	MRR	Mean Rank
Vanilla-CE	rb-b	71.46	71.58	71.35	42.93	90.30	0.6442	3.46
	db-b	71.63	69.16	75.20	43.95	91.54	0.6438	3.40
Vanilla-WCE	rb-b	71.72	69.62	74.56	43.91	91.38	0.6224	3.88
	db-b	73.52	71.37	**76.37**	47.48	**91.88**	0.6325	3.91
Vanilla-FOC	rb-b	72.35	70.41	74.89	45.08	91.48	0.6412	3.70
	db-b	72.68	70.83	75.05	45.68	91.53	0.6305	3.96
ASNQ-Transfer→Pw-Adapt	rb-b	71.66	73.18	70.40	43.50	89.76	0.6446	**3.36**
	db-b	**74.07**	**72.91**	75.41	**48.26**	91.66	**0.6596**	3.61
Contrastive-CONCT	rb-b	57.54	68.75	57.85	24.26	74.19	0.5120	4.67
	db-b	62.06	66.60	60.32	26.17	83.09	0.4966	5.06
Contrastive-SIMPL	rb-b	62.94	69.50	60.98	29.26	82.30	0.5816	3.98
	db-b	64.45	64.63	64.27	28.90	87.82	0.5874	3.99
	rb-l	65.11	69.56	62.97	31.86	85.09	0.6271	3.55

Table 2: Results for the experiments on paragraph-wise filtering on the dev set of the Pw dataset. F1, Recall and Precision are computed in macro forms.

results of the experiments under paragraph-wise filtering.

As can be seen, rbl model under Contrastive-SIMPL was the best performing model on most classification metrics while decisively leading the chart on the ranking metrics. A detailed analysis and interpretation of the results can be found in Appendix D. This model was used to prepare the Filtered data containing condensed articles for downstream tasks.

6.2 Spoiler Type Classification

Table 3 shows the results of our experiments on the spoiler type classification task. On the WCS Corpus, we include experiments using the SIMPL (WCS-SIMPL) and CONCT (WCS-CONCT) strategies. On the Filtered data, we show results on the better performing SIMPL strategy (Filtered-SIMPL) for $k = 5$. Results for the Oracle-based experiments are added on top for establishing upper limits on the Filtering strategy. For the experiments on full data, the SIMPL strategy outperformed the CONCT strategy for data feeding for reasons similar to those attributed in Section 6.1. Using the Filtered data with the SIMPL strategy improved the results, thus showing the benefit of information condensation for the task. Among each set of experiments, rbl outperformed its counterparts on most metrics.

The best performing rbl model under Filtered-SIMPL was evaluated on the test set, which gave a Balanced Accuracy of 74.14%, being the winning entry (rank 1) on the task.

6.3 Spoiler Generation

We show the results for phrase and passage spoiler extraction tasks in Table 4. Like for spoiler type classification, we include results on the Oracle-based experiments followed by experiments on the WCS Corpus and the Filtered data. As a common observation throughout the experiments, task-specific training helps - the rb-l-squad model already finetuned on the SQuAD 2.0 dataset easily outperforms its counterpart rb-l which is directly finetuned on the task. Among directly finetuned models, the rb-l model in turn outperforms db-b in virtue of larger number of pretrained parameters for modeling. On the WCS Corpus, we show the results on a unified QA model (WCS-phrase+passage) for phrase and passage spoiler extraction. The results are easily outperformed by QA models trained individually on phrase (WCS-phrase) and passage (WCS-passage) spoiler extraction objectives showing the need for spoiler-specific modeling. Using the Filtered data generally leads to an improvement in performance as compared to the full WCS Corpus for db-b and rb-l that are directly finetuned on the task. However, for rb-l-squad, we obtain only a minor improvement on phrase spoilers and a decrease in performance on passage spoilers. We analyze this behavior of rb-l-squad in a greater depth in Section 7.3.

7 Analysis

7.1 Differences based on Spoiler Type

Figure 4 displays the performance of the best performing (Filtered-SIMPL: rb-l) model for spoiler type classification for individual spoiler types. As can be seen from the confusion matrix, phrase and passage spoiler types are most easily confused with each other. In comparison, multipart is the least confused with the other types. The behaviour of multipart being predicted more in place of the ac-

Experiment	Model	Balanced Accuracy	F1	Precision	MCC	Accuracy
Oracle-SIMPL	*db-b*	*78.20*	*78.58*	*79.00*	*65.02*	*78.00*
	rb-b	*73.36*	*74.92*	*77.34*	*59.01*	*74.38*
	rb-l	*80.87*	*80.71*	*80.73*	*67.87*	*79.63*
WCS-CONCT	db-b	68.92	69.76	71.31	53.67	70.87
	rb-b	66.09	66.98	71.93	49.54	68.37
	rb-l	67.69	69.55	74.05	53.11	70.63
WCS-SIMPL	db-b	72.32	73.33	74.71	58.21	73.88
	rb-b	67.34	67.39	68.99	50.25	68.25
	rb-l	73.38	73.40	73.47	58.01	73.50
Filtered-SIMPL	db-b	74.14	**73.92**	75.91	**60.62**	74.25
	rb-l	**75.11**	73.78	**76.77**	60.52	**74.88**

Table 3: Results on spoiler type classification task for dev set. Balanced Accuracy is the same as macro-Recall.

Experiment	Model	Phrase Spoilers, *n=335*				Passage Spoilers, *n=322*			
		BLEU-4	METEOR	BERTScore	Exact Match	BLEU-4	METEOR	BERTScore	Exact Match
Oracle-phrase / Oracle-passage	*db-b*	*41.68*	*60.14*	*95.74*	*59.40*	*57.98*	*73.67*	*94.54*	*20.81*
	rb-l	*44.42*	*63.61*	*96.41*	*64.18*	*52.89*	*68.30*	*93.97*	*15.53*
	rb-l-squad	*48.41*	*67.90*	*97.20*	*69.55*	*59.87*	*73.91*	*95.15*	*18.94*
WCS-phrase+passage	db-b	31.94	49.24	93.47	45.37	22.66	32.41	88.72	6.83
	rb-l	33.24	50.98	94.08	49.85	24.40	34.23	89.17	8.70
	rb-l-squad	35.59	54.66	94.51	53.43	31.66	42.26	90.56	11.80
WCS-phrase / WCS-passage	db-b	32.53	50.74	93.52	47.15	25.32	36.61	89.08	7.14
	rb-l	34.93	51.66	94.55	50.15	27.62	39.33	89.58	8.38
	rb-l-squad	39.74	57.20	95.54	**59.10**	**36.90**	**49.30**	**91.37**	**13.66**
Filtered-phrase / Filtered-passage	db-b	32.89	48.52	93.81	47.76	27.28	38.07	89.39	7.45
	rb-l	36.18	52.27	94.55	52.54	28.31	41.84	89.80	6.83
	rb-l-squad	**40.14**	**58.76**	**95.84**	57.91	35.02	48.53	90.97	10.25

Table 4: Results for phrase and passage spoiler generation (extraction) on the dev set.

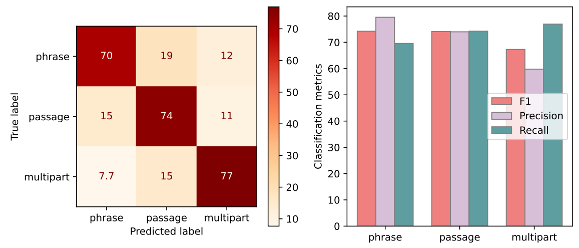

Figure 4: Performance analysis of the best model on spoiler type classification with confusion matrix (left) and bar chart for type-wise classification metrics (right).

tual class can also be reflected from the bar plot which shows the lowest precision and highest recall for the type. The highest type-wise F1 is obtained on the passage type.

7.2 Spoiler Loss Due to Chosen k

	phrase	passage	multipart
Contrastive-SIMPL: rb-l	2.78	3.80	5.36
Contrastive-SIMPL: db-b	2.97	4.29	6.24

Table 5: Type-wise differences in Mean Rank for the two best-performing models on the Pw task.

In Table 5, we compare the Mean Rank per spoiler type for the top two models for paragraph-wise filtering. As can be seen, paragraphs containing phrase spoilers are the easiest to identify and rank higher, followed by passage and multipart. In condensing an article, we select a value of k to select from the top ranked paragraphs. We analyze the loss of spoilers resulting from different values of k in Figure 5. It can be seen that articles with multipart spoilers are the most difficult candidates for achieving condensation, having highest losses throughout. Our selection of $k = 5$ stems from the observation of the decrease in spoiler loss being reduced beyond this point for all the types, thus being a suitable value for achieving high recall with an effective condensation of the article.

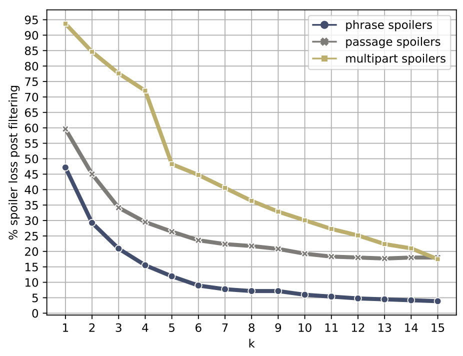

Figure 5: Plot of percentage loss of spoiler containing paragraphs in an article for different values of k.

7.3 Performance Variation with Filtering

We study the variation of phrase and passage spoiler extraction for $k = 2$ to 15 across two models: rb-l directly finetuned on the task versus rb-l-squad already finetuned on SQuAD 2.0 using Figure 6. To conduct this analysis, we trained the models individually for each value of k and spoiler type using the hyperparameter settings in Section 5.1. For phrase spoiler extraction on rb-l, we find that condensation helps: the model achieves high BLEU scores at lower value of k which is followed by a decreasing trend on the higher values. In case of passage spoiler extraction on rb-l, the condensation helped with a larger value of k (peaking on $k = 9$) due to significant loss of spoiler containing paragraphs in the initial values of k.

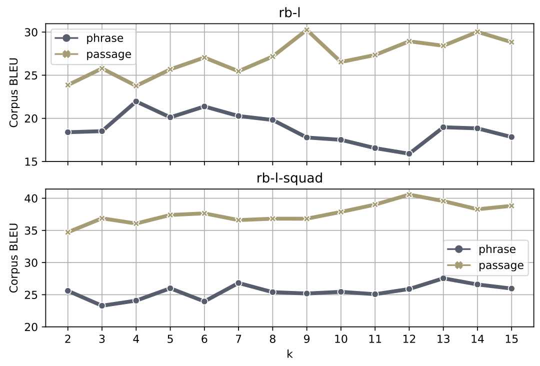

Figure 6: Variation in Corpus BLEU score of rb-l and rb-l-squad models for different values of k.

The behavior on both the spoiler types for rb-l-squad brings out differing findings. The model improves performance with increasing values of k, with peaking achieved at higher values. This can be attributed to the rb-l-squad model being already trained to handle large, coherent contexts from the SQuAD 2.0 dataset as opposed to the lack of coherence in the Filtered data which contains discretely chosen paragraphs from the article. The lack of coherence in the article, coupled with the spoiler loss due to filtering contributed to the lesser performance on lower values of k.

7.4 Qualitative Analysis

spoiler type	$O == E$	$O \in E$	$E \in O$	otherwise
phrase	57.91	9.25	10.45	22.39
passage	10.25	30.74	6.83	52.18

Table 6: $\%$ overlap statistics for phrase and passage spoiler extraction between the output (O) and expected (E) spans by the best models on each.

We show a few examples of extracted spoilers for phrase and passage spoiler extraction in Appendix E and F respectively. We also compare the $\%$ overlap between the generated and extracted spoiler spans by the best performing models on phrase and passage spoiler extraction in Table 6. In phrase extraction, we observe a good examples of perfect overlap between the output and expected spans, along with an almost equal percent of the samples where the either is a subset of the other. For the samples where there was no complete overlap as well, we found good semantic similarity between the output and the expected spoilers. Refer to Table 7 for illustrative examples in this regard.

For passage spoiler extraction, we found a substantial number of instances where the output was a subset of the expected as compared to that for phrase, indicating that the model struggles to extract more specific information needed in the spoiler span. As can be observed in Table 8, we find instances where the model offers alternative spoilers which are not lexically similar. This brings up the shortcomings in the lexical matching-based evaluation metrics like BLEU as well.

8 Conclusion

Clickbait spoiling, as opposed to clickbait detection, which frequently includes filtering out clickbait postings from users' timelines, subverts the enthusiasm aroused by clickbait by revealing the inner details in advance. Our idea of Information Condensation - providing a precise spoiler containing fragments of an article instead of the entire article, worked well on the spoiler type classification task and spoiler generation task on phrase type spoilers. We tried multiple techniques for Information Condensation, but found Contrastive learning based approach to be the most optimum and used it to filter paragraphs for both the tasks. For spoiler generation of passage type, the contrastive learning approach didn't result in an improvement due to loss of spoiler containing paragraphs and issues with textual coherence. Future work can look at the specific issue of spoiler containing paragraphs and would aim at reducing this loss (as results from Oracle experiments validates the correctness of approach). Further we can expand our experiments to generate multipart type of spoilers and analyse our findings.

9 Acknowledgments

We sincerely thank the shared task authors for putting such a challenging task. Special thanks to Maik Fröbe for always being helpful and thoughtful. We would also like to thank our institute, IIIT Hyderabad for providing us with all the necessary resources for this work.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop clickbait: Detecting and preventing clickbaits in online news media. In *2016 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)*, pages 9–16. IEEE.

Yimin Chen, Niall J Conroy, and Victoria L Rubin. 2015. Misleading online content: recognizing clickbait as" false news". In *Proceedings of the 2015 ACM on workshop on multimodal deception detection*, pages 15–19.

Maik Fröbe, Tim Gollub, Matthias Hagen, and Martin Potthast. 2023a. SemEval-2023 Task 5: Clickbait Spoiling. In *17th International Workshop on Semantic Evaluation (SemEval-2023)*.

Maik Fröbe, Matti Wiegmann, Nikolay Kolyada, Bastian Grahm, Theresa Elstner, Frank Loebe, Matthias Hagen, Benno Stein, and Martin Potthast. 2023b. Continuous Integration for Reproducible Shared Tasks with TIRA.io. In *Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023)*, Lecture Notes in Computer Science, Berlin Heidelberg New York. Springer.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2019. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *AAAI Conference on Artificial Intelligence*.

Matthias Hagen, Maik Fröbe, Artur Jurk, and Martin Potthast. 2022. Clickbait Spoiling via Question Answering and Passage Retrieval. In *60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, pages 7025–7036. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *ArXiv*, abs/2006.03654.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Paulina Mormol. 2019. 'i urge you to see this...'. clickbait as one of the dominant features of contemporary online headlines. *Social Communication*, 5(2):1–10.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. 2016. Clickbait detection. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 810–817. Springer.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Victoria L Rubin, Niall J Conroy, and Yimin Chen. 2015. Towards news verification: Deception detection methods for news discourse. In *Hawaii International Conference on System Sciences*, pages 5–8.

Peng Xu, Chien-Sheng Wu, Andrea Madotto, and Pascale Fung. 2019. Clickbait? sensational headline generation with auto-tuned reinforcement learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

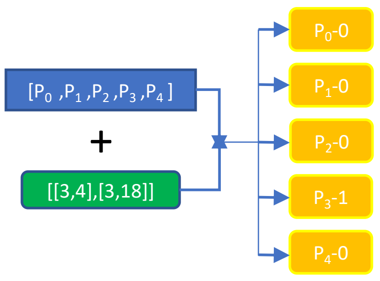

Figure 7: The blue and green box refer to a datapoint's article content and spoiler positions (marking start and end positions) in the WCS corpus. The orange boxes are datapoints in Pw Dataset, with 0 and 1 specifying the presence of a spoiler.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *ArXiv*, abs/1904.09675.

A Additional Data Curation

The Pw Dataset is a derived dataset that we curated using the spoiler positions present in the WCS Corpus as illustrated in Figure 7. Each article in WCS corpus has an average of 14.2 paragraphs. For curating a datapoint in the Pw corpus, each entry in the WCS corpus is divided on the number of paragraphs in the article. An extra binary label is added to signify whether the paragraph contains any part of the spoiler. A value of 1 indicates that the spoiler is present in that particular paragraph, and 0 suggests that it is unnecessary for that unique spoiler. The Pw Dataset has a stark class imbalance as the ratio of label 1 to label 0 is 1 to 10.

Apart from the given datasets, we use ASNQ and SQuAD 2.0 datasets as auxiliary sources for training. ASNQ dataset is used as a primary task for classification before finetuning on the Pw Dataset. We ensured that the ratio of positive is to negative sample in the ASNQ dataset is 1:10 (similar to Pw Dataset). Whereas, SQuAD 2.0 is used for a precursor training step prior to finetuning for spoiler generation.

B Architectural Specifics

B.1 Classification-based Models

For the *base* encoder models, the output embedding size (d) is 768, whereas for *large* encoder models, it is 1024. The encoder output embedding is projected through a separate linear layer ($d*d$) each for the clickbait and article embeddings in the CONCT strategy following which they are concatenated to obtain a unified representation ($2d$). The projected embeddings are then passed through two linear layers ($2d * d$, $d * d/2$) and the classification layer ($d/2 * 3$). In contrast, in the SIMPL strategy, we rather use single linear layer on the encoder output embedding for classification.

B.2 Contrastive Learning-based Models

Similar to the classification-based experiments, we compute projections ($d * d$) for clickbait and paragraph passed to the model in the CONCT strategy followed by concatenation ($2 * d$) and application of 2 linear layers ($2d * d$, $d * d/2$) for hidden representations and a final linear layer ($d/2 * 1$) for scoring. The SIMPL strategy involved a single layer ($d * d/2$) for hidden representation on the encoder output followed by the output layer ($d/2 * 1$).

C Spoiler Position Mapping for Extractive QA

The spoiler positions for each spoiler segment present in the data were given as $((st_i, st_j), (en_i, en_j))$, where st, en denote the start and end positions, and i, j denote the paragraph index and within-paragraph character offset respectively. These positions were mapped to the overall and start and end tokens indices in the tokenized article before feeding as labels to the QA model. Performing the oracle-based and filtering-based experiments involved a remapping of the paragraph indices i to suit the truncated article. An empty span was used as the training label in the small number of cases where the span was not present in the filtered data or where the span was out of bounds of the encoder model input for the complete article.

D Analysis of Results for Paragraph-wise Filtering

Among the two paradigms chosen to model the problem, the contrastive learning-based paradigm turned out to be a better strategy to achieve paragraph filtering, as can be observed from the classification and ranking metrics in Table 2. Comparing the *base* versions of the respective models, DeBERTa outperformed its RoBERTa counterpart in the classification-based models, while underperforming on the contrastive-based models. Within the classification models, there was no clear difference observed in the performance achieved by the three vanilla models (Vanilla-CE/WCE/FOC) on the classification metrics, with Vanilla-FOC turning out to be marginally better on the ranking metrics. There was a small benefit from the ASNQ transfer step as can be observed from the performance improvement in ASNQ-Transfer$\rightarrow$Pw-Adapt, showing the need for a better model adaptation with suited datasets as opposed to objective designing. Among the contrastive models, the SIMPL strategy clearly outperformed its CONCT counterpart. The reason for this can be attributed to the SIMPL strategy involving a computation of self-attention across the full clickbait-paragraph sequence over the transformer encoder pipeline leading to richer information exchange between the two pieces of input. Opposed to this in the CONCAT strategy, the two representations are computed separately by the encoder. Thus, the interaction between the pair of inputs occurs only in the embedding space, leading to a weaker modeling.

E Examples for Phrase Spoiler Extraction

We analyze the spoiler extraction performance of the best performing Filtered-phrase: rb-l-squad model in Table 7. We show four categories of outputs observed: (1) accurately matching with the expected, (2) extracting a semantically similar but lexically different alternative (3) output being a part of the expected (4) expected being a part of the output.

F Examples for Passage Spoiler Extraction

In Table 8, we analyze the performance of the best performing model WCS-passage: rb-l-squad. We show four categories of outputs extracted: (1) same or almost same matching with the expected passage span, (2) the extracted span offering an alternative to the expected by either being semantically similar or offering a different perspective, (3) the output and the expected spoiler spans differ in the amount of specificity of information in each other, and (4) containing very different (mostly incorrect) extraction than expected.

	Clickbait	Output	Expected
Output == Expected (Exact)	An unlikely company is crushing America's biggest clothing stores	Amazon	Amazon
	NJ police searching for 2 men who they say tried to abduct a 10-year-old girl when she was doing this	playing hide-and-seek	playing hide-and-seek
Output == Expected (Semantically)	Apparently All Taylor Swift and Tom Hiddleston Do Is...	beach strolling	Take Walks on Beaches
	When This Guy Bought A Used Car, He Never Thought He'd Find THIS Hidden Inside.	stacks upon stacks of bills	money
Output < Expected	This is how much coffee Americans drinks every day	2.1	2.1 coffee drinks
	The cutest little European city you've probably never heard of	Lviv	Lviv, Ukraine
Output > Expected	Blocking this color light may help you sleep better	blue wavelength light	blue
	Fashion brand lets models go un-Photoshopped and makeup-free	Rag & Bone's DIY Project	Rag & Bone

Table 7: Some examples produced by the best performing model (Filtered-phrase: rb-l-squad) for phrase spoiler type extraction.

	Clickbait	Output	Expected
Output Expected	This chain will give you free burgers for life, but on one condition	The company wants you to change your last name to "burger."	The company wants you to change your last name to "burger."
	Dog Dies One Hour After Hiking With His Owner, Veterinarian Gives Shocking Reason Why	the plant the dog was chewing on was deadly water hemlock	the plant the dog was chewing on was deadly water hemlock
Alternative spoilers	A mother hears her dead son's heart beat	she meets the man who received his heart in a transplant	Dawn Grace lost her son four years ago, and now she meets the man who received his heart in a transplant.
	Why it's definitely worth it to get your flu shot	reduced their risk of heart attack, stroke, or other cardiovascular health problems	reduced the risk of flu-related hospitalization
Difference in specificity	Coming this fall	His untitled Daily Show companion series	Daily Show companion series will air Monday-Thursday at 11:30 p.m. ET/PT, beginning this fall
	Are Shorter Work Days Better For Your Health &; Productivity?	sharply reduced absenteeism, improved productivity, enhanced creativity, and reduced turnover	An ongoing study out of Sweden seems to indicate that a shorter work day may actually result in more productivity.
Very different or incorrect generations	This is the difference between a job and a calling	work is most fulfilling when it's a calling	it's a calling because you find a deep sense of purpose and positive impact in your role
	I was really bad at sports in high school. This new study helps me understand why.	practice just doesn't matter that much	Some people are just better at sports than others

Table 8: Some examples produced by the best performing model (WCS-passage: rb-l-squad) for passage spoiler type extraction.