Billie-Newman at SemEval-2023 Task 5: Clickbait Classification and Question Answering with Pre-Trained Language Models, Named Entity Recognition and Rule-Based Approaches

Andreas Kruff and **Anh Huy Matthias Tran**
Claudiusstraße 1, 50678 Köln
andreas.kruff@smail.th-koeln.de
anh_huy_matthias.tran@smail.th-koeln.de

Abstract

In this paper, we describe the implementations of our systems for the SemEval-2023 Task 5 'Clickbait Spoiling', which involves the classification of clickbait posts in sub-task 1 and the spoiler generation and question answering of clickbait posts in sub-task 2, ultimately achieving a balanced accuracy of 0.593 and a BLEU score of 0.322 on the test datasets in sub-task 1 and sub-task 2 respectively. For this, we propose the usage of RoBERTa transformer models and modify them for each specific downstream task. In sub-task 1, we use the pre-trained RoBERTa model and use it in conjunction with NER, a spoiler-title ratio, a regex check for enumerations and lists and input reformulation. In sub-task 2, we propose the usage of the RoBERTa-SQuAD2.0 model for extractive question answering in combination with a contextual rule-based approach for multi-type spoilers in order to generate spoiler answers.

1 Introduction

Clickbait involves a text, usually accompanied by a thumbnail or a link, specifically designed to invoke the interest or curiosity of a user. This is done with the intent of enticing the user into clicking the text in order to satisfy their created curiosity. In Task 5 from SemEval-2023, the Clickbait Challenge engages with the study of multi-class classification and extractive question answering in respect to these aforementioned clickbait posts (Hagen et al., 2022).
For this purpose, the task suggests the act of clickbait spoiling, which involves the generation of a short text that answers this supposed curiosity gap (Fröbe et al., 2023a). While models engaging in clickbait spoiling can be used for the purpose of saving a user a click on clickbait posts, the application of models able to question answer a set of questions reliably extends to many more use cases such as answering search requests formed as questions in classic information retrieval, making insights gained from question answering models valuable.

In order to facilitate the study, task 5 is divided into two subtasks:

Sub-task 1: Spoiler Type Classification
This sub-task involves the automated classification of clickbait posts into the category of

- **phrase**: a post that can be answered with a single phrase such as persons, dates or organisations
- **passage**: a post that requires an entire passage to be answered
- **multi**: a post that can be answered with a list or an enumeration of items

Therefore, sub-task 1 can be identified as a **multi-class classification** problem.

Sub-task 2: Spoiler Generation
This sub-task involves satisfying the curiosity of a proposed user via question answering.
This is done by inspecting the clickbait post and the linked content for relevant passages and using those passages to generate the spoiler answer, making **extractive question answering** a possible solution for this task.

Our main strategy for tackling both of these sub-tasks involves the usage of transformer models such as RoBERTa and using transfer learning to adapt the original base model to their respective sub-task. For sub-task 1, this entails the usage of the RoBERTa model in combination with NER, as well as input reformulation and custom metrics such as spoiler-to-title ratio or enumeration checks with the aim to create a more accurate and domain-specific spoiler classification model. For

sub-task 2, this entails the usage of a RoBERTa model fine-tuned using the SQuAD2.0 data set in combination with a contextual rule-based approach for identifying multi-spoiler answers in order to generate appropriate spoiler answers for clickbait posts.
The code for both tasks has been submitted to TIRA as fully tested and functional docker images and can be assessed there (Fröbe et al., 2023b). Alternatively, the docker images and their respective code can also be inspected in the GitHub Repository 'DSC_ANLP'[1].

2 Background

For the Clickbait Challenge, Task 5 provided participants with a high-quality annotated data set called the 'Webis Clickbait Spoiling Corpus 2022' that serves as the basis for the systems, essentially used for training and evaluating resulting models (Fröbe et al., 2022). It contains 5000 clickbait posts crawled from social media platforms such as Facebook, Reddit and Twitter, accompanied by complete annotations and labels about their respective type of spoiler in sub-task 1 and the manually created spoiler answers for sub-task 2.
In specifics, these clickbait posts contain manually cleaned excerpts from the articles behind the links and also classifies which exact passages classify as categorized spoilers. To aid participants in classifying and generating spoilers, all these posts and their respective spoilers are classified into the three types, including short phrase spoilers, long passage spoilers and multi spoilers, while the spoiler answers are mainly categorized as extractive spoilers, with abstractive, rephrased spoilers being the absolute minority.

Table 1: Class distribution in the Webis Clickbait Spoiling Corpus 2022

Class	Train	Validation	Test
Phrase	1367	335	undisclosed
Passage	1274	322	undisclosed
Multi	559	143	undisclosed
Total	3200	800	1000

Dataset The dataset is provided in a predefined train, validation and test split, with the test split being kept private until the end of the shared task. The overall spoiler type distribution, categorized by their respective split, is further defined in Table [1]. Each post contains various amount of fields that help identifying posts and their respective context and spoilers, as shown in detail in Table [2], with the grand majority of present language of the posts being in the English language domain.

Table 2: Relevant fields in the Webis Clickbait Spoiling Corpus 2022

Field Type	Related Field
Identifiers	[uuid], [postID]
Source	[source],[postPlatform], [targetMedia], [targetUrl]
Context	[postText],[targetParagraphs], [targetTitle],[targetDescription], [targetKeywords]
Task-related Field	[Spoiler], [Tags]

3 System Overview

In this section, we will discuss the various components that are included in the implementations of our systems for sub-task 1 and sub-task 2 in closer detail.

3.1 Sub-task 1: Spoiler Classification

In sub-task 1, the task entails the classification of a given clickbait posts into the three spoiler types phrase, passage or multi, making it a **multi-class classification problem**. In order to approach this problem, our system called **RoBERTa-NER** involves the use of four different components that are used as seen in Figure [1]:

- RoBERTa
- NER
- Spoiler-Title Ratio
- Enumeration Check
- Input Reformulation

RoBERTa As the core of our approach in sub-task 1, we chose to use the transformer model approach for solving the multi-class classification problem. For this we aimed to employ a pre-trained transformer model and use it for transfer learning on sub-task 1 as the downstream task. Ultimately, our group chose to use the RoBERTa base model

[1] https://github.com/AH-Tran/DSC_ANLP

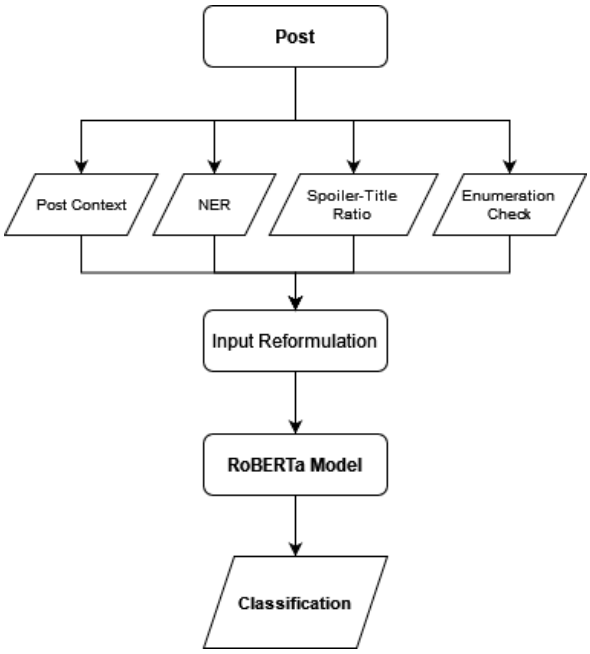

Figure 1: Overview of the system for sub-task 1

as provided by Hugging Face (Liu et al., 2019). In essence, RoBERTa functions as an adaptation of BERT with modifications on the pre-training steps, masking and batch sizes. In specific, it consists of 12 transformer layers with 12 self-attention heads per layer. In difference to the BERT model, RoBERTa base is pre-trained on a comparatively larger corpus of English data. This data consists of a larger union of data, which incidentally also includes large data sets such as:

- **CC-news**: a dataset containing over 63 million English news articles
- **OpenWebText**: a WebText dataset created in Open Source

Since the Webis Clickbait Spoiling Corpus largely deals with social media posts an news articles, RoBERTa as a pre-trained model seems more task-relevant for the downstream task than the original BERT model.

NER In order to enrich our data set, we aimed to create additional features in the form of 'Named Entity Recognition' (NER) for the specific purpose of exploring their usage in identifying and classifying spoiler types. In our system, named entities are categorized into the following types, as seen in Figure [2]:

- Organisations (ner_orgs)
- Persons (ner_persons)
- Dates (ner_dates)
- Locations (ner_locations)

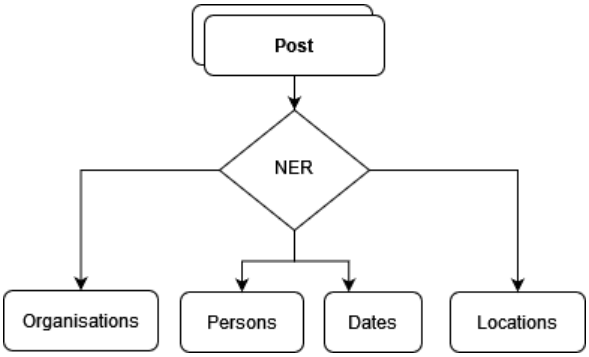

Figure 2: Overview of NER on a given post

This is achieved by taking the whole context of the posts from the article and posts of a clickbait post and using spaCy to extract relevant named entities out of the context (see Table [3]).

Table 3: Example of postId '42800616490403430 'and its extracted named entities

postId	ner_orgs	ner_persons	ner_dates	ner_locations
42[...]05	[Youtube]	[Kyle, Josh]	[this week]	[]

All of this is done under the assumption that allowing the language model to recognize the feature difference between normal text and special entities would be potentially beneficial for the multi-class classification of spoiler types.

Enumeration Check This additional feature was implemented under the assumption that applying a simple regex check for enumeration and lists on the whole context of a clickbait post allows for a more simple classification of multi-type spoilers.

Spoiler-Title Ratio This feature is an additional measure that inspects the length of a title (postText) in relation to the length of its related full article (targetDescription) in a normalized ratio between 0 and 1 and is calculated as follows:

$$Spoiler\_Title\_Ratio[i] = \frac{postText[i]}{targetDescription[i]}$$

This is done under the assumption that:

- **Passage Spoilers** likely have a lower spoiler-title ratio(i.e. Spoiler_Title_Ratio < 0.5)
- **Phrase Spoilers** having a high spoiler-title ratio (i.e. Spoiler_Title_Ratio > 0.5)

where we assume that passages are innately longer than titles while phrases are innately shorter in comparison to titles.

Input Reformulation Furthermore, our paper experimented with the use of Input Reformulation in the context of transformer models. This was done under the assumption that it might be advantageous to feed the resulting RoBERTa transformer model natural sentences enriched with all additional features and named entities.
Initial experiments involved adding each new named entity by their individual name as seen in Table [3]. This, however, resulted in exponentially increasing the string size of each data entry, especially for long passage spoiler-types, making it unfeasible to utilize all features in a reasonable maximum length size when training the RoBERTa model. To alleviate this, we then opted to simply mention named entities as simple number counts instead of individual entities (see Table [4]).

Table 4: Example of postId '428006164904034305 'and its extracted named entities in countable form

postId	ner_orgs	ner_persons	ner_dates	ner_locations
42[...]05	[1]	[2]	[1]	[0]

This allowed us to capture named entity information in shorter string sizes at the cost of specific in-depth named entity information where the system transforms the data of a specific post into a natural sentence complete with all calculated features (see Figure [3]).

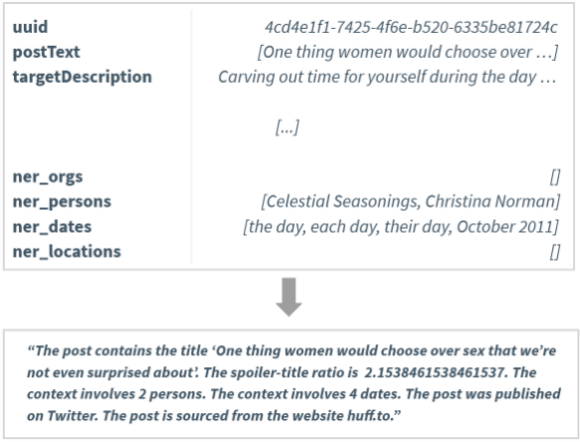

Figure 3: Example of a Long-Form Input Reformulation of a post's dataframe into a natural sentence

Alternatively, we also experimented with shortening the resulting natural sentence even further, making it more alike to a simple enumeration of fact features, as seen in Figure [4].
The short sentences are then used as the input for the RoBERTa model during the training process and any input will be subject to the same preprocessing on the final model when assigning the spoiler class for sub-task 1.

Figure 4: Example of a Short-Form Input Reformulation of a post's dataframe into a natural sentence

3.2 Sub-task 2: Spoiler Generation

Sub-task 2 of the Clickbait challenge requires a system to generate appropriate **spoiler answers** in response to a given clickbait post.
For this, we make use of a similar approach as in sub-task 1, where we utilize the transformer model approach and apply transfer learning to train the model for the downstream task of **extractive question answering**. This approach is then combined with the implementation of a simple rule-based approach based on the identification of multi-type spoilers via regex. The system overview for our approach can be seen Figure [5].

RoBERTa In order to generate spoiler answers, we utilize the RoBERTa Base SQuAD2.0 transformer model as provided by deepset. It is initially trained on question-answer pairs for the purpose of enabling downstream tasks involving extractive question answering, making it an appropriate choice for sub-task 2 (Chan et al., 2022). The purpose of this is similar as seen in section 3.1, where a larger training data set on more relevant data sources potentially enable a more performant model for this domain. In comparison to the RoBERTa model used in task 1, this model is fine-tuned using the SQuAD2.0 dataset, which will be defined in the following subsection.

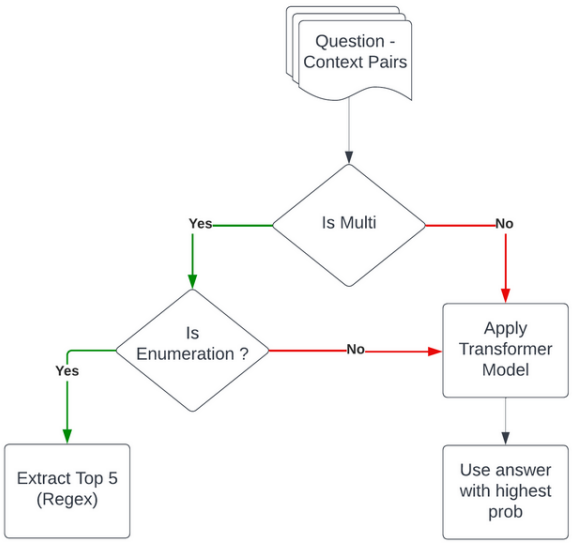

Figure 5: Overview of the system for sub-task 2

SQuAD2.0 The Standford Question Answering Dataset (SQuAD) first introduced by the Stanford NLP Group is a reading comprehension dataset, consisting of questions where the answer to every question is either a segment of text or a span from the corresponding reading passage (Rajpurkar et al., 2016). It, however, also combines over 100,000 questions with over 50,000 unanswerable questions, requiring question answering systems to not only answer questions but to also decide whether the question is answerable in the first place and abstain from answering if it is indeed unanswerable and none of the generated answers manage to reach a satisfying score threshold.
The usage of this dataset requires the corpus with its posts as questions and the annotated spoiler answers to be reformatted into the proper SQuAD2.0 format, requiring fields such as [title], [questions], [answers], [is_impossible]-flags and [context]. An example of a data entry of the Webis Corpus in the SQuAD2.0 format can be found in the Appendix as Figure [7].

Rule-based Approach Initial testing of the models recognized good performance for phrase and passage spoilers, but subpar performance on multi-type spoilers. For this reason, we proposed the usage of an enumeration check, similar to the one mentioned in Section 3.1.
In sub-task 2, the enumeration check is used as an additional rule-based approach, where it checks whether a question is answerable by either an enumeration or list of items, as denoted in Figure [5] under the 'Is Enumeration¿ decision field. After a positive determination, the system then checks the 'targetParagraph' field of the corpus for further enumerations and listings, and uses them as the accepted generated spoiler answer. These regex steps work as denoted in Figure [6].

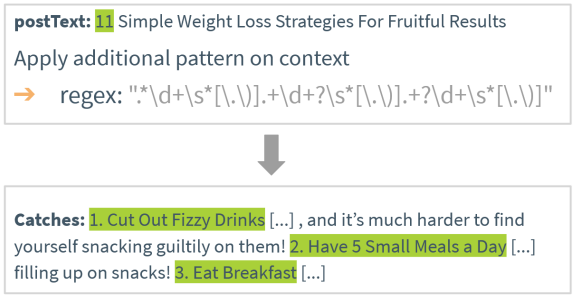

Figure 6: Regex application example on the postText field capturing enumeration and list

With the introduction of the rule-based approach, we ultimately submitted two different systems, denoted as **v1** and **v2**:

- **v1**: A simple transformer-model approach using a trained RoBERTa-SQuad2.0 model
- **v2**: The transformer-model approach from **v1**, modified with **rule-based logic** for multi type spoilers

4 Experimental Setup

This section explains the experimental setups that were used for implementing our systems for **sub-task 1 spoiler classification** and **sub-task 2 spoiler generation**.

4.1 Sub-task 1: Spoiler Classification

In order to train the RoBERTa model for the purpose of multi-class classification, we used the original dataset, involving the training and validation dataset. Due to the test dataset being undisclosed until after the shared task finishes, we made the conscious decision to split the validation dataset into two separate datasets with a 50:50 split and a randomized seed, resulting in an overall data split of 3200 for the training dataset, 400 for the validation dataset and 400 for the improvised test data set as seen in Table [5].

Table 5: Data split distribution for sub-task 1 & sub-task 2

Task	Train	Validation	Test
Sub-task 1	3200	400	400
Sub-task 2	3200	800	-

Preprocessing For the preprocessing, we use the following fields in order to create the 'context':

- **[postText]**
- **[targetParagraphs]**

By utilizing these fields as context, we can extract named entities with the NER module from spaCy. Afterwards, we inspected the ratio between the [postText] and the [targetDescription] to calculate the 'Spoiler-Title Ratio' as seen in Section 3.1. Finally, we use short input reformulation in order to transform the enriched data frame into a natural sentence, which we fed to the trained model as input.

Hyperparameters For the purpose of training the initial RoBERTa model for the downstream of multi-class classification task on the enriched data frame, we use the following hyperparameter set-up:

```
batch_size = 24
doc_stride = 192
max_seq_length = 300
learning_rate = 1e-5
num_train_epochs = 4
```

All other settings were used under the standard options as provided by the simpletransformers framework (Rajapakse, 2022).

External Tools & Libraries The implementation of the training procedure was done with the simpletransformers library, a framework that allows the user to easily access and train models for personal use. It is built upon the transformers framework and utilizes models retrieved from Huggingface for standard use (Rajapakse, 2022).
The NER component of this system was enabled by the spaCy module[2], which enables users to use specific libraries and models for many kinds of NLP tasks including the recognition of named entities.
In this paper, we used the en_core_web_sm model, an English language model trained on the OntoNotes 5 dataset (Explosion, 2023) which consists of written text found on the internet such as blogs, news and comments, making it particularly relevant for this sub-task's domain (Consortium, 2023).
Due to the docker image upload to TIRA, we decided to forgo using the larger language models and instead use one of the smaller models available in order to keep the resulting docker image size low while retaining a comparable performance.

[2]https://spacy.io/models/en

4.2 Sub-task 2: Spoiler Generation

For sub-task 2, there were concerns whether the available dataset provided enough samples for creating a performant extractive question answering model. This is why, unlike in sub-task 1, the decision was made to avoid diluting the dataset into separate validation and test splits and instead use the training dataset and validation dataset in full as provided by the challenge organizers.

Preprocessing For the purpose of fitting our training data for the QA task properly, we reformatted the questions into the SQuAD2.0 format as previously defined in Section 3.2 and Figure [7]. For the training procedure, the paragraphs were reassembled in order to train the model on the complete full text.

Hyperparameters In the training process of the initial RoBERTa model on the reformatted Webis Corpus for extractive question answering, we use the following hyperparameter set-up:

```
language_model = roberta-based-squad2
batch_size = 24
n_epochs = 5
max_seq_len = 384
doc_stride = 192
embeds_dropout_prob = 0.1
learning_rate = 3e-5
schedule_opts = {linearWarmup, 0.2}
```

External Tools & Libraries In order to train the RoBERTa model for the extractive question answering task, we utilized the FARM framework for implementing the system (deepset ai, 2022). It offers a variety of datasets and functions that enable an easier process into creating custom models for machine learning tasks such as question answering.

Table 6: Overview of the effectiveness in spoiler type prediction (subtask 1 at SemEval 2023 Task 5) measured as balanced accuracy over all three spoiler types and precision (Pr.), recall (Rec.), and F1 score (F1) for phrase, passage, and multi spoilers on the test set.

Submission			Accuracy	Phrase			Passage			Multi		
Team	Approach	Run		Pr.	Rec.	F1	Pr.	Rec.	F1	Pr.	Rec.	F1
billie-newman	RoBERTa-NER	2023-01-25-22-33-38	0.59	0.78	0.30	0.43	0.60	0.83	0.70	0.40	0.65	0.50

Table 7: Overview of the effectiveness in spoiler generation (subtask 2 at SemEval 2023 Task 5) measured as BLEU-4 (BL4), BERTScore (BSc.) and METEOR (MET) over all clickbait posts respectively those requiring phrase, passage, or multi spoilers on the test set.

Submission			All			Phrase			Passage			Multi		
Team	Approach	Run	BL4	BSc.	MET	BL4	BSc.	MET	BL4	BSc.	MET	BL4	BSc.	MET
billie-newman	v1	2023-01-24-17-01-57	0.31	0.89	0.29	0.49	0.92	0.23	0.20	0.87	0.31	0.12	0.87	0.28
billie-newman	v2	2023-01-24-17-37-56	0.32	0.90	0.30	0.49	0.92	0.23	0.20	0.87	0.31	0.20	0.88	0.34

5 Results

In this section, we will discuss the performance of our submitted systems. It involves the **RoBERTa-NER** system outlined in Section 3.1 and both the **v1** and the **v2** systems outlined in Section 3.2 for sub-task 1 and 2 respectively.
All systems were submitted to the TIRA platform as executable docker images for the sake of reproducibility and were compared against the performance of the naive and transformer baselines as provided by the task organizers, where the naive baseline for sub-task 1 simply always predicts a ‘passage‘ spoiler-type for each classification task and spoils each clickbait post with the title of the linked page in sub-task 2 [3].

5.1 Sub-task 1: Spoiler Classification

In Table [8] we can see the performance of our system **RoBERTa-NER** in comparison against the naive and transformer baseline on the **validation dataset**.

Table 8: Comparison between the system ‘RoBERTa-NER‘ against the baseline on the validation dataset

Model	Balanced Accuracy (in %)
Naive (Baseline)	33.3
Transformer (Baseline)	73.4
RoBERTa-NER	58.87

While it outperforms the naive baseline, which simply predicts a single class for every input, it does not manage to outperform the simple transformer baseline provided by the challenge organizers. This leads us to believe that enriching the dataset with NER entities for multi-class classification does not lead to better performance, at least in our implementation. Other arguments could be found in the reasoning that our decision to simply capture named entities as enumerations instead of individual names for the sake of keeping string sizes low led to an information loss within the dataset that worsened the performance of the system.

The overview of the overall performance of the system **RoBERTa-NER** for sub-task 1 on the **test dataset** can be found in Table [6], reaching an overall accuracy of almost 59 % over all spoiler types. When inspecting the system’s performance for each spoiler type, we can see that it reached the highest **[precision]** while predicting phrases at 78 %. This, however, is also accompanied by a very low **[recall]** at 30 %, implying that while the system managed to make a lot of correct positive predictions for phrase-type spoilers, it also assigned a lot of false negatives, essentially labelling 70 % of actual phrase-type spoilers as something else. This discrepancy is also present at the **[F1]** score on phrase-type spoilers, sitting at the value of 0.43.
Meanwhile, the system managed to perform overall the best when it comes to classifying passage spoiler-types, performing a **[precision]** of 60 %, a **[recall]** of 83 % sitting at an overall **[F1]** score of 0.70.
For multi-type spoilers, it performs overall at the

[3] https://github.com/pan-webis-de/pan-code/tree/master/semeval23/baselines

average of 0.5 for the **[F1]** score. While it obtains a higher **[recall]** than for phrase spoiler-types of 65 %, its **[precision]** for multi-type spoilers stands at a low 40 %.
This could be explained by assuming that using the enumeration-check feature for performing this multi-class classification task leads to a high case of false positives, where the system falsely classifies actual phrase or passage spoiler-types as multi-type spoilers.

5.2 Sub-task 2: Spoiler Generation

In Table [9] we can see the performance of our systems using the RoBERTa-SQuAD2.0 transformer model in comparison to the provided baselines on the **validation dataset**.
The two different systems, denoted as **v1** and **v2**, were implemented as previously explained in Section 3.2.

Table 9: Comparison between the system v1 and v2 against the baseline on the validation dataset

Model	BLEU Score
Naive (Baseline)	0.021
Transformer (Baseline)	0.382
RoBERTa v1	0.3171
RoBERTa v2	0.3258

While both systems manage to exceed the naive baseline and they both also fail to outperform the basic transformer baseline.
However, we can see a clear difference between the **v1** and **v2** system. The **v2** system achieves a slightly higher BLEU score of 0.3258, resulting in a roughly 1 % performance increase in comparison to the **v1** system.
Given that the only difference between the systems is that the **v2** system employs the rule-based approach for handling multi-type spoiler answers outlined in Section 3.2, while the **v1** system does not, it seems plausible to assume that this specific feature caused this performance difference.

In Table 10, we can observe the performance of our sub-task 2 systems on the **test dataset** concerning the metrics BLEU-4 (BL4), BERTscore (BSc.) and METEOR (MET), where the systems reach an overall **[BL4]** score of 0.3171 for v1 and 0.3258 for v2 with the rule-based approach respectively.
When comparing the **v1** and **v2** system with each other, it becomes apparent that the **v2** system manages to set itself slightly apart from the former with slightly higher **[BL4], [BSc.] and [MET]]**. This difference is caused by **v2's** better performance for generating appropriate multi-type spoiler answers, where we can see a larger increase from 0.12 to 0.20 in **[BL4]**, from 0.28 to 0.34 in **[MET]** and a smaller increase from 0.87 to 0.88 in **[BSc.]**. Similar to how it was the case in the validation set, the inclusion of the rule-based approach for multi-type spoiler answers leads to a sizeable performance increase when it comes to generating multi-type spoiler answers, affirming the previous assumption.

6 Conclusion

All in all, our systems did not manage to outperform either of the basic transformer baselines provided by the challenge organizers. In sub-task 1, the inclusion of features based on named entities, spoiler-title ratios, multi-enumeration checks and input reformulation led to a decrease in average accuracy in comparison to the basic transformer baseline, while in sub-task 2 we saw a more minor decrease in BLEU score when compared to the provided baseline. The visible decrease of performance in sub-task 1 in comparison to the transformer baseline could possibly be explained by the implementation of the spoiler-title-ratio metric, which was not rounded after the second decimal. Thus, it might have led the model to being trained and possibly overfitted on very precise numbers up to the 16th or further decimal, therefore causing the model to not perform well on the test set.
For sub-task 2 we submitted two different systems: **v1** without additional rule-based logic and **v2** with additional rule-based logic for multi type spoilers. Upon inspection of the final results, we see a minor performance increase in the latter **v2** system on both the validation and test set, When inspecting and comparing the systems in respect to multi-type spoiler answers in detail, we can see an even larger performance difference between the **v1** and **v2** systems, overall suggesting that the addition of rule-based logic in combination with the transformer model approach had a positive effect on the observed performance scores.
For future research, it would make sense to further delve into the positive insights gained in this paper. Originally, our team planned to implement a complete rule-based approach for question answering covering all types of spoilers by following

a simple logic routine explained in papers such as (Riloff and Thelen, 2000). The approaches in this paper, however, did not seem very suitable for the task at hand, as its context is centered around rule-based question answering approaches for elementary school types of questions involving reading comprehension test, making it an inherently different use case than the use case found in the SemEval-2023 Task 5 context involving news- and blog posts functioning as clickbait.

Seeing how the simple rule-based logic addition in sub-task 2 led to a small positive increase in performance across the board, would suggest that further research into implementing rule-based logic such as taking into account semantic or syntactic patterns, in conjunction with transformer-based approaches might be worth looking into for further research.

Finally, it might also be useful for further research to find ways to combine the features of sub-task 1 and sub-task 2 together in one unified system, where a reliable multi-class classification model predicts the spoiler type correctly, which then dictates how and which rule-based approach the question answering system uses for generating the appropriate spoiler answer.

7 Acknowledgments

For this project we wish to acknowledge and thank Maik Froebe for the continuous and quick assistance in interfacing with TIRA and submitting our system to the TIRA platform successfully as fully operational docker images.

References

Branden Chan, Timo Moeller, Malte Pietsch, and Tanay Soni. 2022. roberta-base for qa. [last accessed: 24.02.2023].

Linguistic Data Consortium. 2023. Ontonotes release 5.0. [last accessed: 24.02.2023].

deepset ai. 2022. Farm - framework for adapting representation models. [last accessed: 24.02.2023].

Explosion. 2023. English - available trained pipelines for english. [last accessed: 24.02.2023].

Maik Fröbe, Tim Gollub, Matthias Hagen, and Martin Potthast. 2022. Webis clickbait spoiling corpus 2022. [last accessed: 24.02.2023].

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.

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.

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. *CoRR*, abs/1907.11692.

Thilina Rajapakse. 2022. Classification models. [last accessed: 24.02.2023].

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Ellen Riloff and Michael Thelen. 2000. A rule-based question answering system for reading comprehension tests. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Sytems - Volume 6*, ANLP/NAACL-ReadingComp '00, page 13–19, USA. Association for Computational Linguistics.

A Appendix


```
"title": "You'll Never Believe What This Family Saw in the Sky Outside Their House in Finland.",
"paragraphs": [
  {
    "qas": [
      {
        "question": "You'll Never Believe What This Family Saw in the Sky Outside Their House in Finland.",
        "id": "c4e0edb3-16f0-49eb-b3a5-cdd24c546c74",
        "answers": [
          {
            "text": "rainbow colours in the sky and a halo spanning 360 degrees",
            "answer_start": 138
          }
        ],
        "is_impossible": false
      }
    ],
    "context": "\"It was cold and very foggy, the temperature was around -10 degrees Celsius,\" said
     Hänninen. \"When the clouds began to break, there were rainbow colours in the sky and a halo spanning
     360 degrees! It was worth taking a picture or two.\" If I ever stepped outside and saw this in my
     backyard, I might think the aliens were invading! "
  }
]
```


Figure 7: Example of a data entry of the Webis Corpus formatted into the SQuAD2.0 format