TohokuNLP at SemEval-2023 Task 5: Clickbait Spoiling via Simple Seq2seq Generation and Ensembling

Hiroto Kurita[1†], **Ikumi Ito**[1†], **Hiroaki Funayama**[1,2], **Shota Sasaki**[3,2‡], **Shoji Moriya**[1], **Ye Mengyu**[1], **Kazuma Kokuta**[1], **Ryujin Hatakeyama**[1], **Shusaku Sone**[1], **Kentaro Inui**[1,2]

[1]Tohoku University [2]RIKEN AIP
[3]CyberAgent, Inc.
{hiroto.kurita,ikumi.ito.p8,h.funa,shoji.moriya.q7,ye.mengyu.s1,
kokuta.kazuma.r3,hatakeyama.ryujin.q7,sone.shusaku.r8}@dc.tohoku.ac.jp
sasaki_shota@cyberagent.co.jp kentaro.inui@tohoku.ac.jp

Abstract

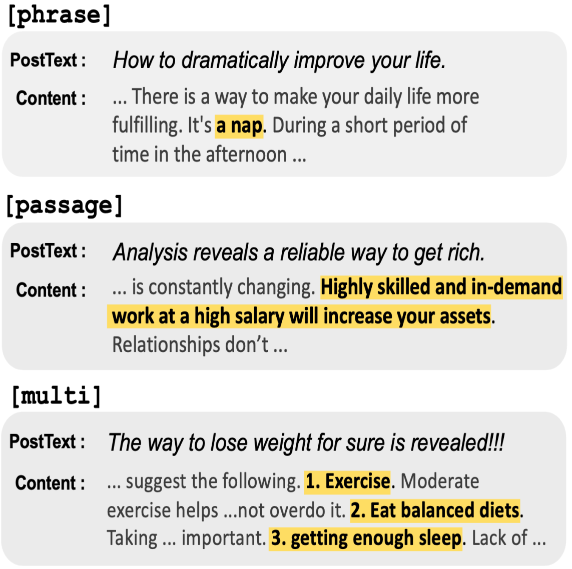
This paper describes our system submitted to SemEval-2023 Task 5: Clickbait Spoiling. We work on spoiler generation of the subtask 2 and develop a system which comprises two parts: 1) simple seq2seq spoiler generation and 2) post-hoc model ensembling. Using this simple method, we address the challenge of generating multipart spoiler. In the test set, our submitted system[1] outperformed the baseline by a large margin (approximately 10 points above on the BLEU score) for mixed types of spoilers. We also found that our system successfully handled the challenge of the multipart spoiler, confirming the effectiveness of our approach.

1 Introduction

Clickbait is a type of text on social media that is specifically designed to exploit users' curiosity and lure them into clicking on a linked webpage. Due to the deceptive and misleading nature of the content often presented, clickbait is generally considered harmful (Hagen et al., 2022). Thus addressing the clickbait issue is a crucial concern for social media users.

Subtask 2 of SemEval-2023 Task 5 aims to generate spoilers of clickbait based on posted text and the linked webpage (Fröbe et al., 2023a). Hagen et al. (2022) proposed a method for generating spoilers via encoder-based question answering approach. However, their method has limitations in handling a specific type of spoiler where the answer is spread across several sections of the target article.

To tackle this problem, we develop a simple yet effective sequence-to-sequence (seq2seq) approach for the subtask 2: spoiler generation. In our approach, the target spoiler is directly generated from the seq2seq model, rather than predicting its spans in the context. This enables us to handle the challenge of multipart spoiler with ease without implementing more complex approaches such as multiple-span extraction (Segal et al., 2020). Experimental results demonstrate that our approach successfully addresses the multipart spoilers, leading to an overall score improvement without degrading the scores of single-part spoilers. Furthermore, we incorporate a post-hoc ensemble of models with multiple seeds in our system, leading to enhanced performance over a single model.



[phrase]

PostText : *How to dramatically improve your life.*

Content : ... There is a way to make your daily life more fulfilling. It's **a nap**. During a short period of time in the afternoon ...

[passage]

PostText : *Analysis reveals a reliable way to get rich.*

Content : ... is constantly changing. **Highly skilled and in-demand work at a high salary will increase your assets**. Relationships don't ...

[multi]

PostText : *The way to lose weight for sure is revealed!!!*

Content : ... suggest the following. **1. Exercise**. Moderate exercise helps ...not overdo it. **2. Eat balanced diets**. Taking ... important. **3. getting enough sleep**. Lack of ...

Figure 1: Overview of SemEval-2023 Task 5: Clickbait Spoiling. The task is to extract one or more spoilers for `postText` from content (`targetTitle` and `targetParagraphs`). The text highlighted with yellow in the content is the spoiler to be extracted.

In analysis, we show that our system generates spoilers extractively rather than abstractively, even though we adapt the seq2seq generation approach. We also examine the errors made by our system and discuss the characteristics and potential difficulty of this task.

[†] equal contributions

[‡] Work done while at RIKEN AIP and Tohoku University.

[1] The docker image of our final submission on TIRA(Fröbe et al., 2023b) is `registry.webis.de/code-research/tira/tira-user-john-king/finalsub:0.0.4` and the implementation is available at `https://github.com/cl-tohoku/semeval2023-task5-seq2seq-ensemble`.

Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023), pages 1756–1762
July 13-14, 2023 ©2023 Association for Computational Linguistics

2 Task Description

The SemEval-2023 Task 5 organizers offer the Webis Clickbait Spoiling Corpus 2022 dataset (Hagen et al., 2022). This dataset collects clickbait posts on social media (Facebook, Reddit, and Twitter), linked web pages in the posts, and various related information. The following information in the dataset is employed in this task (2023):

- `postText`: The clickbait posted on social media that entices a click.
- `targetTitle`: The title of the linked article on `postText`.
- `targetParagraphs`: The main content of the linked article on `postText`.
- `spoiler`: The human-extracted clickbait spoiler for `postText` either from `targetTitle` or `targetParagraphs`.

The task is to extract `spoiler` from `targetTitle` and `targetParagraphs` that spoils the clickbait post of `postText` (Figure 1). `spoiler` to be extracted are classified into the following three categories based on their structures (2023):

- `phrase`: Spoiler of single word or phrase.
- `passage`: Spoiler composed from continuous sentences.
- `multi`: Spoiler comprised of multiple discontinuous words or sentences.

The provided training data consist of 3200 samples. The counts for each type in the training data are 1367 (around 43%) for `phrase`, 1274 (around 40%) for `passage`, and 559 (around 17%) for `multi`. The agreement between the model's predictions and the ground truths is evaluated based on the following three metrics: BLEU-4 (Papineni et al., 2002), BERTScore (Zhang* et al., 2020), and METEOR 1.5 (Denkowski and Lavie, 2014), an extended version of METEOR (Banerjee and Lavie, 2005).

3 Related Work

Hagen et al. (2022) employ an encoder-based question answering model for clickbait spoiling. They treat the target article as context and posted clickbait as question. Then the encoder-based question answering model predicts the spans of the target spoiler in the context. However, their method only assumes one correct span; thus, it cannot handle the challenge of generating multipart spoilers where answers have multiple spans. Several studies in the field of question answering have tackled the issue of answering multi-span questions. For instance, Segal et al. (2020) have formulated the question answering task as a token classification problem, where each token is classified as either an answer or not. However, implementing these multiple-span extraction methods can be complex. In this work, we propose a simple seq2seq generation approach to address the challenge of multipart spoiler generations, avoiding complex implementations.

4 System Overview

This section presents an overview of the proposed system. Our system generates spoilers in two steps; first, multiseed seq2seq models generate spoilers, and then the generated outputs are post-processively ensembled using a similarity-based ensemble approach.

4.1 Seq2seq Spoiler Generation

Seq2seq models take an input sequence and generate an output sequence directly. We use this seq2seq generation method for spoiler generation in a question answering manner. Given input sequence, which is concatenations of `postText`, `targetTitle` and `targetParagraphs`, the seq2seq model is expected to generate a sequence of target spoiler directly.

Input format. The input is provided in the form of "question : [`postText`] context : [`targetTitle`] - [`targetParagraphs`]". Since we use seq2seq models that have been partially trained with question answering during pre-training, incorporating question and context prefixes could help generate spoilers by using their inherent question answering abilities.

Output format. The model generates the sequence for the target spoilers directly. As for multipart spoiler, the model generates a concatenated sequence of several spans with white space since they have several answer spans.

With this simple seq2seq approach, we can address `phrase`, `passage` and `multi` spoilers simultaneously without implementing more complex methods (e.g., multiple-span extraction).

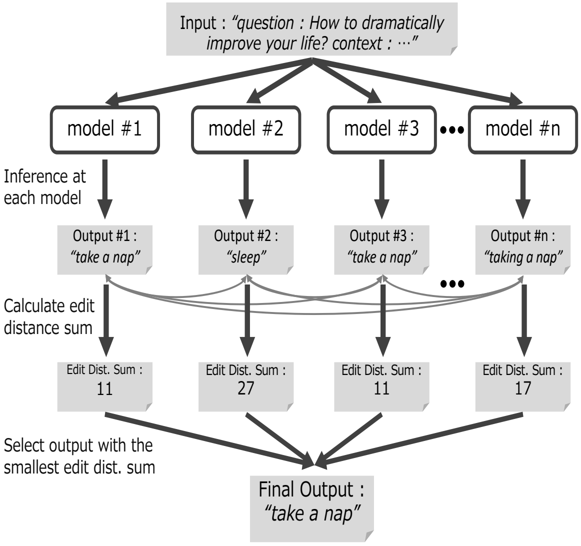


Figure 2: Overview of our post-poc model ensembling following Kobayashi (2018). First, each model generates its output. Then, the ensemble algorithm computes the edit distance for all combinations among the outputs of each model. Finally, the output with the smallest total edit distance to the other outputs is chosen as the system output.

4.2 Post-Hoc Model Ensembling

Some examples that cannot be successfully generated by a single model may be effectively generated by models trained with different seeds. Kobayashi (2018) has demonstrated that a simple post-hoc model ensembling approach inspired by majority voting in classification tasks also works effectively in generation tasks. We incorporate this post-hoc ensembling method in our system to select a majority-like spoiler prediction among outputs from different seed models (Figure 2). Algorithm 1 outlines the adapted ensembling approach. First, the algorithm takes predicted spoilers from various seed models as inputs, calculates the edit distances of each predicted spoiler, and sums the computed distances over each model. The final output is chosen by the spoiler with the smallest summed edit distances. Kobayashi (2018) employ cosine as a similarity function, however, we use edit distance in our system for simplicity.

5 Experiment

5.1 Base Model Selection

To choose a seq2seq model for our system, we trained various seq2seq models on the given training data and evaluated their performance.

Algorithm 1 Post-hoc ensemble algorithm

```
Input: Set $X$ of predicted spoilers from different
    seed models, function EditDist which calculates
    edit distance
Output: Selected spoiler $y$
  $\mathrm{distances} \leftarrow \{\}$
  for $x \in X$ do
      $\mathrm{distSum} \leftarrow \sum_{x' \in X, x \neq x'} \mathrm{EditDist}(\mathrm{x}, \mathrm{x'})$
      $\mathrm{distances}[x] \leftarrow \mathrm{distSum}$
  end for
  $y \leftarrow \mathrm{argmin}(\mathrm{distances})$
  return $y$
```

Table 1: Score comparison of different model types and sizes. All models are evaluated on the validation data. `phrase`, `passage`, and `multi` represent the scores when the predictions are evaluated only on each spoiler. `all` represents the score when all spoiler types are evaluated together. All values of the evaluation metrics are multiplied by 100.

	BLEU-4	BERTScore	METEOR
`t5-base`			
`- all`	38.68	91.35	39.90
`t5-large`			
`- all`	43.87	92.34	48.72
`flan-t5-base`			
`- all`	39.53	91.45	36.85
`flan-t5-large`			
`- all`	**47.09**	**92.77**	**51.24**
`- phrase`	62.91	94.85	41.19
`- passage`	33.75	90.98	45.77
`- multi`	40.07	91.93	62.50
`DeBERTa`			
`- all`	38.20	91.47	40.37
`- phrase`	65.52	95.46	53.54
`- passage`	22.54	89.11	42.27
`- multi`	9.45	87.44	29.24

Settings. We used two pre-trained seq2seq models, T5 (Raffel et al., 2020) and Flan-T5 (Chung et al., 2022) both for base and large size. We selected these pre-trained models to leverage the knowledge and capabilities acquired during their multitask pre-training, including question answering. We fine-tuned these seq2seq models with the given training dataset. As a baseline, we used a large size of DeBERTa (He et al., 2021) fine-tuned for the question answering task, provided by the task organizers[2]. This model was first fine-tuned on SQuAD (Rajpurkar et al., 2018) and then further trained on the clickbait dataset. Training details for the seq2seq models are showed in Appendix A.

[2] https://github.com/pan-webis-de/pan-code/tree/master/semeval23/baselines/transformer-baseline-task-2

Table 2: Ensemble effectiveness on the test data. phrase, passage, and multi, and all represent same as in Table1. All values of the evaluation metrics were multiplied by 100. We selected the Ensemble setting for the final submission.

	BLEU-4	BERTScore	METEOR
Single			
- all	47.86	92.94	48.35
- phrase	65.59	95.22	39.51
- passage	30.89	90.64	41.79
- multi	44.08	92.70	63.95
Ensemble			
- all	**48.25**	**93.05**	**49.71**
- phrase	65.91	95.30	40.30
- passage	31.56	90.80	43.44
- multi	43.96	92.79	64.83

Results. Table 1 presents the performance of each model on the validation data. For all setting, the large-size seq2seq models consistently outperformed their base-size counterparts. Furthermore, when comparing models of the same size, Flan-T5 consistently outperformed T5. Based on these findings, we selected the large-size of Flan-T5 as the base model for subsequent experiments. Comparing large-size of Flan-T5 and DeBERTa, Flan-T5 outperformed DeBERTa on all and passage, but underperformed on phrase. As for multi, the Flan-T5 model showed competitive performance on all metrics, confirming that our approach successfully handled the multipart spoiler.

5.2 Post-Hoc Model Ensembling

In this experiment, we investigated the impact of the post-hoc model ensembling introduced in Section 4.2.

Settings. We prepared five Flan-T5 large models trained with different seeds. Then, we applied post-hoc ensembling to the outputs from each model. Training details, including hyperparameters, are provided in Appendix A. We compared the following two settings:

- Single: The average scores of five Flan-T5 large models trained with various seeds.
- Ensemble: The scores of the post-hoc ensembling of five Flan-T5 large models trained with different seeds.

Results. Table 2 presents the performance of the Single and Ensemble on the test data. The findings demonstrated that the ensemble method outperformed the single model, confirming the effectiveness of the ensemble. We submitted this ensemble of five Flan-T5 large models as our final system since we confirmed its effectiveness on the validation data.

Table 3: Effectiveness of spoiler type information on test data. Single is the average performance of five flan-t5-large models of different seeds. Ensemble is the performance when the post-hoc ensemble is adapted to the outputs of the five models. Single and Ensemble do not use spoiler type information. Oracle first determines the type of spoiler with an oracle classifier, then uses a model trained only on the data of each type. All values of the evaluation metrics are multiplied by 100.

	BLEU-4	BERTScore	METEOR
Single	47.86	92.94	48.35
Ensemble	48.25	93.05	49.71
Oracle	**52.14**	**93.73**	**54.64**

6 Analysis

In this section, we examine our submitted system with the model ensembling.

6.1 Extractiveness of Seq2seq Model

Our system relies on the seq2seq models to generate spoilers. Therefore, our approach is not necessarily extractive. However, a natural question arises whether this seq2seq model is generating spoilers in an extractive or abstractive manner since our submitted system produces competitive results. We analyzed the proportion of the model's outputs for phrase and passage included in targetTitle or targetParagraphs[3]. As a result, we discovered that 741 of 826 spoilers were extractively generated from targetTitle or targetParagraphs. This result demonstrates that the model generates spoilers extractively rather than abstractively, indicating that seq2seq models can learn to identify spoilers from given contexts.

6.2 Potential Usage of Spoiler Type Classifier

Hagen et al. (2022) have demonstrated that combining a high-performing spoiler type classifier can enhance the spoiler generation performance. To examine this hypothesis also holds for our system, we evaluated the model performance with and without the spoiler type information. First, we trained

[3] Since the output of the model for multi is generated as a single sentence consisting of multiple spoilers, it is difficult to verify whether each spoiler exists as it is in targetTitle or targetParagraphs. For this reason, we excluded multi from the analysis.

Table 4: Examples of inconsistencies between the model predictions and the gold spoilers in the test data. Type (1) is an example where the gold spoiler includes the model's generation. Type (2) is an example where the generation of the model includes the gold spoiler. Type (3) is an example where the model generates text, which is completely different from the gold spoiler. All three predicted sentences are extractively generated from `targetTitle` or `targetParagraphs`.

Type	PostText	Gold Spoiler	Prediction
(1)	If You Get Dizzy When You Stand Up, Use this Fighter Pilot Secret	clench the muscles in your lower body to push blood back into your upper body	clench the muscles in your lower body
(2)	The surprising way Neanderthals got herpes	When modern humans met Neanderthals in Europe	When modern humans met Neanderthals in Europe, we may have given groups of Neanderthals several harmful pathogens.
(3)	Why are Apple's iPhone icons shaped as rounded squares?	Steve Jobs had been a big fan of shapes with subtle rounded corners	there's something more soothing and welcoming about rounded corners relative to sharp corners

and evaluated three Flan-T5 large models with data of identical spoiler type each. This setting corresponds to the situation where we had a perfect spoiler type classifier and fully used the type information for the spoiler generation. We compared the above setting with our submitted system, which ignores the spoiler type information. Table 3 illustrates the model performances of the two settings on the test data. We discovered that the setting with the oracle classifier (`Oracle`) outperformed the setting without type information (`Single`, `Ensemble`) in all evaluation metrics. This indicates that even in the seq2seq method, spoiler type information can boost the performance of spoiler generation. In addition to combining high-performing spoiler type classifiers, incorporating multi-task learning, where the model learns spoiler classification and generation at the same time, would be future work.

7 Discussion

Table 2 illustrates that the performance of the `passage` is low compared to the other two types of spoilers. We examined the inclusion relationship between the model's predictions and the gold spoilers on the `passage` to analyze how the model generates `passage` spoilers. We classified the errors into three categories: (1) the generated text includes a gold spoiler, (2) gold spoiler includes the generated text, (3) cases except (1) and (2). We had 67 cases for (1), 84 cases for (2), and 204 cases for (3) out of 355 error cases in the test set for `passage`. Table 4 shows the generation examples of the three types of `passage` spoiler errors. For (1), (2), and (3) exemplified in Table 4, although all of them cannot fully capture the answer span, none of these examples can be considered completely wrong, considering the meaning of `postText` and predicted spoilers. This demonstrates that it is challenging to uniquely determine the answer spoiler because of the characteristics of the task and the dataset. It is also assumed that annotations of these spoilers can vary even among the annotators. Therefore, it is important to consider the upper limit of the solvability of this task by measuring the inter-annotators agreement. Understanding where the current spoiler generation system has reached against the upper limit of the task and estimating the potential growth from the gap would be a big step forward for this task.

8 Conclusion

We presented our system submitted to the subtask 2 of SemEval-2023 Task 5: Clickbait Spoiling. Our system comprised two parts: 1) simple seq2seq spoiler generation and 2) post-hoc model ensembling. Experimental findings demonstrated that our system effectively addressed the multispan spoiler issue through seq2seq generation. Additionally, our system tended to generate sentences by identifying the appropriate position in the document, even though it was not specifically trained to extract text. The analysis suggests that integrating reliable type classifiers can further improve the performance of our system. This will be future work.

Acknowledgements

We would like to thank the task organizers for technical supports for the docker submission and the reviewers for helpful feedback. We also thank Keito Kudo for the technical advice. This work was supported by JSPS KAKENHI Grant Number JP22H00524, JST SPRING, Grant Number JPMJSP2114.

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.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models.

Michael Denkowski and Alon Lavie. 2014. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Maik Fröbe, Tim Gollub, Benno Stein, 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.

Webis Group. 2023. Clickbait Challenge at SemEval 2023 - Clickbait Spoiling. `https://pan.webis.de/semeval23/pan23-web/clickbait-challenge.html`.

Matthias Hagen, Maik Fröbe, Artur Jurk, and Martin Potthast. 2022. Clickbait Spoiling via Question Answering and Passage Retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7025–7036, Dublin, Ireland. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION. In *International Conference on Learning Representations(ICLR)*.

Hayato Kobayashi. 2018. Frustratingly Easy Model Ensemble for Abstractive Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4165–4176, Brussels, Belgium. Association for Computational Linguistics.

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.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

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.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A Simple and Effective Model for Answering Multi-span Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System*

Demonstrations, pages 38–45, Online. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Training details

We implemented our experiment by Hugging Face (Wolf et al., 2020). During training, we used linear learning rate scheduler, with a maximum learning rate of 0.0001. We set max-update to 1000 and 2000 updates for base and large size, respectively, and selected checkpoints of 1000 and 1800 updates for each size. For the experiment in Section 5.1, we fixed the model seeds to 42. We used the model seeds of 43,45,46,47 and 48 for the experiment in Section 5.2.