Improving Faithfulness by Augmenting Negative Summaries from Fake Documents

Anonymous ACL-IJCNLP submission

Abstract

Current abstractive summarization systems tend to hallucinate content that is unfaithful to the source document, posing a risk of misinformation. To mitigate hallucination, we must teach the model to distinguish hallucinated summaries from faithful ones. However, the commonly used maximum likelihood training does not disentangle factual errors from other model errors. To address this issue, we propose a back-translation-style approach to augment negative samples that mimic factual errors made by the model. Specifically, we train an *elaboration* model that generates hallucinated documents given the reference summaries, and then generate negative summaries from the fake documents. We incorporate the negative samples into training through a controlled generator, which produces faithful/unfaithful summaries conditioned on the control codes. Additionally, we find that adding textual entailment data through multitasking further boosts the performance. Experiments on XSum, Gigaword, and WikiHow show that our method consistently improves faithfulness without sacrificing informativeness according to both human evaluation and automatic metrics.[1]

1 Introduction

Despite the fast progress on fluency and coherence of text summarization systems, a common challenge is that the generated summaries are often unfaithful to the source document, containing hallucinated, non-factual content (Cao et al., 2018; Falke et al., 2019). Current summarization models are usually trained by maximum likelihood estimation (MLE), where unfaithful and faithful summaries are penalized equally if they both deviate from the reference. As a result, if the model fails to imitate the reference, it is likely to "over-generalize" and produce hallucinated content.

In this work, we address the issue by explicitly teaching the model to discriminate between positive (groundtruth) and negative (unfaithful) summaries. The key challenge is to generate realistic negative samples. Existing work on negative data augmentation mostly focuses on corrupting the reference (e.g., replacing entities) or sampling low-probability model outputs (Cao and Wang, 2021; Kryscinski et al., 2020a; Kang and Hashimoto, 2020). However, the synthetic data often does not resemble actual hallucinations from the model (Goyal and Durrett, 2021) and many methods rely on external tools such as NER taggers.

To generate unfaithful summaries, we propose a simple method inspired by back-translation (Sennrich et al., 2016) (Fig. 1). Specifically, we first generate fake documents using an *elaboration* model that is trained to produce a document given the summary. We then generate summaries from the fake documents, which are assumed to be unfaithful since they are likely to contain hallucinated information in the fake documents. Given the reference summaries and the augmented negative samples, we train a controlled generation model that generates either faithful or unfaithful summaries conditioned on a faithfulness control code. At inference time, we control the model to generate only faithful summaries. We call our approach CoFE (**Co**ntrolled **F**aithfulness via **E**laboration). The controlled generation framework also makes it easy to incorporate additional data: we show that jointly training on natural language inference (NLI) datasets to generate entailed (faithful) and non-entailed (unfaithful) hypothesis further improves the result.

We evaluate CoFE on three summarization datasets. Both automatic metrics and human evaluation show that our method consistently outperforms

[1]Code is available at https://github.com/COFE2022/CoFE.

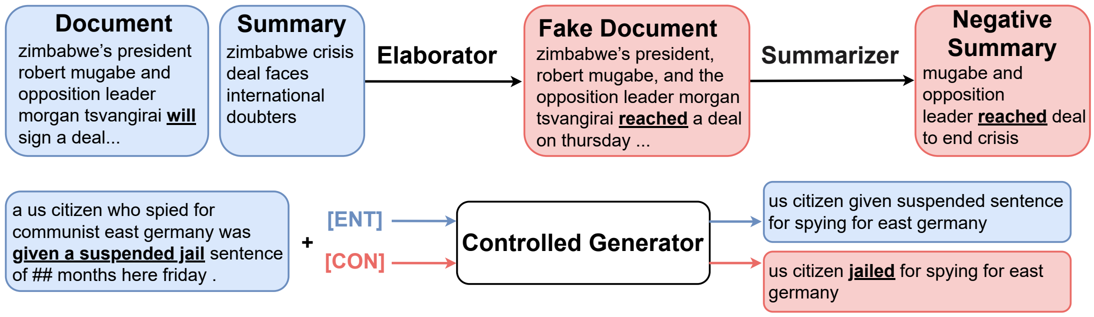


Figure 1: Overview of CoFE. Errors in the generated negative summaries are underlined.

prior methods in terms of faithfulness and content similarity to the reference, without sacrificing abstractiveness (Ladhak et al., 2021).

2 Approach

To learn a summarization model, the commonly used MLE aims to imitate the reference and does not distinguish different types of errors, thus the model may be misaligned with the desired behavior in downstream applications. For example, a faithful summary missing a detail would be preferred over a summary with hallucinated details, even if both have low likelihood under the data distribution. Therefore, additional inductive bias is needed to specify what unfaithful summaries are. Therefore, we augment negative examples and jointly model the distributions of both faithful and unfaithful summaries. At decoding time, we generate the most likely *faithful* summary.

Negative data augmentation. The key challenge in generating negative summaries is to simulate actual model errors. Prior approaches largely focus on named entities errors. However, different domains exhibit diverse hallucination errors (Goyal and Durrett, 2021); in addition, certain domains may not contain entities that can be easily detected by off-the-shelf taggers (e.g., stories or instructions). Our key insight is that the reverse summarization process—expanding a summary into a document—requires the model to hallucinate details, thus provides a domain-general way to produce unfaithful information. Instead of manipulating the reference summary directly, we expand it into a fake document, and generate negative summaries from it using the summarization model.

More formally, given a set of document-summary pairs (x, y), we train a backward elaboration model $p_{\text{back}}(x \mid y)$ as well as a forward summarization model $p_{\text{for}}(y \mid x)$. Then, given a reference summary y, we first generate a fake document $\hat{x}$ from p_{back}, then generate the negative sample y_{neg} from $\hat{x}$ using p_{for}, forming a pair of positive and negative samples (x, y) and (x, y_{neg}). To avoid data leakage (i.e. training models and generating summaries on the same data), we split the training data into K folds; the negative examples in each fold are generated by elaboration and summarization models trained on the rest $K - 1$ folds. We use $K = 5$ in the experiments.

Controlled generation. Given the positive and negative samples, we would like the model to learn to discriminate faithful summaries from unfaithful ones. Inspired by controlled generation methods (Keskar et al., 2019), we train the model to generate faithful or unfaithful summaries conditioned on a control code. In practice, we prepend a prefix at the beginning of the document (`[ENT]` for positive examples and `[CON]` for negative examples). At inference time, we always prepend `[ENT]` to generate faithful summaries.

Training. Our training data consists of positive examples (i.e. the original dataset) and generated negative samples, marked with different prefixes. Let $\mathcal{L}_{\text{pos}}$, $\mathcal{L}_{\text{neg}}$ denote negative log-likelihood (NLL) losses on the positive and negative examples. We use a multitasking loss that is a weighted sum of the two losses to balance the contribution from different types of examples: $\mathcal{L} = \mathcal{L}_{\text{pos}} + \lambda_1 \mathcal{L}_{\text{neg}}$.

Adding NLI datasets. We hypothesize that incorporating NLI data through multitasking would transfer knowledge about entailment to the generator, allowing it to better model faithful and unfaithful summaries. Specifically, an entailed hypothesis could be considered as the faithful summary, and non-entailed hypothesis as the unfaithful summary. Thus, the NLI sentence pairs can be naturally incorporated into our training framework. Let $\mathcal{L}_{\text{NLI}}$ denote the NLL loss on the auxiliary NLI examples. The loss function becomes: $\mathcal{L} = \mathcal{L}_{\text{pos}} + \lambda_1 \mathcal{L}_{\text{neg}} + \lambda_2 \mathcal{L}_{\text{NLI}}$.

3 Experiments

Datasets. We evaluate our approach on 3 datasets,[2] including: (i) **XSum** (Narayan-Chen et al., 2019), a dataset of BBC news articles paired with one-sentence summaries; (ii) **Gigaword** (Rush et al., 2015), a headline generation dataset compiled from the Gigaword corpus (Graff et al., 2003); and (iii) **Wikihow** (Koupaee and Wang, 2018), a dataset of how-to articles compiled from WikiHow,[3] each paired with paragraph headlines as the summary. For the auxiliary NLI data, we use **SNLI** (Bowman et al., 2015) and **MultiNLI** (Williams et al., 2018), both containing pairs of premise and hypothesis sentences.

Baselines. We compare with three baselines: (i) **MLE**, the standard training algorithm; (ii) **Loss Truncation (LT)** (Kang and Hashimoto, 2020) that adaptively removes high-loss examples which are assumed to be noisy/unfaithful; and (iii) **CLIFF** (Cao and Wang, 2021), a contrastive learning method based on generated negative samples.[4]

Implementation. All generation models (including the baselines) are fine-tuned from BART-large (Lewis et al., 2019). We decode from all models using beam search with a beam size of 6. For CoFE, we train the model using Fairseq (Ott et al., 2019) with a learning rate of 3e-5. We generate one negative sample for each document in the original dataset. To ensure that negative examples are different from the references, we remove the top 10% summaries ranked by their edit distances to the reference. To train the controlled generator, we set coefficients (λ_1, λ_2) of the loss terms such that the reweighted number of examples in the original dataset, the negative samples, and optionally the NLI datasets have the ratio $1 : 0.5 : 0.5$. Details for other baselines are in Appendix B.

Metrics. A good summary must cover important content, be faithful to the document, and be succinct. We evaluate the generated summaries from the following aspects:

- Content selection. We use similarity to the reference as a proxy measure, and report ROUGE (Lin, 2004) and BertScore (Zhang et al., 2020).
- Faithfulness. For automatic evaluation, we use QuestEval (Scialom et al., 2021), a QA-based metric; and FactCC (Kryscinski et al., 2020b), a learned faithfulness predictor. For human evaluation, we randomly selected 100 examples from each dataset. Given a document with the generated summaries from all systems (including the references), we ask annotators from Amazon Mechanical Turk to evaluate whether each summary is supported by the document. Each output is evaluated by 3 annotators. If two or more annotators vote "supported", then we consider the output faithful. More details are described in Appendix B.
- Extractiveness. Ladhak et al. (2021) show that it is important to measure the extractiveness of the summaries to determine whether a method improves faithfulness mainly by copying from the document. Therefore, we also report *coverage* and *density* that measure the percentage of the words and the average length of text spans copied from the document (Grusky et al., 2020).

Results. Table 1 shows our main results. CoFE outperforms the baselines in human evaluated faithfulness accuracy on 2 out of the 3 datasets. On Gigaword, LT performs the best but it also incurs the largest drop in ROUGE and BertScore and increase in copying. CLIFF is good at fixing entity errors, but it has less advantage on datasets like WikiHow that contain fewer entities detectable by off-the-shelf taggers. On average, CoFE is less extractive than CLIFF and LT, indicating that our faithfulness improvements are not simply due to more copying (Ladhak et al., 2021). Finally, we find that adding NLI brings a marginal improvement on top of our negative samples.

Are generated negative summaries really unfaithful? Our method relies on the assumption that elaboration of summaries introduces hallucinations, which results in unfaithful summaries. To verify this, we assess whether our generated negative samples are true negatives. We randomly sample 1000 documents for each dataset and compare the negative samples generated by our method vs. CLIFF. We report the QuestEval scores as well as human-annotated faithfulness scores on a sub-

[2] We did not include the CNN/DailyMail dataset (See et al., 2017) since the reference summaries in this dataset tend to be more extractive, and models exhibit much fewer faithfulness errors (Durmus et al., 2020).

[3] https://www.wikihow.com/Main-Page.

[4] For CLIFF, we use SysLowCon which is reported to be the best amongst their methods for negative sample generation.

Dataset	Method	Ref. Similarity (↑)		Faithfulness (↑)			Extractiveness (↓)	
		RL	BS	Human Acc / # Votes	QuestEval	FactCC	Coverage	Density
XSUM	MLE	**37.21**	45.36	64% / 192	29.15	22.93	0.7596	1.6986
	LT	35.77	**47.39**	61% / 188	29.22	20.29	0.7564	1.7473
	CLIFF	36.41	52.78	68% / 192	29.22	**24.74**	0.7670	1.6904
	CoFE	36.38	52.09	68% / 194	29.53	23.56	0.7534	1.6460
	CoFE +NLI	36.98	52.90	**70%** / **196**	**29.76**	23.61	**0.7528**	**1.5961**
Gigaword	MLE	33.95	27.77	70% / 206	39.16	50.90	**0.7302**	**1.9415**
	LT	34.22	26.35	**76%** / 204	**41.97**	**56.27**	0.8026	2.7106
	CLIFF	**35.59**	**30.78**	73% / 201	39.08	50.89	0.7406	2.1100
	CoFE	35.53	30.70	73% / **210**	39.26	52.14	0.7315	2.0937
	CoFE +NLI	34.02	27.77	74% / 211	39.58	52.74	0.7390	2.1518
WikiHow	MLE	37.93	43.55	87% / 233	26.87	96.11	0.8091	1.8473
	LT	38.01	43.61	83% / 228	26.88	96.01	0.8302	2.0126
	CLIFF	37.29	42.73	83% / 233	27.46	**96.36**	0.8092	1.8058
	CoFE	37.86	**43.67**	84% / 232	28.02	95.66	**0.7962**	1.8362
	CoFE +NLI	**38.23**	43.08	**88%** / **238**	**28.21**	96.25	0.7963	**1.8261**

Table 1: Main results. The best result per metric for each datasets is **bolded**. For "Extractiveness", lower is better. RL and BS denotes ROUGE-L and BertScore-P. For human evaluation, we report the percentage of faithful summaries based on majority vote (Human Acc) and the total number of votes for faithfulness (# Votes). CoFE outperforms the baselines on average without decreasing overlap with the reference or increasing copying.

set of 100 documents (following the same procedure described in Metrics). The results are shown in Table 2. As a sanity check, the faithfulness scores of negative samples are much lower than those in Table 1, suggesting a qualitative difference between the negative and positive samples. Compared to CLIFF, our method gets lower QuestEval and human-annotated faithfulness scores across all datasets, showing that our negative samples are more likely to be unfaithful.

Dataset	Method	QuestEval (↓)	Human Acc (↓)
XSUM	CoFE	24.34	19%
	CLIFF	27.65	60%
Gigaword	CoFE	33.69	34%
	CLIFF	39.42	40%
WikiHow	CoFE	24.72	32%
	CLIFF	28.31	39%

Table 2: Quality of generated negative samples. Lower number is better (more likely to be true negatives).

Is faithfulness controllable? We use the controlled generator to model distributions of both faithful and unfaithful summaries. To verify the effect of the control code, we measure the change in ROUGE scores on XSum after toggling the control code from faithful (`[ENT]`) to unfaithful (`[CON]`). As expected, we observe that R1/R2 drops from 45.26/22.19 to 37.29/15.82, indicating that the model has learned to discriminate faithful and unfaithful summaries.

4 Related Work

Recent work in abstractive summarization has shown that state-of-the-art models sometimes generate non-factual information that is not consistent with the article (Falke et al., 2019; Cao et al., 2018). This has spurred efforts in building automated metrics for factuality (Kryscinski et al., 2020a; Durmus et al., 2020; Wang et al., 2020; Goyal and Durrett, 2020) and more faithful systems (Xu et al., 2020; Filippova, 2020).

Prior work has proposed to filter the training dataset to remove noisy examples to improve faithfulness. For example, Kang and Hashimoto (2020) drop high-loss examples from training observing that these examples are usually of lower quality. Nan et al. (2021) discard sentences from gold summaries if there is an entity that does not match the entities in the document. Goyal and Durrett (2021) take a more fine-grained approach, and use a dependency arc-based entailment metric (Goyal and Durrett, 2020) to filter noisy tokens from the summary.

On modeling, prior work has incorporated additional information such as relation triplets (Cao et al., 2018), knowledge graph of relations (Zhu et al., 2021) and topical information (Aralikatte et al., 2021) from the document. Another line of work aims to fix some of these faithfulness errors as a post-processing step by revising the generated outputs (Dong et al., 2020; Chen et al., 2021; Zhao et al., 2020; Cao et al., 2020).

References

Rahul Aralikatte, Shashi Narayan, Joshua Maynez, Sascha Rothe, and Ryan McDonald. 2021. Focus attention: Promoting faithfulness and diversity in summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6078–6095, Online. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. 2020. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, Online. Association for Computational Linguistics.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Sihao Chen, Fan Zhang, Kazoo Sone, and Dan Roth. 2021. Improving faithfulness in abstractive summarization with contrast candidate generation and selection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5935–5941, Online. Association for Computational Linguistics.

Xin Luna Dong, Hannaneh Hajishirzi, Colin Lockard, and Prashant Shiralkar. 2020. Multi-modal information extraction from text, semi-structured, and tabular data on the web. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 23–26, Online. Association for Computational Linguistics.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Katja Filippova. 2020. Controlled hallucinations: Learning to generate faithfully from noisy data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 864–870, Online. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Evaluating factuality in generation with dependency-level entailment. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3592–3603, Online. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2021. Annotating and modeling fine-grained factuality in summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1449–1462, Online. Association for Computational Linguistics.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

Max Grusky, Mor Naaman, and Yoav Artzi. 2020. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies.

Daniel Kang and Tatsunori B. Hashimoto. 2020. Improved natural language generation via loss truncation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 718–731, Online. Association for Computational Linguistics.

N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020a. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020b. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

F. Ladhak, E. Durmus, H. He, C. Cardie, and K. McKeown. 2021. Faithful or extractive? on mitigating the faithfulness-abstractiveness trade-off in abstractive summarization. *arXiv preprint arXiv:2108.13684*.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization asks for fact-based evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Jiacheng Xu, Shrey Desai, and Greg Durrett. 2020. Understanding neural abstractive summarization models via uncertainty. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6275–6281, Online. Association for Computational Linguistics.

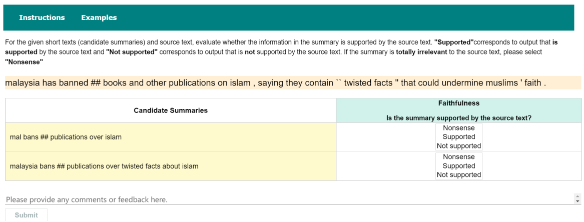
Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert.

Zheng Zhao, Shay B. Cohen, and Bonnie Webber. 2020. Reducing quantity hallucinations in abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2237–2249, Online. Association for Computational Linguistics.

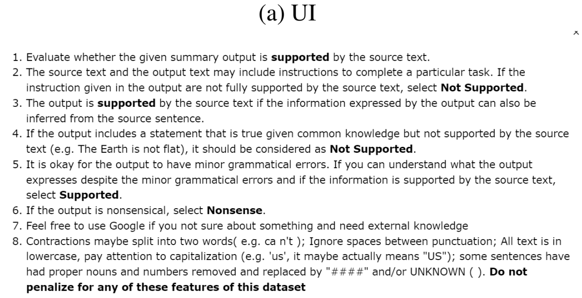
Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2021. Enhancing factual consistency of abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 718–733, Online. Association for Computational Linguistics.

A Human-Evaluation Setup

We use Amazon Mechanical Turk as human-evaluations platform. The prompt is shown in Fig. 2. We only hire annotators in U.S. and with more than 98% hit receive rate.

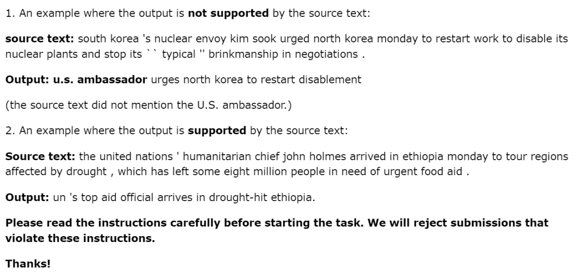


(a) UI



1. Evaluate whether the given summary output is **supported** by the source text.
2. The source text and the output text may include instructions to complete a particular task. If the instruction given in the output are not fully supported by the source text, select **Not Supported**.
3. The output is **supported** by the source text if the information expressed by the output can also be inferred from the source sentence.
4. If the output includes a statement that is true given common knowledge but not supported by the source text (e.g. The Earth is not flat), it should be considered as **Not Supported**.
5. It is okay for the output to have minor grammatical errors. If you can understand what the output expresses despite the minor grammatical errors and if the information is supported by the source text, select **Supported**.
6. If the output is nonsensical, select **Nonsense**.
7. Feel free to use Google if you not sure about something and need external knowledge
8. Contractions maybe split into two words(e.g. ca n't); Ignore spaces between punctuation; All text is in lowercase, pay attention to capitalization (e.g. 'us', it maybe actually means "US"); some sentences have had proper nouns and numbers removed and replaced by "####" and/or UNKNOWN ( ). **Do not penalize for any of these features of this dataset**

(b) Instructions



1. An example where the output is **not supported** by the source text:

source text: south korea 's nuclear envoy kim sook urged north korea monday to restart work to disable its nuclear plants and stop its `` typical '' brinkmanship in negotiations .

Output: u.s. ambassador urges north korea to restart disablement

(the source text did not mention the U.S. ambassador.)

2. An example where the output is **supported** by the source text:

Source text: the united nations ' humanitarian chief john holmes arrived in ethiopia monday to tour regions affected by drought , which has left some eight million people in need of urgent food aid .

Output: un 's top aid official arrives in drought-hit ethiopia.

Please read the instructions carefully before starting the task. We will reject submissions that violate these instructions.

Thanks!

(c) Example

Figure 2: Amazon MTURK setup

B Experiment Detail

Model details. For both the summarization model, the elaboration model, and the controlled generator, we fine-tune a pre-trained BART model (Lewis et al., 2019) using Fairseq (Ott et al., 2019) and the default learning rate $3e-5$. All summaries are generated using beam search with a beam size of 6. Linear-scale the max update steps of learning-rate scheduler according to the number of samples in the training data.

For hyperparameters, we follow the setting of fine-tuning BART on XSUM (Lewis et al., 2019), which uses 8 cards, UPDATE_FREQ is 4, TOTAL_NUM_UPDATES is 20000. Linear scale the max-update-step by extra number of negative data and NLI data. For the weights of different tasks, an intuitive idea is to fix "the ratio of the product of the number of samples and their weights for different tasks". We set Product$_{summarzation}$: Product$_{negative}$: Product$_{NLI}$ $= 1 : 0.5 : 0.5$. For example, if we have 1000 positive and 1000 negative samples in training set, the weight of positive data is 1, the weight of negative data is 0.5. If we filter half negative samples out, reduce it into 500 samples, then the weight of two tasks is 1.

Other baselines: For MLE, the repository of BART releases hyperparameters and checkpoint for XSUM. Based on the hyperparameters for xsum, we scale the max-update-step linearly according to the size of training set of gigaword and wikihow. For Loss-truncation, besides the hyperparameters in MLE, there are some hyperparameters for the loss function. We follow the settings in their paper. For CLIFF, we only use "SysLowCon" as the negative data augmentation method, which is the best single method they claimed in the paper. They release the checkpoints of XSUM and hyperparameters in their github repository. We only re-scale the max-update-step.

Computational Resources and Model Size. CoFE on one dataset requires training 11 models, including 10 models for generating negative samples, since each fold needs an elaborator and a summarizer. On a 4 RTX8000 GPU node, each model needs 2 hours to fine-tune. It takes 22 hours to get the final generated output. BART-large has 400M parameters.