Source Identification in Abstractive Summarization

Yoshi Suhara*
NVIDIA
ysuhara@nvidia.com

Dimitris Alikaniotis
Grammarly
dimitris.alikaniotis@grammarly.com

Abstract

Neural abstractive summarization models make summaries in an end-to-end manner, and little is known about how the source information is actually converted into summaries. In this paper, we define input sentences that contain essential information in the generated summary as *source sentences* and study how abstractive summaries are made by analyzing the source sentences. To this end, we annotate source sentences for reference summaries and system summaries generated by PEGASUS on document-summary pairs sampled from the CNN/DailyMail and XSum datasets. We also formulate automatic source sentence detection and compare multiple methods to establish a strong baseline for the task. Experimental results show that the perplexity-based method performs well in highly abstractive settings, while similarity-based methods perform robustly in relatively extractive settings.[1]

1 Introduction

Text summarization research has enjoyed recent advances in neural networks and pre-trained language models, which make abstractive summarization the most common approach (Liu and Lapata, 2019; Rothe et al., 2020; Zhang et al., 2020a). While continuing efforts in improving factuality and faithfulness (Kryscinski et al., 2020; Nan et al., 2021) have been made, abstractive summarization models, when trained properly, can create concise and coherent summaries from source documents.

Different from extractive summaries, for which we know the source information, it is not clear how an abstractive summary gathers various pieces of information that spread over different sentences in the input document (or input documents for multi-document summarization). Identifying source information is essential for the explainability and interpretability of summaries.

Therefore, in this paper, we aim to disentangle the abstractive summarization mechanism by identifying sentences that contain essential source information described in the generated summary. Existing studies use lexical similarity (e.g., ROUGE) and semantic similarity (e.g., BERTScore) for detecting sentences in the input document (Vig et al., 2021; Syed et al., 2021) to help understand what the key source information for a generated summary. Another line of work analyzes cross-attention weights for abstractive summarization (Baan et al., 2019) and data-to-text generation (Juraska and Walker, 2021). However, the approach mostly focuses on lexical and semantic similarity between the generated summary and input sentences without considering which input sentences provide source information.

To this end, we define input sentences that contain essential information for the generated summary as *source sentences* and aim to understand how abstractive summaries are composed by analyzing source sentences. We annotate source sentences for both reference summaries and system summaries generated by PEGASUS (Zhang et al., 2020b) on the XSum and CNN/Daily Mail (CNN/DM) datasets, which are among the most popular summarization benchmarks in English. We also formulate the automatic source sentence detection task to verify the effectiveness of existing methods (i.e., attention-based and similarity-based) for detecting source sentences. We develop a simple-yet-effective method based on perplexity gain—the difference in perplexity between the original text and the text after a specific sentence has been removed. We show that it significantly outperforms the existing methods in abstractive settings.

The contributions of the paper are as follows:

- We propose the novel task of automatic source sentence detection and create SourceSum,

*Work done while at Grammarly.

[1]Our code and data are available at https://github.com/suhara/sourcesum.

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics Volume 2: Short Papers, pages 212–224
March 17-22, 2024

which annotates source sentences of reference summaries and system summaries generated by PEGASUS on document-summary pairs sampled from XSum and CNNDM.
- We develop a simple-yet-effective perplexity gain method to detect source sentences and report that in a more abstractive setting, the perplexity gain method performs well while similarity-based methods can be a solid solution to extractive settings.

2 SourceSum

In this paper, we used XSum[2] (Narayan et al., 2018) and CNN/DM[3] (See et al., 2017) as the source datasets, as (1) they are the most common summarization benchmarks and (2) they have different levels of abstractiveness (Narayan et al., 2018), to make the benchmark comprehensive and robust.

2.1 Corpus creation

For each dataset, we randomly sampled document-summary pairs. We used a commonly used summarization model PEGASUS (Zhang et al., 2020b) fine-tuned on either of the datasets.

In addition to generated summaries, we collect annotations for document-reference-summary pairs for the same set of examples, as abstractive summarization models may cause hallucinations, which would affect the quality of the benchmark. This setting also enables us to conduct a comparative analysis of reference and generated summaries.

Souce sentence annotation For each document-summary pair, the annotator is asked to judge if each sentence contributes to the summary after reading the summary and document (Q1 in Figure 1). The judgment criteria are whether the sentence (1) **contributes to summary:** This sentence would be valuable in writing the summary, or (2) **does not contribute to summary:** The summary could be written without this sentence.

Reconstructability annotation After completing the source sentence annotation step, the annotator was asked to answer a question "Could you write this summary based solely on the sentences that you identified as important?" to flag hallucinated summaries and ensure that SourceSum consists of self-contained document-summary pairs.

[2] https://huggingface.co/datasets/xsum
[3] https://huggingface.co/datasets/cnn_dailymail version: 3.0.0

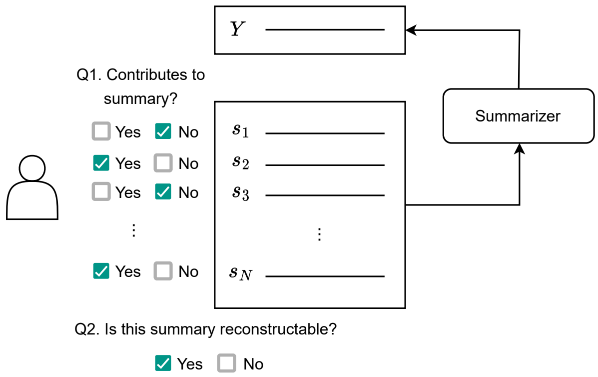


Figure 1: Annotation flow for SourceSum. For each document-summary pair, the human annotator is asked to annotate each sentence (Q1), followed by the reconstructability question (Q2).

This step is important for document-reference-summary pairs as well. As the reference summaries were taken from the introductory sentence (XSum) and the summary bullets (CNN/DM) of each article, it is not ensured that the reference summaries can be created solely from the original article, as reported in Wang et al. (2020).

2.2 Dataset Statistics

We hired expert annotators to annotate source sentences on 2,000 document-summary pairs from XSum and CNN/DM. The inter-annotator agreement ratios (Krippendorff's alpha) for the reconstructability annotation and source sentence annotation are 0.8 and 0.8, respectively. As shown in Table 4 and somewhat surprisingly, more than half of XSum summaries are not reconstructable, while most CNN/DM summaries are. After removing document-summary pairs that were judged non-reconstructable, SourceSum consists of 1,211 document-summary pairs.

The basic statistics of SourceSum are shown in Table 1. Note that the summary is split into sentences for statistics calculation for the CNN/DM[4]. The novel n-gram statistics show that PEGASUS generates quite extractive summaries (e.g., 2.9% of unique unigram in generated summaries) for CNN/DM while generated summaries are still more abstractive for XSum. This indicates that the behaviors of the two PEGASUS models fine-tuned on XSum and CNN/DM are different with respect to the abstractiveness of the generated summary.

[4] XSum only contains single-sentence summaries.

SourceSum	# pairs	# sent	# src sent	Input len	Summ len	% of novel n-grams in summary unigram	bigram	trigram	4-gram
$\text{XSum}_{\text{PEGASUS}}$	119	10.28	3.09 (30.1%)	275.09	19.51	24.26	73.54	88.90	94.30
$\text{XSum}_{\text{Reference}}$	119	10.28	3.40 (33.1%)	275.09	23.71	33.93	82.54	94.10	97.58
$\text{CNN/DM}_{\text{PEGASUS}}$	468	11.58	1.72 (14.9%)	309.07	16.95	2.90	19.26	29.96	37.20
$\text{CNN/DM}_{\text{Reference}}$	505	11.56	2.03 (17.6%)	305.79	15.87	13.53	50.45	67.92	77.02

Table 1: Statistics of SourceSum. Input len and Summ len are token counts using the PEGASUS tokenizer.

3 Source Sentence Detection

Problem Formulation Given an input document X, which consists of N sentences $(s_1, \ldots, s_N)$, and a system summary Y generated by a summarization model θ, the task is to identify a proper subset of input sentences D' that are essential to creating Y. The task can be cast as a sentence-scoring problem, where the score of each input sentence $R(s)$, assuming the threshold value d to be a hyperparameter (i.e., $D' = \{s \in D | R(s) > d\}$).

3.1 Similarity-based Method

A simple approach is to choose sentences based on the similarity between the summary and input sentences. The idea has been implemented in Vig et al. (2021); Syed et al. (2021), which use ROUGE and BERTScore for the similarity calculation. ROUGE puts more emphasis on lexical similarity while BERTScore takes semantic similarity into account.

$$R(s, Y) = \mathrm{sim}(s, Y) \quad (1)$$

Note that the similarity-based method is input- and model-agnostic, and it does not use X and θ for relevance score calculation. We also tested more sophisticated methods SimCSE (Gao et al., 2021) and a PMI-based extractive summarization method (Padmakumar and He, 2021), in addition to GPT-3.5 (`text-davinci-003`) (Ouyang et al., 2022). The prompt used for GPT-3.5 can be found in Appendix (Table 5).

We also used LexRank (Erkan and Radev, 2004) as another baseline, as it can be used as a sentence-scoring method based on the centrality of the input sentence graph (i.e., summary-agnostic).

3.2 Cross-attention Weights

As the decoder takes input information via cross-attention, one approach is to calculate the importance of each sentence using cross-attention weights (Juraska and Walker, 2021):

$$R(s, Y|X;\theta) = \frac{1}{|s||Y|}\sum_{x\in s}\sum_{y\in Y} w(x, y; \theta), \quad (2)$$

where $w(x, y; \theta)$ denotes the cross-attention weight of the attention vector for the token x in the encoder against the token y in the decoder. As the decoder typically has multiple attention heads on multiple Transformer layers, we calculate the average over the multiple heads and layers.

3.3 Perplexity Gain

Different from the similarity-based method, the attention-based method is *model-specific*, but is still an indirect method. Therefore, we consider a more direct way to calculate the importance of each sentence based on *perplexity gain* after removing the sentence:

$$R(s, Y|X;\theta) = \mathrm{PPL}(Y|X_{\backslash s};\theta) - \mathrm{PPL}(Y|X;\theta), \quad (3)$$

where $\mathrm{PPL}(Y|X;\theta)$ denotes the perplexity of the summary Y generated by the model θ given the input document X. The intuition behind this method is that the model should be *more perplexed* (i.e., less confident) to generate the same summary if more relevant sentence is removed.

4 Evaluation

Evaluation metrics To make the evaluation independent of the choice of threshold selection, we used ranking metrics for evaluation, namely NDCG and MAP (Manning et al., 2008). For NDCG, we used the total votes as the score to consider sentences with more votes more important. For MAP calculation, we binarized annotations and considered source sentences if two annotators agree it is relevant.

Results As shown in Table 2, Perplexity Gain outperforms the other methods for the XSum dataset, whereas the similarity-based methods perform best on the CNN/DM-Pegasus (SimCSE, BERTScore) and CNN/DM-Reference (ROUGE). The results confirm our hypothesis on the abstractiveness of summaries that it is necessary to access the summarization model for source identification.

	XSum$_{\text{PEGASUS}}$ NDCG	XSum$_{\text{PEGASUS}}$ MAP	XSum$_{\text{Ref}}$ NDCG	XSum$_{\text{Ref}}$ MAP	CNN/DM$_{\text{PEGASUS}}$ NDCG	CNN/DM$_{\text{PEGASUS}}$ MAP	CNN/DM$_{\text{Ref}}$ NDCG	CNN/DM$_{\text{Ref}}$ MAP
LexRank (Erkan and Radev, 2004)	.7499	.5302	.7687	.5435	.6596	.4226	.6841	.4540
BERTScore (Syed et al., 2021)	.8499	.6878	.8762	.7312	.9134	**.8536**	.8851	.7926
ROUGE (Vig et al., 2021)	.8475	.6740	.8523	.6756	.9110	.8484	**.8984**	**.8087**
SimCSE (Gao et al., 2021)	.8579	.7016	.8661	.7093	**.9141**	.8469	.9048	.8169
PMI (Padmakumar and He, 2021)	.8193	.6316	.8329	.6480	.8069	.6919	.7353	.5592
GPT-3.5 (Ouyang et al., 2022)	.8233	.5405	.8422	.5764	.8095	.5039	.8252	.5561
Cross-attention (Juraska and Walker, 2021)	.7048	.4757	—	—	.6312	.3544	—	—
Perplexity Gain	**.8976**	**.7753**	**.8983**	**.7710**	.8798	.8138	.8570	.7465

Table 2: Performance of the source sentence detection methods on SourceSum.

Model	Input	XSum R1	XSum R2	XSum RL	CNN/DM R1	CNN/DM R2	CNN/DM RL
PEGASUS	All sentences	53.40	30.49	45.38	47.13	25.75	35.54
	Source sentences only	48.36↓	25.62↓	40.44↓	47.55↑	25.68↓	36.16↑
BART	All sentences	50.32	26.35	40.83	45.56	23.32	32.70
	Source sentences only	47.29↓	22.73↓	38.82↓	47.53↑	24.92↑	34.58↑
LexRank	All sentences	19.84	3.08	14.46	37.30	15.94	23.45
	Source sentences only	23.36↑	5.74↑	17.49↑	45.45↑	23.47↑	27.04↑

Table 3: Summarization performance of PEGASUS, BART, and LexRank on SourceSum (XSum and CNN/DM). Using only source sentences as input improves LexRank's performance on both datasets, while significant degradation is observed for PEGASUS and BART on XSum.

5 Analysis

Are summaries reconstructable? As reference summaries for the XSum (CNN/DM) dataset were scraped from the introductory sentence (the summary bullets), it is not ensured that reference summaries can be created only from the input documents. The same thing can be said for summaries generated by abstractive summarization models, which may hallucinate content. To analyze this, we annotated document-summary pairs with respect to the reconstructability (§2.1).

Table 4 shows that more than half of XSum summaries are not reconstructable, while most of CNN/DM summaries are. Compared to the reference summaries, summaries generated by the Pegasus models are slight more reconstructable, as expected. The higher reconstructability of CNN/DM is also supported by the lower abstractiveness (i.e., lower novel n-grams).

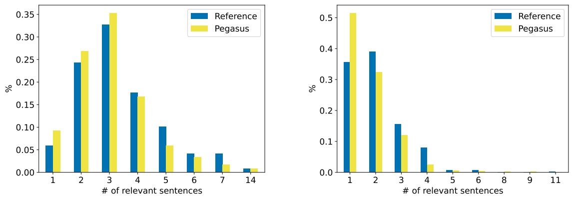
How many source sentences are used per summary? Figure 2 shows the distribution of the number of source sentences per one summary sentence. As shown in the figure, XSum summaries have more source sentences (3.40 on average) than CNN/DM summaries (1.72 on average). The trend is aligned with the abstractiveness/extractiveness of those datasets. Regarding the differences in reference and generated summaries, PEGASUS amplifies the characteristics of each dataset—Generated summaries tend to have more (less) source sentences on XSum (CNN/DM).

Reconstructable?	XSum Ref.	XSum PEGASUS	CNN/DM Ref.	CNN/DM PEGASUS
Yes	30.3%	37.3%	87.7%	95.0%
Partly	18.1%	15.4%	4.1%	3.0%
No	51.7%	47.3%	8.2%	2.0%

Table 4: reconstructability of reference/generated summaries. More than half of XSum reference summaries cannot be created only from the input document.

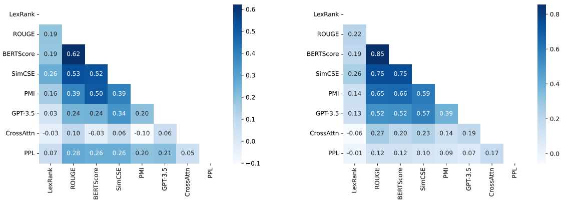
Are non-source sentences unnecessary? We have defined source sentences from which summaries can be made. A natural question is whether the other "non-source" sentences are necessary for generating the same abstractive summaries. To answer the question, we evaluated the quality of summaries by PEGASUS, BART (Lewis et al., 2020), and LexRank under two settings: (1) All sentences and (2) source sentence only.

Results are shown in Table 3. Interestingly and somewhat surprisingly, by removing non-source sentences, PEGASUS and BART show significant



(a) XSum (b) CNN/DM

Figure 2: Distribution of the number of (ground-truth) source sentences. Generated summaries tend to have more source sentences on XSum while having fewer source sentences on CNN/DM.



(a) XSum (b) CNN/DM

Figure 3: Correlation analysis of the source sentence detection methods.

degradations on XSum while slight improvements are observed on CNN/DM. In fact, we confirm some degree of hallucinations when generating with source sentence only, as shown in Table 7. We consider that especially in an abstractive setting, non-source sentences still provide context information, which helps give *confidence* to the summarization model. From the results, we confirm that abstractive summarization by the pre-trained Transformer model is more complicated than simply selecting and rewriting source information. The quality improvements for LexRank are reasonable as LexRank should be a higher chance to select relevant sentences in the source sentence-only setting.

Do different methods detect different source sentences? Table 2 does not show if different methods detect the same or different source sentences. To analyze this, we calculated correlation coefficients of scores calculated by the different methods. Figure 3 shows that the scores of the similarity-based methods are highly correlated while Perplexity Gain and Cross Attention detect source sentences differently.

6 Related Work

It is hard to interpret how commonly used Transformer-based summarization models generate abstractive summaries. Xu and Durrett (2021) developed an ablation-attribution framework that identifies the generation model by comparing behaviors of a language model and a summarization model. Baan et al. (2019) investigated the interpretability of multi-head attention in abstractive summarization and found that attention heads can be pruned without a significant performance drop.

Another line of work analyzes how multiple sentences are fused into summary sentences (Lebanoff et al., 2019a,b, 2020a,b). Lebanoff et al. (2020b) created a dataset that contains fine-grained point-of-correspondence between a summary and two source sentences. Our work covers beyond the scope of their work as SourceSum assigns source sentence labels to all source sentences on both generated and human summaries.

One simple-yet-effective approach for explainability is to highlight sentences similar to the generated summary. Vig et al. (2021) and Syed et al. (2021) use ROUGE and BERTScore to capture the lexical and semantic similarity to help the user understand the source information of the generated summary. Juraska and Walker (2021) use cross-attention to understand the behavior of the data-to-text model. Wang et al. (2021) develops a hybrid summarization model that takes into account sentence similarity to improve explainability and faithfulness. Saha et al. (2023) develops a framework that uses neural modules to construct a tree representation to understand the relationship between a human-written summary and the input document. This paper is aligned with the line of work but rather focuses on formulating the source sentence detection task and creating a benchmark, so we can evaluate and compare different methods quantitatively and qualitatively.

7 Conclusion

In this paper, we formulate the source sentence detection task, which finds input sentences that are essential to generating the given abstract summary, to study how abstractive summaries are made. We annotated source sentences for reference summaries and system summaries generated by PEGASUS on XSum and CNN/DM and created a benchmark SourceSum. Experimental results on SourceSum show that Perplexity Gain, which is based on the perplexity increase when the target sentence is removed, performs the best in highly abstractive settings (XSum), while similarity-based methods perform robustly in extractive settings (CNN/DM).

Limitations

As we shed light on a new perspective on abstractive summarization, the paper has certain limitations. First, our benchmark SourceSum is made for single-document summarization in a single domain (news) in a single language (English), which not necessarily ensuring the generalizability for other domains and languages. For multi-document summarization, we believe that the same annotation and evaluation framework can be applied and is interesting future work. Second, the annotation is sentence-level in SourceSum. There may be a chance that annotated source sentences also contain information unnecessary to generate the summary. We carefully discussed the annotation guideline and decided to use sentence-level annotation to ensure the annotation quality. Last but not least, the benchmark is created on top of a Transformer-based encoder-decoder model PEGASUS and the results do not necessarily apply to other encoder-decoder models or autoregressive models such as GPT series. With those limitations, we still believe that the paper and the benchmark are beneficial for the community in providing insights into abstractive summarization models.

Acknowledgements

We thank Jennifer Bellik for her support on the annotation collection and thank the anonymous reviewers for their helpful comments.

References

Joris Baan, Maartje ter Hoeve, Marlies van der Wees, Anne Schuth, and Maarten de Rijke. 2019. Understanding multi-head attention in abstractive summarization.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Juraj Juraska and Marilyn Walker. 2021. Attention is indeed all you need: Semantically attention-guided decoding for data-to-text NLG. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 416–431, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. 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.

Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020a. Learning to fuse sentences with transformers for summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4136–4142, Online. Association for Computational Linguistics.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019a. Analyzing sentence fusion in abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 104–110, Hong Kong, China. Association for Computational Linguistics.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020b. Understanding points of correspondence between sentences for abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 191–198, Online. Association for Computational Linguistics.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019b. Scoring sentence singletons and pairs for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of* *the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.

Feng Nan, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. 2021. Improving factual consistency of abstractive summarization via question answering. 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 6881–6894, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Vishakh Padmakumar and He He. 2021. Unsupervised extractive summarization using pointwise mutual information. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2505–2512, Online. Association for Computational Linguistics.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Swarnadeep Saha, Shiyue Zhang, Peter Hase, and Mohit Bansal. 2023. Summarization programs: Interpretable abstractive summarization with neural modular trees. In *ICLR*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Shahbaz Syed, Tariq Yousef, Khalid Al Khatib, Stefan Jänicke, and Martin Potthast. 2021. Summary explorer: Visualizing the state of the art in text summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 185–194, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jesse Vig, Wojciech Kryscinski, Karan Goel, and Nazneen Rajani. 2021. SummVis: Interactive visual analysis of models, data, and evaluation for text summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 150–158, Online. 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.

Haonan Wang, Yang Gao, Yu Bai, Mirella Lapata, and Heyan Huang. 2021. Exploring explainable selection to control abstractive summarization. In *AAAI*.

Jiacheng Xu and Greg Durrett. 2021. Dissecting generation modes for abstractive summarization models via ablation and attribution. 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 6925–6940, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2021a. Leveraging lead bias for zero-shot abstractive news summarization. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1462–1471, New York, NY, USA. Association for Computing Machinery.

Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2021b. Leveraging lead bias for zero-shot abstractive news summarization.

In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1462–1471, New York, NY, USA. Association for Computing Machinery.

A Annotation collection

A.1 Data preparation

Following the official script used to fine-tune the summarization models, we filtered out examples whose number of tokens in the input document is greater or less than certain numbers.

XSum We sampled document-summary pairs from the XSum dataset[5]. We filtered examples whose number of tokens is greater than 56 and less than 512.

CNN/DM We sampled document-summary pairs from the CNN/DailyMail dataset[6]. We filtered examples whose number of tokens is greater than 142 less than 1024.

A.2 Summary generation

XSum `pegasus-xsum`[7] with the default generation configuration (length_penalty = 0.6, max_length = 64, num_beams = 8).

CNN/DM `pegasus-cnn_dailymail`[8] with the default generation configuration (length_penalty = 0.8, min_length = 32, max_length = 128, num_beams = 8). Summaries are split by `<n>` into sentences.

A.3 Pilot Study

We conducted two pilot studies to revise the annotation guideline while helping the annotators familiar with the annotation task. We initially used ternary labels (Essential, Related, Unrelated) for annotation. However, the inter-annotator agreement was not sufficiently high (Krippendorff's alpha was 0.443). Thus, we decided to use binary labels and further clarify the label definitions. Also, we decided to exclude input documents that consist of more than 15 sentences, based on the feedback from the annotators, to reduce the cognitive load and to ensure the annotation quality.

[5] https://huggingface.co/datasets/xsum

[6] https://huggingface.co/datasets/cnn_dailymail 3.0.0

[7] https://huggingface.co/google/pegasus-xsum

[8] https://huggingface.co/google/pegasus-cnn_dailymail

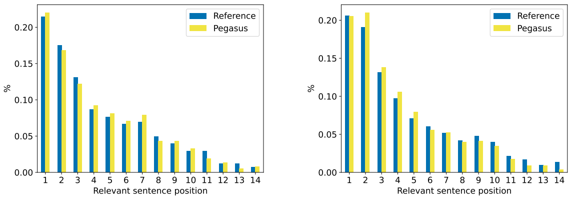
This task is to identify the sentences in a document that contribute to a given summary of that document. This annotation is a sentence-labeling task. For each snippet, you'll see a summary (labeled Summary:) and a sentence of a short news article (labeled Sentence:).

The output will be a score from 0 to 100, 0 with "doesn't contribute to summary" with the highest confidence and 100 with "contribute to summary" with the highest confidence.

Summary: `{summary}`
Sentence: `{sentence}`

Score:

Table 5: Prompt for GPT-3.5 used in the experiment.



(a) XSum (b) CNN/DM

Figure 4: Distribution of source sentence absolute positions. Both plots support that a commonly used lead-3 .

A.4 Annotation guideline

Figure 1 depicts the annotation workflow. For each document-summary pair, the human annotator submits source sentence labels followed by a reconstructability label. The full annotation guideline and reconstructability judgment guideline are shown in Tables 8 and 9.

B Source Sentence Detection

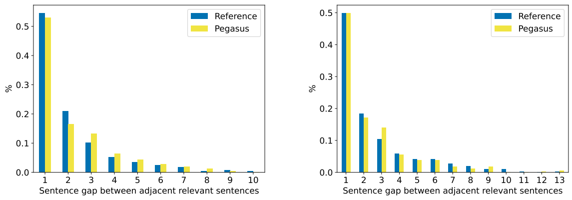
Table 5 is the prompt used for GPT-3.5 to obtain source-sentence scores.

C Analysis

In this section, we report a more detailed analysis on SourceSum.

C.1 Source sentence distribution

Figure 4 shows the sentence positions of source sentences. As expected, source sentences tend to appear at the beginning of the document, which supports the idea of using lead sentences as simple-yet-effective heuristics for news summarization (Zhu et al., 2021a). The position bias has also been reported in (Kryscinski et al., 2019) and (Zhu et al., 2021b). However, the plots also show that source



(a) XSum (b) CNN/DM

Figure 5: Distribution of the sentence interval between adjacent (ground-truth) source sentences. For example, if source sentence positions are 1, 3, and 7, the sentence intervals for the example are 2 and 4.

sentences spread over the document, which indicates that summarization involves more complex textual processing.

The sentence intervals between adjacent source sentences follow a similar distribution on XSum and CNN/DM. Figure 5 shows that source sentences generally distribute closely in the source document.

C.2 Qualitative Analysis

Table 6 shows ground truth and detected source sentences for a summary. Ground-truth source sentences are highlighted in green and the top-k results by each method are tagged at the end of each sentence. In this examples, we highlight the same number of source sentences as the number of ground-truth source sentences (i.e., $k = 2$ in the table). In this example, only Perplexity Gain successfully detected (S1) and (S8) as the source sentences for the summary.

Summary: The Nigg Energy Park in Ross-shire has been awarded a contract to assemble offshore wind turbines.
Input document: (S1) The site owned by Global Energy Group joins Wick Harbour in Caithness in securing work on the Â£2.6bn Beatrice Offshore Windfarm Ltd (Bowl) project. Perplexity$_1$ (S2) Siemens, one of the companies involved in Bowl, will use the yard for assembling turbines from spring 2018. (S3) Once assembled the turbines would be towed out to the wind farm site. ROUGE$_2$ BERTScore$_2$ (S4) The project, which also involves energy giant SSE, is to be created about eight miles off Wick. (S5) Global said Nigg's involvement would help to secure work for more than 100 people. (S6) The Scottish government, Highland Council, Highlands and Islands Enterprise, Scottish Council for Development and Industry (SCDI) and Scottish Renewables have welcomed the announcement. (S7) Business, Innovation and Energy Minister, Paul Wheelhouse, said: "Offshore renewables represent a huge opportunity for Scotland; an opportunity to build up new industries and to deliver on Scotland's ambitious renewable energy and carbon reduction targets for 2020 and beyond. (S8) "I am delighted that this multi-million pound contract between Global Energy Group and Siemens will enable Nigg Energy Park to develop into a genuine multi-energy site, securing around 100 direct and indirect jobs and associated supply chain opportunities. Perplexity$_2$ (S9) "This contract arising from installation of the Beatrice Offshore Wind farm will provide a very welcome boost to the local economy in Ross-shire and the wider Highland Council area." ROUGE$_1$ BERTScore$_1$ (S10) Regional director for the Highlands and Islands, Fraser Grieve, said: "Today's announcement of Nigg's involvement in the Beatrice Offshore wind project shows the positive economic impact that this major development will have on the region over the coming years. Cross-attention$_2$ (S11) "Nigg, and the wider Cromarty Firth, has much to offer and this agreement is not only a boost for the Global Energy Group but will benefit the supply chain through the area." (S12) Lindsay Roberts, senior policy manager at renewable energy industry group Scottish Renewables, said: "The contract signed today will help breathe new life into this Highland port. (S13) "Scotland's offshore wind industry has huge potential for both our economy and our environment, and it's great to see Nigg reaping the benefits. (S14) "As other wind farms with planning consent in the Scottish North Sea begin to develop, agreements like this will play a key role in securing benefits not just for communities on the east coast, but for the whole of Scotland." Cross-attention$_1$

Table 6: Output examples of the source sentence detection methods. The source sentences are highlighted in green. Tag(s) appended to the end of a sentence denote the method names and the ranks. In this example, only Perplexity Gain successfully detected (S1) and (S8) as the source sentences.

Input document: The Tories won 37 of 64 seats to claim a majority and wipe out Labour's 22-seat majority from 2013. Labour picked up 24 seats this time around, the Liberal Democrats won three and UKIP finished with none. Towns where seats turned from red to blue included Swadlincote, Matlock, Glossop, Buxton, Ripley, Belper and Ilkeston. Turnout was 38%. Election 2017: Full results from across England Conservative leader Barry Lewis described the result as "brilliant". "We didn't think at this point in the electoral cycle we'd be taking control of Derbyshire County Council," he said. "We fought a really good campaign on local issues and I think that's really helped. We got our manifesto out early and really hit the doorsteps." This was Labour's last stand - its last county council to be defended in England. And its defences have proven to be weak. The Conservatives have won across the south and centre of the county - in places like Heanor, Ilkeston and Ripley. They've also taken seats off the Lib Dems. And it was a bad night too for UKIP - their share of the vote in Derbyshire collapsed.	
Reference summary: The Conservatives have taken control of Derbyshire County Council with a massive swing from Labour.	
With all sentences (PEGASUS): The Conservatives have taken control of Derbyshire County Council.	
With source sentences only (PEGASUS): Conservative leader Simon Danczuk has said he is "delighted" his party has taken control of Derbyshire County Council.	
Input document: Stuart Campbell was arrested in the west of England on Friday following a complaint from a woman in south London. She had made allegations of harassment taking place over a two-year period. Mr Campbell, who was released on bail, said it concerned some tweets and insisted they were not threatening. He accused the media of "innuendo" designed to encourage "speculations". The blogger, a former computer games reviewer who was born in Stirling but lives in Bath, has been a vocal campaigner for Scottish independence and launched the Wings Over Scotland blog in 2011. On Friday he tweeted that he would be posting less frequently than usual because of "reasons totally outwith my control (don't ask)". End of Twitter post by @WingsScotland A spokesman for the Metropolitan Police said: "Police are investigating an allegation of online harassment. "The allegation was made after a woman, aged in her 30s, attended a south London police station. The harassment is said to have taken place over the past two years." Mr Campbell has been bailed, pending further inquiries, to a date in mid-September. In a statement on the Wings Over Scotland website, Mr Campbell responded to a report of his arrest which appeared in The Herald newspaper. He said that piece "has been written for maximum innuendo to allow the wildest speculations on social media - which are of course duly taking place - but the alleged events relate entirely to some tweets from our Twitter account, none of which have been deleted and all of which are still publicly visible. "Nothing more sinister or serious than some tweets has occurred or been alleged to have occurred. None of the tweets involved are in ANY way threatening, not even in a joking sense. That's all we'll be saying on the subject at this time."	
Reference summary: The pro-independence blogger behind the Wings Over Scotland website has been arrested for alleged online harassment.	
With all sentences (PEGASUS): A pro-independence blogger has been arrested on suspicion of online harassment.	
With source sentences only (PEGASUS): A prominent Scottish independence blogger has been released without charge after being arrested on suspicion of online harassment.	

Table 7: Examples of summaries generated with all sentences and with source sentences only (XSum). The source sentences in the input document are highlighted in green. Incorrect/hallucinated words are highlighted in purple.

Annotation guideline

Goals: Your task in this annotation is to provide the “highlighting” for document-summary pairs, and check the validity of summaries.

1. To identify the sentences in a document that contribute to a given summary of that document.
2. To determine whether a given summary is valid (all the important points in it are captured in the document itself).

Instructions: This annotation is a sentence-labeling task. For each snippet, you’ll see a summary (labeled SUMMARY:) and a short news article (labeled DOCUMENT:).

Summary: The summary appears in multiple places for each snippet in order to eliminate the need to scroll up and down. It is first shown before the document because it often functions as the first sentence of the article. Secondly, the summary appears in the **Prompt** box to the right of the editable window, so that you can always refer to it without needing to scroll.

Lastly, the summary appears at the bottom of the editable window, labeled SUMMARY: again. This final repetition is pre-tagged with the question Reconstructable? so that you can label it. As yourself, “Could I reconstruct all the important points of this summary based only the sentences I labeled as ‘1: contributes’?” and answer **Yes, reconstructable** or **No, not reconstructable**.

Document: The document is pre-annotated with sentence-boundaries. The end of each sentence is tagged with the question 0 or 1?. Mark sentences that are important to the provided summary as **1: contributes to summary**. Mark sentences that are not important to the summary as **0: doesn’t contribute to summary**.

Documents in this annotation are either CNN (three fifths) or BBC (two fifths) news articles. Some summaries are written by the articles’ authors, others are generated by models.

For Duplicates: You will sometimes see the same document multiple times, paired with a different summary each time. This can happen for two reasons:

1. because we are considering multiple sources of summaries, and
2. because original summaries for some articles were multiple sentences, and we are only displaying one summary sentence at a time.

Each document-summary *pair* that you see should be unique, however.

Annotation steps:

1. Read the summary at the top of the editing window, then read the document.
2. Evaluate each sentence for whether it provides information that contributes to the summary. (You can refer to the summary in the prompt on the right if you’ve scrolled down from the summary in the editing window.) Label every sentence in the document with one of the following labels:
   - **1: contributes to summary**: This sentence would be valuable in writing the summary.
   - **0: doesn’t contribute to summary**: The summary could be written without this sentence.

3. Now that you’ve read the document, assess whether the important points of the summary (repeated at the bottom of the document) are also present in the document itself. Answer the question, "Could you write this summary based solely on the sentences that you identified as important?"

- If so, label the summary at the bottom of the document with **Yes, reconstructable**.
- If you would need additional information to write the summary, OR if the summary contradicts the document, then label it as **No, not reconstructable**.
- You can also change the labels of sentences in the document if you realize that more of them are needed in order to write the summary.

4. When all sentences have been labeled and you’ve evaluated the summary, click "Submit" and review your annotations.

- Read over just the sentences that you marked as **1: contributes to summary**, and confirm that each of them contains information that the summary directly includes.
- If you labeled the summary as **Yes, reconstructable**, verify that all the important information in it is contained in the sentences marked **1: contributes**.

Table 8: Annotation guideline.

Reconstructability judgment guideline—Determining whether a summary is reconstructable
We'll count a summary as valid and reconstructable if all the **important points** in it can be reconstructed from the document by a reader who is part of the document's **intended audience**. What counts as an "important point" is somewhat subjective, but here is some guidance: **Important to be able to reconstruct from the document:** • All named entities (e.g., Wales Under-20, Samoa, World Rugby U20 Championship, Georgia): If a name appears in the summary, it is an important point in the summary. Only mark the summary as reconstructable if the name or entity also appears in the document. It's okay if a co-referring expression (but not the exact name itself) appears in the document. • Events • Approximate quantities; exact values don't need to be reconstructable (e.g., "10,000 free racquets" in the summary could be supported by "many free racquets" in the document; "1.9% increase" could be supported by "about 2% increase") **Not important to be able to reconstruct from the document:** • *Expansions of acronyms or abbreviations*: If the full phrase that an acronym stands for appears in the summary but not in the document, the summary can still be considered reconstructable; the expansion of the acronym is a minor point in the summary, not an important point. Different expressions that refer to the person or place mentioned in the summary qualify as • *Exact numbers* are not important. No need to break out the calculator. Information sources (e.g., "State television reports", "Official figures show"). **Some summary examples with important information in *italics*:** • *Wales Under-20* ran in *eight tries* to *beat Samoa* and secure their *first win* of *the World Rugby U20 Championship* in *Georgia*.   – If the document provides enough information to conclude that there were several tries, but doesn't specify eight tries, that's fine. • *Shares* in the baby formula milk firm *Bellamy* have *plunged* after a warning that *new import regulations in China* will cut into revenues. • *Iran's President Mahmoud Ahmadinejad has sacked Health Minister Marziyeh Vahid Dastjerdi*, the *sole woman in his cabinet*, state television reports. **Summary should be reconstructable by the document's intended audience** • For many articles from the BBC news corpus, you may not have the contextual knowledge that the author assumes the audience to have. This is particularly glaring in the case of sports articles. • We don't mean for you to have to Google proper nouns in order to do this annotation. If you can infer from the document that two expressions co-refer (e.g., "Prime Minister" in the summary and the individual's actual name in the document; country name in the summary and the specific town in the document), then you can consider the entity to be "reconstructable" even if you don't personally have the real-world knowledge to verify that the entities are the same. • The exception is if you can't make sense of the article at all without doing a search. Please leave a comment on Anagram if you need to use a search engine to get relevant context in order to comprehend the basics of the article.

Table 9: Reconstructability judgment guideline.