Revisiting Zero-Shot Abstractive Summarization in the Era of Large Language Models from the Perspective of Position Bias

Anonymous ACL submission

Abstract

We characterize and study zero-shot abstractive summarization in Large Language Models (LLMs) by measuring *position bias*, which we propose as a general formulation of the more restrictive *lead bias* phenomenon studied previously in the literature. Position bias captures the tendency of a model unfairly prioritizing information from certain parts of the input text over others, leading to undesirable behavior. Through numerous experiments on four diverse real-world datasets, we study position bias in multiple LLM models such as *GPT 3.5-Turbo*, *Llama-2*, and *Dolly-v2*, as well as state-of-the-art pretrained encoder-decoder abstractive summarization models such as *Pegasus* and *BART*. Our findings lead to novel insights and discussion on performance and position bias of models for zero-shot summarization tasks.

1 Introduction

Deep learning based abstractive text summarization models and Large Language Models (LLMs) have shown remarkable progress in generating concise and coherent summaries from input articles that are comparable to human-written summaries (Zhang et al., 2023). Building upon this research, we aim to quantitatively measure summarization performance of LLMs (and pretrained encoder-decoder models for reference) by proposing *position bias*, which is a novel and general formulation of the *lead bias* phenomenon (Liu and Lapata, 2019).

Position bias refers to the tendency of models to prioritize information from certain parts of the source text, potentially overlooking crucial details in other parts of the input article. While position bias has been studied previously in the literature as *lead bias*, we posit that lead bias is a specific case of position bias. Most prior works in this domain aim to propose methods that either *incorporate* or *alleviate* lead bias in models for improved performance (Xing et al., 2021; Zhu et al., 2021) without a thorough analysis of the problem itself. It is also important to note that a formal definition for lead bias is still currently lacking in related work.

In contrast to lead bias, position bias seeks to decipher if models are over-utilizing sentences from any section(s) of the input articles, instead of just the leading segment. Moreover, a model's output summary can only be considered positionally biased if it overwhelmingly uses sentences from section(s) of the input that the human-written (or *gold*) summaries do not use themselves. For instance, if gold summaries for a dataset are lead biased and the model generates lead biased summaries, this is desirable behavior and cannot constitute position bias. In this scenario, if the model were to generate *tail biased* summaries, it would be regarded as position bias. An example of a *positionally biased* summary is shown in Figure 1.

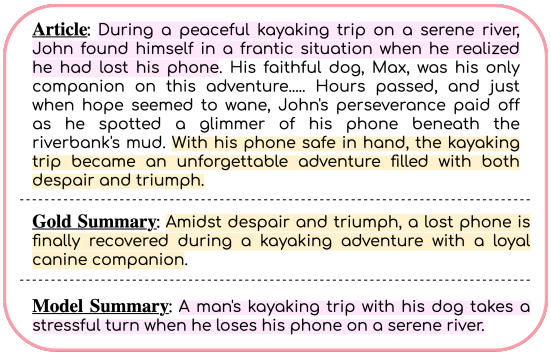

Figure 1: An example of *position bias* where gold summary is *tail biased* and model summary is *lead biased*.

We show how position bias can be empirically estimated by generating a distributional mapping between summary sentences and the article sentences used to generate the summary. Then, position bias can be measured using a metric such as Wasserstein distance (Vaserstein, 1969) between the model generated summary distribution and the gold summary distribution. Position bias measurements augmented with traditional metrics such as ROUGE scores (Lin, 2004) can provide a more holistic evaluation of zero-shot summarization models.

In summary, we make the following contributions in this work:

- We generalize and formalize the notion of lead bias as *position bias* in zero-shot abstractive summarization. Lead bias can then be understood as a specific case of position bias (Section 3.2).
- We show how position bias can be empirically estimated for a given zero-shot summarization model and hence, can be employed as a metric for summarization quality alongside traditional metrics such as ROUGE scores. We conduct extensive experiments to benchmark LLMs (*GPT 3.5-Turbo*[1], *Llama2-13B-chat*[2], and *Dolly-v2-7B* (Conover et al., 2023)) and pretrained encoder-decoder models (*Pegasus* (Zhang et al., 2020) and *BART* (Lewis et al., 2020)) on 4 diverse datasets: *CNN/DM* (See et al., 2017), *Reddit TL;DR* (Kim et al., 2018), *News Summary* (Ahmed et al., 2018), and *XSum* (Narayan et al., 2018) (Section 4).
- Using our findings, we compile novel insights to aid practitioners in selecting the right model for their zero-shot summarization tasks. (Section 5)

2 Related Works

Related work has studied the more specific phenomenon of lead bias in summarization. Both Grenander et al. (2019) and Xing et al. (2021) propose approaches and architectural changes to models that can reduce lead bias in *extractive summarization*, where summary sentences are selected directly from the source text. In contrast, in our work we study position bias more generally in *abstractive summarization*. Interestingly, Zhu et al. (2021) seek to leverage lead biased pre-training to improve performance on news articles, which are known to be lead biased. Prior work has also analyzed LLMs for their performance as zero-shot abstractive summarizers (Retkowski, 2023). Goyal et al. (2022) study GPT-3 specifically in the context of news summarization and Zhang et al. (2023) benchmark the summarization performance of multiple LLMs on the CNN/DM and XSum datasets. Tam et al. (2023) study the factuality of summaries generated by LLMs and Shen et al. (2023) use GPT 3.5-Turbo for evaluating summaries generated by other models. Unlike our work, none of these works have studied position bias of zero-shot abstractive summarization in LLMs.

[1] https://platform.openai.com/docs/models/gpt-3-5

[2] https://huggingface.co/meta-llama/Llama-2-13b-chat-hf

3 Proposed Approach

3.1 Zero-Shot Abstractive Summarization

A zero-shot abstractive text summarization model $\mathcal{A}$ operates on the dataset tuple $D = (X, G)$ where X is a set of articles and G are their corresponding reference *gold* summaries, generally written by human annotators. Moreover, each article and its corresponding gold summary consists of a variable number of sentences. The model $\mathcal{A}$ then takes in as input the set of articles in the set X and outputs a summary, i.e., $\mathcal{A}(X) = S$ where S is the generated summary. Traditionally, the model is evaluated by comparing the generated summaries (S) with the gold summaries (G) using the ROUGE metric (Lin, 2004). We use $R^1/R^2/R^L$ to denote average ROUGE-1, ROUGE-2, and ROUGE-L scores.

3.2 Formulating and Estimating Position Bias

Let an article $x \in X$ have $|x| = N_x$ number of sentences. We also obtain the set of generated summaries as $S = \mathcal{A}(X)$ where each $s \in S$ has N_s number of sentences. Since we consider abstractive summarization[3], let us also assume we have a mapping function ϕ that takes in a summary sentence $s_i \in s$ and maps it back to a sentence $x_j \in x$ in the article that it was primarily derived from. Any similarity function can be employed as a useful approximation for such a mapping function ϕ.[4]

Most works on lead bias implicitly assert that lead bias exists if for most $s_i \in s$, $\phi(s_i)$ maps to some x_j that lies between the first $(0, k']$ sentence positions of the article. Here $k' \ll N_x$ and can be a dataset specific parameter– for example, for the Lead-3 (Liu and Lapata, 2019) evaluation metric, $k' = 3$. However, this does not seem to be a reasonable definition, especially when considering general position. For example, consider a model which tends to derive information for generating summaries by using only the last few sentences of the article. This *tail bias* might also constitute undesirable behavior if the gold summaries are not tail biased themselves, but will not be accounted for in the lead bias paradigm. Hence, it is better to reason about position more generally.

Since articles can be of differing lengths, position becomes specific to an individual article. To overcome this issue, we divide each article x into K segments of approximately equal length

[3] In *extractive summarization*, there is an exact one-to-one mapping between summary and article sentences.

[4] ROUGE or TF-IDF vector similarities are some examples.

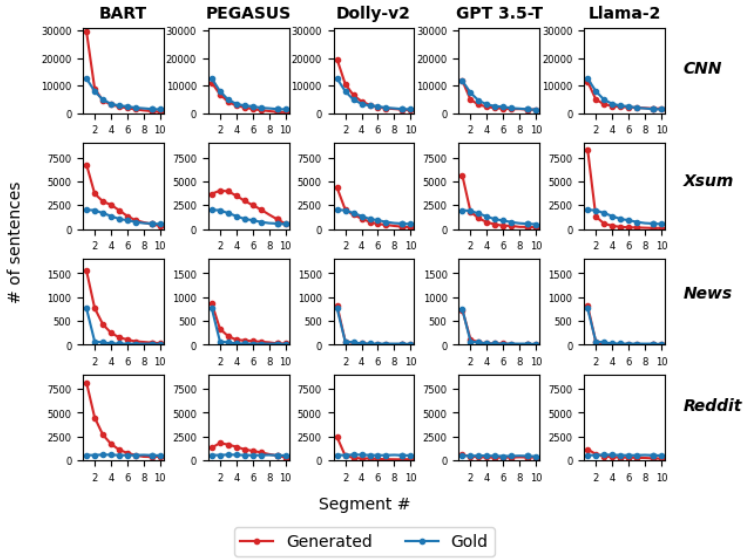

Figure 2: Visualizing positional distributions of gold and model generated summaries for all datasets. The more "different" these distributions are for a given dataset/model, the more *position biased* the model is for that dataset.

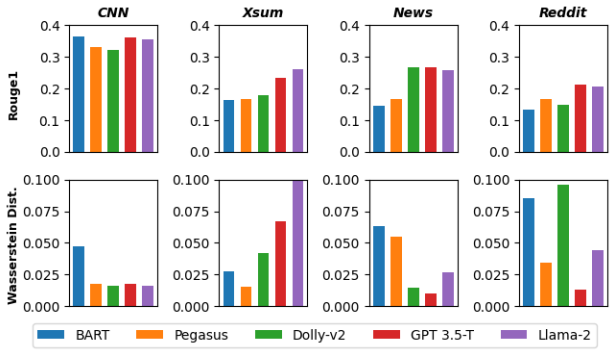

Figure 3: Measuring *performance* (R^1 score) and *position bias* (Wasserstein distance between gold and generated summaries' positional distributions). Lower Wasserstein distance values correspond to lower position bias.

(refer to Appendix A for how to do this) where $K \leq \min_{x \in X} |x|$. This results in each article having at least one sentence in a segment, and we now have a uniform way of measuring sentence position across articles irrespective of their length.

To quantify position bias, we first obtain which article sentences the summary sentences are derived from using ϕ, for both gold G and generated $\mathcal{A}(X)$ summaries. Then, we can map these article sentences to article segments to obtain a general sense of position in the article. We now have a distributional mapping of summary sentences to article segments. Using Wasserstein distance (Vaserstein, 1969) between the G and $\mathcal{A}(X)$ *positional distributions* we can then measure position bias.

4 Results

For experiments, we consider the *CNN/DM*, *XSum*, *News*, and *Reddit* datasets. All datasets are different, in terms of domain, inherent position biases, or article and gold summary length. We choose only instruction-tuned LLMs as they are more performant at summarization (Retkowski, 2023) and we cover different model sizes: GPT 3.5-T is large (175B params), Llama-2 is mid-size (13B params), and Dolly-v2 is small (7B params). We also consider SOTA pretrained encoder-decoder models such as Pegasus and BART, although these models are not performant unless fine-tuned (i.e. many-shot learning). All experiments are done on the test set of datasets (more details in Appendix F).

We first visualize the positional distributions generated using our mapping procedure for the gold summaries and model generated summaries in Figure 2 with $K = 10$. As can be seen, *CNN* and *News* contain lead biased gold summaries and *Reddit* and *XSum* are positionally uniformly distributed. It can also be seen that the LLMs tend to have low position bias on *CNN, News,* and *Reddit* datasets. However, for *XSum*, which constitutes the *extreme sum-*

marization setting, LLMs as well as BART/Pegasus exhibit much more lead bias, which is absent from the gold summaries. In *XSum*, the articles are up to 286 sentences long, and summary lengths are required to be between 1-2 sentences long. This is the largest jump from article $\rightarrow$ summary and might explain models' tendency to pick a single sentence from the leading segments of the article. Comparing even with *Reddit* where article lengths are up to 23 sentences long and summaries are 1-17 sentences long, it is evident that *XSum* poses a unique challenge for summarization models.

Next, in Figure 3, we measure model performance using the R^1 score between gold and generated summaries (results for R^2, R^L are provided in Appendix B and follow similar trends) and position bias using Wasserstein distance between the positional distributions of gold and generated summaries of Figure 2. As is evident, LLMs attain the highest ROUGE scores on all datasets, and tend to have very low position bias, with the exception of *XSum*. However, even for *XSum*, LLMs achieve excellent performance on ROUGE. BART/Pegasus tend to have low performance and BART is also heavily position biased across all datasets.

5 Discussion

Insights on Model Performance and Biases.

• *GPT 3.5-T consistently attains low position bias and high performance.* Generally, GPT 3.5-T should be the de-facto choice for users, as can be seen in Figure 3. It consistently obtains high ROUGE scores and low position bias values. However, the paid API and closed-source access might be unfavorable to some users. For open-source models, Llama-2 is the better choice compared to Dolly-v2, and at times obtains ROUGE scores higher than even GPT 3.5-T (for e.g. on *XSum*). In comparison, Dolly-v2 at times has arbitrary and unpredictable performance, such as its low ROUGE scores and large position bias on the *Reddit* dataset, unlike the other LLMs on the same dataset.

• *LLMs might exhibit significant lead bias for extreme summarization.* LLMs exhibit strong lead bias in the extreme zero-shot summarization case (Figure 3, *XSum*). For users who wish to undertake a similar task (pick 1-2 sentence summaries from very lengthy articles), LLMs might tend to only select sentences/information from the beginning of the article. If this is undesirable, it would be recommended to instead collect gold summaries and finetune LLMs/models to counteract this.

• *Suitability of encoder-decoder models.* As zero-shot summarizers, pretrained encoder-decoder models like BART and Pegasus have high position bias and low performance. This likely stems from their need to be finetuned on article-gold summary tuples to achieve SOTA performance. However, we would like to caution users to ensure that there is no positional mismatch between the data they finetune on and their evaluation set.[5] While obvious, not ensuring this can lead to low ROUGE scores and high position bias (we demonstrate this in Appendix C).

Choice of Mapping Function ϕ. For experiments in the paper we use TF-IDF vector similarities as ϕ. In our preliminary experiments, we did not observe significant differences for other choices. We provide these additional results when the R^1 score is used instead and compare with the original results (Appendix D). Future work can analyze this choice of ϕ as well as effect of other values of K.

Correlation of Position Bias and ROUGE. An interesting consequence of our ROUGE and position bias results on datasets shows that their correlation is highly data dependent. For e.g., for *XSum* Spearman's correlation shows statistically significant high positive correlation (≈ 0.89) between Wasserstein distances and R^2 scores across all models but for *Reddit* there is significant negative correlation (≈ -0.89). See Appendix E for detailed results. Hence, ROUGE scores are not enough to assess position bias. This also makes intuitive sense as ROUGE simply measures n-gram overlap and cannot holistically evaluate models (Cohan and Goharian, 2016). In future work other evaluation metrics can be studied alongside position bias.

6 Conclusion

We analyze zero-shot abstractive summarization by LLMs via a novel formulation of position bias. Position bias measures the tendency of models to generate summaries which overtly and unfairly utilize certain portions of input text over others. Through extensive experiments on the *CNN/DM, XSum, Reddit, News* datasets, as well as various models (GPT 3.5-T, Llama-2, Dolly-v2, Pegasus, BART), we obtain novel insights about model performance and position bias that contribute to a deeper understanding of the challenges and opportunities in leveraging LLMs for effective abstractive summarization.

[5] For e.g. finetuning on large # of news articles collected over the internet to then summarize in a different domain.

Limitations

Our work formulates the concept of position bias in abstractive summarization and analyzes it in LLMs (and other reference models) across four diverse datasets: *CNN/DM, XSum, Reddit,* and *News Summary*. The main limitation is that position bias of LLMs needs to be evaluated on many more datasets, and on other diverse problem settings beyond the ones considered in our paper. Moreover, the source domain itself could be challenging (legal or medical documents) or the LLM might not have been trained with data from that domain. In such cases, the LLM might default to using certain sections of the input articles over others, resulting in position bias. Another limitation of our work has been the primary use of English language datasets, but it is important to benchmark LLM position bias using summarization datasets from other languages as well. Finally, a limitation of studies on LLMs such as GPT 3.5-Turbo is that they are constantly being updated and improved, and some behaviors might change or become non-existent in future versions. This necessitates assessing model performance/biases over time. In future work, we seek to alleviate these limitations.

Ethics Statement

Our work on position biases in LLMs is important for understanding how these models prioritize information, and whether or not they disproportionately emphasize specific sections of the source text when generating abstractive summaries in a zero-shot setting. As LLMs are further integrated in society and utilized in various application pipelines, it is crucial to understand their behavior in a transparent manner. Through this study, we wish to shed light on this issue and allow practitioners to understand undersirable model behavior with regards to the summarization task better. This work also enables users to understand scenarios in which these models will generate more reliable outputs, leading to safer outcomes in practice.

References

Hadeer Ahmed, Issa Traore, and Sherif Saad. 2018. Detecting opinion spams and fake news using text classification. *Security and Privacy*, 1(1):e9.

Arman Cohan and Nazli Goharian. 2016. Revisiting summarization evaluation for scientific articles. *arXiv preprint arXiv:1604.00400*.

Mike Conover, Matt Hayes, Ankit Mathur, Xiangrui Meng, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, et al. 2023. Free Dolly: Introducing the world's first truly open instruction-tuned LLM.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. News summarization and evaluation in the era of GPT-3. *arXiv preprint arXiv:2209.12356*.

Matt Grenander, Yue Dong, Jackie Chi Kit Cheung, and Annie Louis. 2019. Countering the effects of lead bias in news summarization via multi-stage training and auxiliary losses. 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 6019–6024.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2018. Abstractive summarization of reddit posts with multi-level memory networks. *arXiv preprint arXiv:1811.00783*.

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.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

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)*, page 3721. 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. *arXiv preprint arXiv:1808.08745*.

Fabian Retkowski. 2023. The current state of summarization. *arXiv preprint arXiv:2305.04853*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Chenhui Shen, Liying Cheng, Yang You, and Lidong Bing. 2023. Are large language models good evaluators for abstractive summarization? *arXiv preprint arXiv:2305.13091*.

Derek Tam, Anisha Mascarenhas, Shiyue Zhang, Sarah Kwan, Mohit Bansal, and Colin Raffel. 2023. Evaluating the factual consistency of large language models through news summarization. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Leonid Nisonovich Vaserstein. 1969. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredachi Informatsii*, 5(3):64–72.

Linzi Xing, Wen Xiao, and Giuseppe Carenini. 2021. Demoting the lead bias in news summarization via alternating adversarial learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 948–954.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. 2023. Benchmarking large language models for news summarization. *arXiv preprint arXiv:2301.13848*.

Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2021. 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*, pages 1462–1471.

Appendix

A Dividing Articles into K Segments of (Approximately) Equal Length

To overcome the issue of articles being of differing lengths, we need better mathematical structure for describing *position* across articles in a dataset. For this, we wish to divide an article x into K segments of approximately equal length. To achieve this, the j-th segment will contain the sentences of the article that lie in the interval:
$[(j-1)\cdot c + \min(j-1, d), j\cdot c + \min(j, d) - 1]$, where $K \leq \min_{x\in X} |x|$, $c = \lfloor \frac{N_x}{K} \rfloor$, and $d = N_x \mod K$.

The aim is to distribute the content of the article into K segments in a way that makes the lengths of these segments as equal as possible. Here, the inequality $K \leq \min_{x\in X} |x|$ ensures that the number of desired segments K should not exceed the length of the shortest article in the set of articles X (otherwise it will lead to empty segments for those articles). The content of each segment j (note, j represents the index of the segment from 1 to K) is determined by an interval defined by: $(j-1)\cdot c + \min(j-1, d)$ (lower bound) and $j\cdot c + \min(j, d) - 1$ (upper bound).

Intuitively, $c = \lfloor \frac{N_x}{K} \rfloor$ calculates the approximate length of each segment as it divides the total number of sentences in the article (N_x) by the desired number of segments (K) and rounds down to the nearest whole number. However, N_x might not be fully divisible by K and hence, we might have remainder $d = N_x \mod K$. Hence, d accounts for any additional content that cannot be evenly distributed among the segments and ensures that segments accommodate the variation in article lengths. In this manner, the terms $\min(j-1, d)$ and $\min(j, d)$ in the lower and upper bounds of the interval are used to account for potential variations in segment length due to the remainder d.

B Additional Results for Other ROUGE Metrics

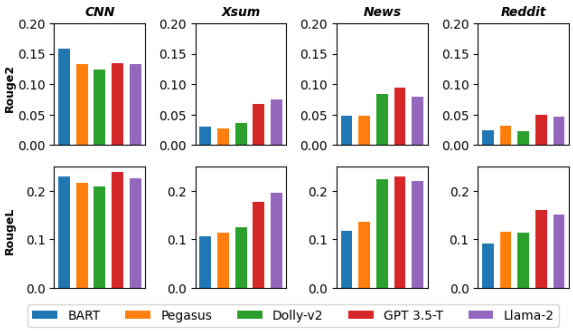

Figure 4: Additional results for R^2 and R^L metrics.

In the main paper in Figure 3 we provided results for the ROUGE-1 (R^1) score. Here, we provide additional results for the ROUGE-2 (R^2) and ROUGE-L (R^L) scores measured between the gold and model generated summaries as Figure 4. It can be seen that the trends are similar to R^1 and LLMs exhibit stellar performance for R^2 and R^L across all datasets.

C Additional Position Bias Results for Finetuning BART and Pegasus

We go beyond the zero-shot setting to provide additional results on measuring position bias when BART and Pegasus are finetuned on the datasets we consider. The training was carried out on one *NVIDIA-A100* with 50 GB memory. We use the HuggingFace Seq2Seq Trainer Class with a batch

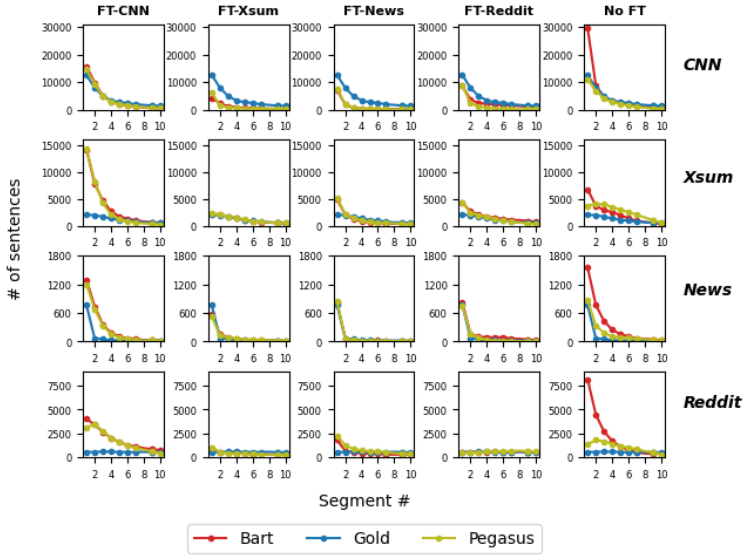

Figure 5: Visualizing positional distributions of gold and Pegasus/BART generated summaries for all datasets with and without finetuning on a particular dataset (training set). For the finetuned models, the diagonal subfigures are the ones that have the same finetuning and evaluation datasets and have low position bias. All other subfigures have a *mismatch* between finetuning and evaluation datasets, and exhibit high levels of position biases. That is, the model generated summary positional distribution is very different from the gold summary positional distribution. The no-finetuning results were also shown in Figure 2 and are provided again for reference.

size of 64, gradient checkpointing of 4 and gradient accumulation. We use mixed-precision training for all models. The learning rate for all models was set to *5.6e-5*. While generating summaries during finetuning we use a single beam and maximum generation length of 128.

We finetune on the training set of each of the 4 datasets and evaluate on all datasets (for reference we again provide the no-finetuning / zero-shot results of Figure 2). Results for the obtained positional distributions are shown in Figure 5. It is evident that if there is mismatch in the finetuning and evaluation datasets for pretrained encoder-decoder models, they exhibit high position bias, leading to biased summarization. Hence, it is important for practitioners to collect article-summary data for finetuning that exactly reflects their evaluation or production use-case.

D Additional Results for Different ϕ

For experiments in the main paper, we opt for TF-IDF vector similarities as the choice of the mapping function ϕ due to computational efficiency (over computing individual ROUGE scores between summary and article sentences for e.g.). However, it is important to examine whether this choice significantly impacts results, trends, and our findings. In initial experiments with different ϕ we concluded that this choice does not affect results. In Figure 6 we provide results that support this by using R^1 (ROUGE-1) as the metric for ϕ on the *Reddit* and *News* datasets for Llama-2 generated summaries. We compare the gold summary and generated summary positional distributions for both datasets when ϕ is computed using TF-IDF vectors and R^1. It is clear that the trends and results are the same for both ϕ. Even the Wasserstein distance values computed between gold and generated summaries do not change much. For e.g. on *Reddit*: for TD-IDF the distance value is 0.044 and for R^1 it is 0.046. Despite no significant differences, we believe future work can explore the choice of ϕ more deeply.

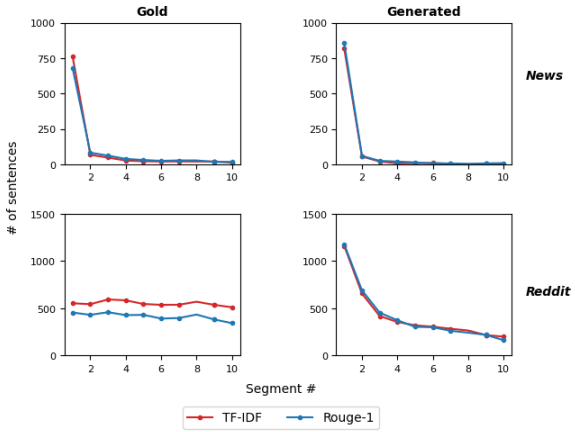

Figure 6: Results on *News* and *Reddit* for Llama-2 when ϕ is either TF-IDF similarity or ROUGE-1.

E Additional Results for Measuring Correlation Between ROUGE and Position Bias

Table 1: Measuring Spearman's correlation coefficient between position bias (Wasserstein distances) and ROUGE metrics for all datasets (* denotes p-values of ≤ 0.1 and ** denotes p-values of ≤ 0.05).

Dataset	Metric	Correlation
CNN/DM	R^1	0.499
	R^2	0.799*
	R^L	0.300
XSum	R^1	0.899**
	R^2	0.999**
	R^L	0.899**
News	R^1	-0.999**
	R^2	-0.899**
	R^L	-0.999**
Reddit	R^1	-0.799*
	R^2	-0.899**
	R^L	-0.799*

In this section, we measure the correlation between ROUGE scores (Lin, 2004) and Wasserstein distance computed between the gold summary and model generated summary distributions. We conduct this experiment using Spearman's correlation coefficient statistic over all models and for each dataset. We utilize the R^1, R^2, R^L ROUGE metrics individually for this analysis, and the results are shown in Table 1. We find that correlation is highly dependent on the dataset: for *CNN* the correlation is not strong and the results are not statistically significant, for *XSum* ROUGE and position bias are positively correlated and statistically significant, and for *News* and *Reddit* results are statistically significant but highly negatively correlated. This indicates that ROUGE itself is not enough to assess position bias and hence, independent position bias measurement is important for holistic summarization evaluation.

F Dataset, Model, and Training Details

F.1 Detailed Dataset Information

XSum (Narayan et al., 2018): The *XSum* dataset contains over 200K short, one-sentence news summaries answering the question "What is the article about?" and was collected by harvesting online articles from the British Broadcasting Corporation (BBC). The testing set consists of 11334 articles. The average number of sentences in the articles are 19.105. The total number of sentences in the summaries are 11334, leading to an average of 1 sentence per summary.

CNN/DM (See et al., 2017): The *CNN/DM* dataset contains 300K unique news articles as written by journalists at CNN and the Daily Mail and is one of the most popular datasets for abstractive/extractive summarization and abstractive question answering. The testing set consists of 11490 articles. The average number of sentences in the articles was 33.37. The total number of sentences in the summaries was 43560 (an average of 3.79 sentences per summary).

Reddit TL;DR (Kim et al., 2018): The *Reddit* dataset consists of 120K posts from the online discussion forum Reddit. The authors used these informal crowd-generated posts as text source, in contrast with existing datasets that mostly use formal documents as source such as news articles. We used an 80-20% train-test split to obtain 4214 articles in the test set. The average number of sentences per article was 22.019. The total number of sentences in the summaries was 6016 which leads to an average of 1.4276 sentences per summary.

News Summary (Ahmed et al., 2018): The *News* dataset was initially created for fake news classification. We used the testing set comprising of 1000 articles. The number of sentences in the summaries are 1012 (an average of 1.012 per summary)

F.2 Models

Pegasus (Zhang et al., 2020): The Pegasus model family is used mainly for text-summarization tasks. We use the *google/pegasus-large* checkpoint[6] from Huggingface as the summarization model.

BART (Lewis et al., 2020): BART is a Seq2Seq encoder-decoder model for language tasks. We use the *facebook/bart-large* checkpoint[7] from Huggingface as the summarization model.

GPT 3.5-T [8]: GPT-3.5-turbo is OpenAI's flagship LLM which has been instruction-tuned and optimized for chat purposes. We utilized the model from Microsoft Azure's OpenAI service and the version was the August 3rd version.

Llama2-13B-chat [9]: Meta developed and publicly released the Llama-2 family of LLMs, a collection of pretrained and fine-tuned generative text models ranging in scale from 7-70 parameters. The chat versions of the models are optimized for dialogue

[6] https://huggingface.co/google/pegasus-large
[7] https://huggingface.co/facebook/bart-large
[8] https://platform.openai.com/docs/models/gpt-3-5
[9] https://huggingface.co/meta-llama/Llama-2-13b-chat-hf

via instruction finetuning. We generated inferences by modifying the PyTorch code provided in the official Github repository: https://github.com/facebookresearch/llama.

Dolly-v2-7B (Conover et al., 2023): Dolly-v2-7B is a 6.9 billion parameter causal language model created by Databricks that is derived from EleutherAI's Pythia-6.9B model and finetuned on a 15K instruction corpus generated by Databricks employees. We used the *databricks/dolly-v2-7b* checkpoint[10] from HuggingFace as the summarization model.

F.3 Generating Summaries via LLMs

We provide the prompts used to generate summaries for each LLM and each dataset (prompts might differ between datasets for the same model due to different summary requirements, and they might differ across models as different models respond to input text differently). Note that *{Article}* in each prompt should be replaced by the article to be summarized. It is also important to note that the prompts were adapted iteratively through multiple experiments to ensure that models followed the prompt as closely as possible. At times models did not follow the prompt specifications exactly and would generate more summary sentences than required for that dataset (for e.g. GPT 3.5-T followed exact prompt specifications 74.9% of the time). Hence, for parity between dataset and model summaries, and fair comparison between all models, we uniformly randomly sampled (so as to not add inductive bias) the number of sentences required from the generated output. Also, due to OpenAI's content moderation policy GPT 3.5-T did not give responses for a minority of inputs (6.16% of all input). We believe future LLM versions will improve along these lines to always follow prompts exactly as specified so post-hoc measures will not be required. We now provide prompts below.

F.3.1 Prompts for GPT 3.5-T

Xsum: *For the following article: {Article}. Return a summary comprising of 1 sentence. Write the sentence in a dash bulleted format.*

CNN/DM: *For the following article: {Article}. Return a summary comprising of 3 sentences. Write each sentence in a dash bulleted format.*

Reddit: *For the following article: {Article}. Return a summary comprising of 1 sentence. Write the sentence in a dash bulleted format.*

News: *For the following article: {Article}. Return a summary comprising of 1 sentence. Write the sentence in a dash bulleted format.*

F.3.2 Prompts for Llama2-13B-chat

Xsum: *For the following article: {Article}. Return a summary comprising of 1 sentence. Write the sentence in a numbered list format.*
For example:
1. First sentence

CNN/DM: *For the following article: {Article}. Return a summary comprising of 3 sentence. Write the sentence in a numbered list format.*
For example:
1. First sentence
2. Second sentence
3. Third sentence

Reddit: *For the following article: {Article}. Return a summary comprising of 1 sentence. Write the sentence in a numbered list format.*
For example:
1. First sentence

News: *For the following article: {Article}. Return a summary comprising of 1 sentence. Write the sentence in a numbered list format.*
For example:
1. First sentence

F.3.3 Prompts for Dolly-v2-7B

Xsum: *Generate a 1 sentence summary for the given article. Article: {Article}.*

CNN/DM: *Generate a 3 sentence summary for the given article. Article: {Article}.*

Reddit: *Generate a 1 sentence summary for the given article. Article: {Article}.*

News: *Generate a 1 sentence summary for the given article. Article: {Article}.*

G Code and Reproducibility

We open-source our code and provide it as a Github repository: https://anonymous.4open.science/r/position_bias. The repository contains explicit instructions for how to reproduce our results and analyze the findings for each model. We used Python 3.8.10 for all experiments. The experiments were conducted on Ubuntu 20.04 using NVIDIA GeForce RTX A6000 GPUs running with CUDA version 12.0.

[10] https://huggingface.co/databricks/dolly-v2-7b