SUWMIT at BioLaySumm2025: Instruction-based Summarization with Contrastive Decoding

Priyam Basu, Jose Cols, Daniel Jarvis, Yongsin Park, Daniel Rodabaugh
Department of Linguistics, University of Washington
{pbasu77,jcols,dljarvi,yongsinp,drodaba}@uw.edu

Abstract

In the following paper, we present our team's approach to subtask 1.1 of the BioLaySumm 2025 shared task, which entails the automated generation of lay summaries from biomedical articles. To this end, we experiment with a variety of methods for text preprocessing, extractive summarization, model fine-tuning, and abstractive summarization. Our final results are generated on a fine-tuned Llama 3.1 Instruct (8B) model, notably achieving top scores on two out of four relevance metrics, as well as the highest overall ranking among this year's participating teams on the plain lay summarization subtask.

1 Introduction

Biomedical articles often contain information of interest to audiences beyond the community of medical researchers and practitioners; however, the large volume of content, in combination with domain-specific technical language, often leaves such text unsuited for consumption by non-experts. The automated generation of lay summaries may, therefore, serve as a tool for improving the accessibility of scientific publications to a broader public by offering a non-technical glance to potential readers (Goldsack et al., 2024). Following previous iterations initiated by Goldsack et al. (2023), the BioLaySumm 2025 shared task presents precisely this objective, calling for teams to make use of the PLOS and eLife datasets (Goldsack et al., 2022; Luo et al., 2022b) to build automated summarization systems with a focus on ease of understanding while maintaining relevance and factuality (Xiao et al., 2025).

Winners of the BioLaySumm 2023 shared task (Turbitt et al., 2023) saw success in generating summaries based on the abstracts of articles and leveraging domain knowledge of GPT-style models, with summaries generated by their system offering better relevance and factuality scores than the fine-tuned BioGPT (Luo et al., 2022a) model they tested against, though at the cost of readability. Winners of the BioLaySumm 2024 (You et al., 2024) subsequently investigated an alternative approach to the fine-tuning of the model, using TextRank (Mihalcea and Tarau, 2004) to extract the most salient content before passing it to a GPT model for summarization, augmented by a BERT-based clustering technique and a keyword-based method to extract definitions from the Wikipedia dataset. Another team, Modi and Karthikeyan (2024), achieved top factuality scores by running preprocessing methods over article abstracts before passing content through an LLM.

Building on the success of these previous teams, we develop and publicly release an open-source,[1] end-to-end pipeline to facilitate rapid experimentation in summarization (Section 3.1). Our best model results from experiments conducted through this pipeline.

2 Data

The shared task organizers have made available two datasets, PLOS and eLife (Goldsack et al., 2022; Luo et al., 2022b), which include biomedical research articles and their corresponding expert-written lay summaries. Together, these datasets comprise a total of 29,119 training instances and 1,617 validation instances, with approximately 85% of instances sourced from PLOS, and the remaining 15% from eLife. Additional dataset statistics are provided in Appendix B.

3 Methods

In this section, we provide an overview of the methodology used for our final submission, which is an abstractive summarization model based on Meta's Llama 3.1 Instruct (8B) (Grattafiori et al., 2024). Although this model did not perform the

[1] https://github.com/whopriyamuw/biolaysumm2025-task

BioNLP 2025 Shared Tasks, pages 240–248
August 1, 2025 ©2025 Association for Computational Linguistics

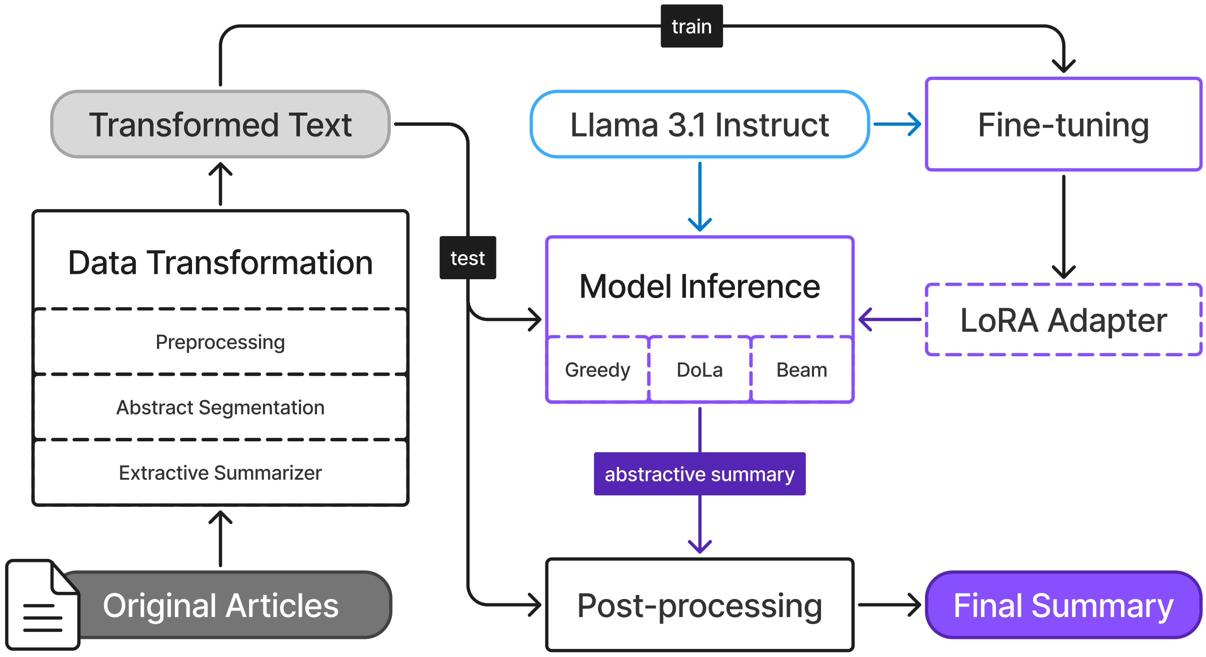

Figure 1: Our proposed pipeline for rapid experimentation comprises four toggleable modules: data transformation, model fine-tuning, model inference, and post-processing. We conducted over 20 experiments using distinct combinations of these modules. Dashed boxes denote optional or composable functionality.

best in all our experiments (Section 4), it offers the most balanced performance across the three groups of evaluation metrics: *relevance*, *readability*, and *factuality* (see Section 3.4).

3.1 Pipeline

Our proposed pipeline, illustrated in Figure 1, is designed to facilitate experimentation through modular and composable functionality, consisting of four components: data transformation, parameter-efficient fine-tuning, model inference, and post-processing. These modules are implemented as Python scripts, on top of the `transformers` (Wolf et al., 2020) and `torchtune` (torchtune maintainers and contributors, 2024) libraries, and can be configured using command-line arguments.

Initially, articles undergo a **data transformation** phase comprising optional preprocessing (Section 4.1), extractive summarization (Section 4.2), and abstract segmentation (Section 4.3). We apply an identical transformation procedure to each of the three splits from the `eLife` and `PLOS` datasets. The resulting transformed texts are then stored as a separate column within a newly derived dataset, alongside the original "article" and "summary" columns. This derived dataset serves as input for all subsequent stages of the pipeline.

The **model inference** module uses the Llama Instruct model, optionally combined with a LoRA adapter (Hu et al., 2021) that was **fine-tuned** on the transformed text to generate abstractive summaries. During inference, multiple decoding strategies are available: greedy decoding, beam search, and DoLa (Chuang et al., 2024).

Finally, the **post-processing** module can be used to refine further the pipeline's output, which can be the abstractive summary or the text resulting from the data transformation stage.

3.2 Fine-tuning

The Llama model was fine-tuned using LoRA (Hu et al., 2021) for 2 epochs, training separate models for the `PLOS` and `eLife` datasets, with varying batch sizes depending on the GPU and input length. When fine-tuning on full articles on an A40 GPU, a batch size of 2 was used for the `PLOS` dataset and 1 for the `eLife` dataset. The model employed `bf16` precision, and activation checkpointing, activation offloading, and `torch.compile` were used to reduce VRAM usage.

LoRA was applied to the query, value, output projection layers within the attention layers, as well as the MLP layers, with a rank of 8, α of 16, and dropout set to 0.0. The model was optimized using fused AdamW (Loshchilov and Hutter, 2019), with a learning rate of 3e-4 and weight decay of 0.01. A cosine learning rate scheduler with 100 warmup steps was used.

The random seed was set to 4 for reproducibility, and prompts from Table 4 were used to instruct the model.

3.3 Abstractive summarization

We add the LoRA adapters trained on full-text articles to the base Llama instruct model to generate the abstractive summaries. The model instructions follow the `system`, `user`, and `assistant` structure defined by the Chat Markup Language. Furthermore, the `system` messages, summarized in Table 4, include specific target grade-level drawing on the instruction-based readability control outlined by Ribeiro et al. (2023).

To decode the output tokens, we apply Decoding by Contrasting Layers (DoLa) (Chuang et al., 2024) on the lower layers, 0, 2, and 20, using a repetition penalty of 1.2. Compared to beam search and greedy decoding, we found DoLa to provide the best balance between *readability* and *factuality*.

Model inference is performed on a single NVIDIA A40 GPU with a batch size of 1, using the `EOS` token for padding, which takes an average runtime of 62 minutes on the `test` split. Furthermore, we limit the maximum number of tokens generated to 384. We selected this value based on the median summary lengths of the training splits and empirical evaluation comparing output lengths of 256 and 512 tokens (see Figure 4). Furthermore, each submission file, `plos.txt` and `elife.txt`, is created using adapter weights tuned to the respective dataset. Except for the `system` message version, all inference parameters remain constant across runs.

3.4 Evaluation

For experimental validation, we train models on the `train` split of the data and evaluate them on the `validation` split using a pipeline made available by the shared task organizers.[2] Summaries are assessed across 11 automated metrics falling into one of three criteria: *relevance*, *readability*, and *factuality*. To compare results, we adopt the ranking approach used in the previous iteration of BioLaySumm (Goldsack et al., 2024). Specifically, we apply min-max normalization to each metric and average the scores within each criterion before calculating an overall average across all criteria. Our model selection is based on achieving the highest average score from this methodology. The metrics are categorized as follows:

Relevance ROUGE (Lin, 2004), BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and BERTScore (Zhang et al., 2020).

Readability Flesch-Kincaid Grade Level (Kincaid et al., 1975), Dale-Chall Readability Score (Dale and Chall, 1948), CLI (Coleman and Liau, 1975), and LENS (Maddela et al., 2023).

Factuality AlignScore (Zha et al., 2023) and SummaC (Laban et al., 2022).

4 Results and Analysis

In this section, we present our experimental setup and findings obtained through our end-to-end pipeline. Table 1 summarizes the results of these experiments.

4.1 Preprocessing

We replicate the preprocessing approach from Modi and Karthikeyan (2024) to remove content within parentheses, braces, and brackets. Additionally, we apply a number-aware regular expression to collapse additional spacing around punctuation marks and other special characters. In Table 1, we denote experiments that utilized preprocessed inputs with a "`pre`" suffix. Our findings indicate that preprocessing leads to improved *relevance* scores and a better `FKGL` score, especially when combined with fine-tuning. However, these improvements are nullified by lower `LENS` and `SummaC` scores. We hypothesize that removing parentheticals from the input prevents the model from including chunk cues in the output, thereby reducing lexical overlap and potentially lowering entailment scores.

4.2 Extractive summarization

Our extractive summarization method follows from You et al. (2024), using TextRank (Mihalcea and Tarau, 2004) and embedding-based similarity matching. For the latter, we experiment with five pre-trained language embedding models explicitly built for processing biomedical text data, namely: `BioBERT` (Lee et al., 2019), `MedEmbed` (Balachandran, 2024), `PubMedBERT` (Gu et al., 2021), `PubMedBERT-MS-MARCO` (Deka et al., 2022), and `Medical-MiniLM-L6`.[3] Sentence embeddings created using these models are used to measure semantic similarity between them. We also test different embedding models using k-values of 20, 30,

[2] https://github.com/gowitheflow-1998/BioLaySumm2025

[3] https://huggingface.co/Manal0809/medical-term-similarity

Input	PEFT	k	Relevance				Readability				Factuality	
			ROUGE	BLEU	METEOR	BertS	FKGL	DCRS	CLI	LENS	AlignS	SummaC
Ext	×	10	0.323	4.561	0.242	0.833	10.154	7.917	11.342	71.258	0.532	0.528
	×	20	0.338	5.266	0.256	0.834	10.408	7.965	11.574	68.262	0.532	0.527
	×	30	0.342	5.530	0.261	0.834	10.868	8.040	11.938	66.199	0.526	0.512
	✓	10	0.366	7.253	0.272	0.854	9.334	7.529	10.075	77.782	0.615	0.622
	✓	20	0.373	7.730	0.278	0.856	9.037	**7.523**	9.901	78.760	0.626	0.637
	✓	30	0.373	7.650	0.277	0.856	9.062	7.526	9.954	79.118	0.633	0.640
	✓	40	0.379	8.421	0.285	0.857	9.004	7.534	10.008	78.472	0.643	0.645
Ext_{pre}	×	10	0.328	4.767	0.247	0.834	10.185	10.589	11.316	71.154	0.533	0.529
	×	20	0.337	5.181	0.259	0.834	10.348	10.739	11.501	68.144	0.516	0.517
	×	30	0.341	5.386	0.261	0.835	10.640	10.927	11.770	67.053	0.531	0.513
Abs +Ext	✓	10	0.379	8.279	0.292	0.855	8.924	10.304	9.966	78.128	0.634	0.610
	✓	20	0.380	8.332	0.294	0.856	8.999	10.261	10.033	77.653	0.635	0.614
	✓	30	0.380	8.373	0.293	0.855	8.829	10.226	9.950	76.940	**0.648**	0.614
	✓	40	0.382	8.651	**0.297**	0.855	8.956	10.232	9.934	76.674	0.646	0.608
Abs $+\text{Ext}_{(\text{abs})}$	✓	10	0.356	7.462	0.278	0.848	8.885	10.171	9.728	76.015	0.594	0.604
	✓	20	0.365	7.845	0.282	0.853	8.869	10.326	9.850	77.129	0.637	0.637
	✓	30	0.372	8.109	0.284	0.854	9.020	10.376	9.975	78.025	0.643	0.643
	✓	40	0.372	8.200	0.289	0.852	8.857	10.283	9.847	75.797	0.641	0.614
Abs	✓	–	0.369	7.532	0.277	0.854	8.783	10.278	**9.803**	**79.448**	0.634	0.663
Abs_{pre}	✓	–	0.373	8.126	0.289	0.853	**8.733**	10.250	9.809	77.527	0.637	0.599
Full	✓	–	**0.385**	**8.694**	0.289	**0.859**	9.308	7.674	10.143	78.670	0.643	**0.663**
	×	–	0.344	5.766	0.259	0.840	12.483	8.450	12.896	67.947	0.600	0.483
$\text{Full}_{\text{post}}$	✓	–	0.384	8.523	0.287	**0.859**	9.329	10.455	10.153	79.206	0.644	0.662

Table 1: Performance of our abstractive summarization experiments on the eLife validation split. We use PEFT to denote models fine-tuned with LoRA and k to represent the extractive summary length. Data inputs are: **(Ext)** extractive summary, **(Ext_{pre})** preprocessed extractive summary, **(Abs+Ext)** abstract concatenated with extractive summary, **($\text{Abs+Ext}_{(\text{abs})}$)** abstract concatenated with extractive summary that excluded the abstract during extraction, **(Abs)** abstract only, **(Abs_{pre})** preprocessed abstract, **(Full)** entire article, and **($\text{Full}_{\text{post}}$)** entire article, with post-processing applied to the generated summary.

and 40 for summary length. The results indicate a consistent preference for the BioBERT embedding model, regardless of the number of sentences selected. As shown in Figure 3, the overall evaluation score correlates positively with the summary length.

4.3 Training data

We fine-tuned the base instruct model at different levels of input granularity and transformations.

Extractive summary In these experiments, we use the summaries extracted via BioBERT embeddings as the only input. Our results indicate that performance generally improves with more context, although this leads to longer training times. We found that the model fine-tuned on extracted summaries with $k = 40$ is comparable to our best model while requiring less training time.

Abstract-only In this setting, the model is trained solely on the abstract, which is the first paragraph of the input article and serves as a condensed, high-level overview of the study. Even without additional context, the model demonstrated solid performance in terms of readability and factual accuracy. This combination offered the best balance between summarization quality and computational efficiency (see Appendix C).

Abstract and extractive summary We concatenate abstracts with extractive summaries to enrich the input, aiming to provide the model with additional context to improve the factual accuracy and clarity of the generated summaries. We explore two configurations: in Abs+Ext, the abstract is concatenated with an extractive summary generated from the full article, whereas in Abs+Ext(abs), we first remove the abstract from the article before producing the extractive summary. Our evaluation indi-

Decoding	**Runtime**	**Relevance**				**Readability**				**Factuality**	
		ROUGE	BLEU	METEOR	BertS	FKGL	DCRS	CLI	LENS	AlignS	SummaC
DoLa	02:35:41	**0.39**	**9.21**	**0.30**	**0.86**	**9.16**	10.39	**10.10**	77.82	**0.67**	**0.65**
Greedy	02:17:50	**0.39**	9.13	0.30	**0.86**	9.23	**10.38**	10.16	78.19	0.66	0.64
Beam search	07:32:55	0.37	6.56	0.29	0.85	11.31	10.39	10.55	**79.61**	0.55	0.49

Table 2: Runtime and evaluation comparison of the three decoding strategies implemented in our pipeline.

cates that repeating key information (as evidenced by comparing `Ext`, `Abs+Ext`, and `Abs+Ext(abs)`) yields improved *relevance* scores; however, we observe a decline in both *readability* and *factuality*. We hypothesize that the concatenation disrupts the logical ordering of information, which is crucial for these criteria.

Full-text The model is trained on the entire article without any data transformation. This setting showed the best performance, possibly due to having more context, and was our model of choice. Our final submission was trained both on the train split and the validation split. The models were trained on `eLife` for 2 epochs and on `PLOS` for 1.4 epochs.

4.4 Decoding strategies

We investigate the effect of three decoding strategies on our evaluation criteria: greedy decoding, beam search, and DoLa (Chuang et al., 2024). As demonstrated in Table 2, beam search performed poorly, showing significantly lower factuality and relevance scores while also requiring additional hours for inference. Summaries generated using DoLa and greedy decoding had comparable performance and runtimes, with the former achieving the best scores in eight out of eleven metrics. Notably, contrastive decoding yielded the highest factuality results.

4.5 Post-processing

In these experiments, we applied the same text processing method detailed in Section 4.1. Additionally, we removed incomplete sentences arising from the decoding limit on the maximum output token length. Specifically, we identified summaries that did not end with a period and discarded all tokens that appeared after the final complete sentence. Surprisingly, this post-processing step resulted in decreased performance across seven of eleven evaluation metrics, including three *readability* scores, despite the intuitive assumption that truncated sentences negatively affect summary quality.

5 Conclusion

In this study, we presented an end-to-end pipeline for generating lay summaries of biomedical articles. Our approach achieved the highest overall rank in subtask 1.1 of BioLaySumm 2025. Our method balances readability and factuality by employing instruction-based readability control and contrastive decoding (Chuang et al., 2024). In particular, we include the Flesch-Kincaid grade-level target in the system message to improve readability, and control over the LoRA weights enabled the application of contrastive decoding for improved factual accuracy.

We posit that investigating more advanced instruction strategies, such as self-reflection and synthesized chain-of-thought (CoT), represents a promising direction for future research. These strategies could incorporate factual claims and lay terminology to improve the model's relevance and factual accuracy. Furthermore, adding a reinforcement learning component, such as Direct Preference Optimization (Rafailov et al., 2023), to our pipeline could help select outputs that better align with the evaluation framework of this task.

Acknowledgments

We thank the organizers of the shared task for their guidance and coordination. We also thank the anonymous reviewers for their valuable feedback and suggestions, which helped improve the quality of this paper.

Parts of this work were completed on Hyak, the University of Washington's high-performance computing cluster. This resource was funded by the Student Technology Fee.

References

Abhinand Balachandran. 2024. Medembed: Medical-focused embedding models.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Pro-*

ceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. 2024. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*.

Meri Coleman and Ta Lin Liau. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60(2):283.

Edgar Dale and Jeanne S. Chall. 1948. A formula for predicting readability: Instructions. *Educational Research Bulletin*, 27(2):37–54.

Pritam Deka, Anna Jurek-Loughrey, and P Deepak. 2022. Improved methods to aid unsupervised evidence-based fact checking for online health news. *Journal of Data Intelligence*, 3(4):474–504.

Tomas Goldsack, Zheheng Luo, Qianqian Xie, Carolina Scarton, Matthew Shardlow, Sophia Ananiadou, and Chenghua Lin. 2023. Overview of the biolaysumm 2023 shared task on lay summarization of biomedical research articles. In *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 468–477, Toronto, Canada. Association for Computational Linguistics.

Tomas Goldsack, Carolina Scarton, Matthew Shardlow, and Chenghua Lin. 2024. Overview of the BioLaySumm 2024 shared task on the lay summarization of biomedical research articles. In *Proceedings of the 23rd Workshop on Biomedical Natural Language Processing*, pages 122–131, Bangkok, Thailand. Association for Computational Linguistics.

Tomas Goldsack, Zhihao Zhang, Chenghua Lin, and Carolina Scarton. 2022. Making science simple: Corpora for the lay summarisation of scientific literature. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10589–10604, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Trans. Comput. Healthcare*, 3(1).

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

J. Peter Kincaid, Jr. Fishburne, Robert P., Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count, and flesch reading ease formula) for navy enlisted personnel. Research Branch Report 8-75, Institute for Simulation and Training, University of Central Florida.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

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.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *Preprint*, arXiv:1711.05101.

Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. 2022a. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6):bbac409.

Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2022b. Readability controllable biomedical document summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4667–4680, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mounica Maddela, Yao Dou, David Heineman, and Wei Xu. 2023. LENS: A learnable evaluation metric for text simplification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16383–16408, Toronto, Canada. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Satyam Modi and T Karthikeyan. 2024. Eulerian at BioLaySumm: Preprocessing over abstract is all you need. In *Proceedings of the 23rd Workshop on Biomedical Natural Language Processing*, pages 826–830, Bangkok, Thailand. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*.

Leonardo F. R. Ribeiro, Mohit Bansal, and Markus Dreyer. 2023. Generating summaries with controllable readability levels. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11669–11687, Singapore. Association for Computational Linguistics.

torchtune maintainers and contributors. 2024. torchtune: Pytorch's finetuning library.

Oisín Turbitt, Robert Bevan, and Mouhamad Aboshokor. 2023. MDC at BioLaySumm task 1: Evaluating GPT models for biomedical lay summarization. In *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 611–619, Toronto, Canada. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chenghao Xiao, Kun Zhao, Xiao Wang, Siwei Wu, Sixing Yan, Tomas Goldsack, Sophia Ananiadou, Noura Al Moubayed, Liang Zhan, William CHEUNG, and Chenghua Lin. 2025. Overview of the biolaysumm 2025 shared task on lay summarization of biomedical research articles and radiology reports. In *The 24nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, Vienna, Austria. Association for Computational Linguistics.

Zhiwen You, Shruthan Radhakrishna, Shufan Ming, and Halil Kilicoglu. 2024. UIUC_BioNLP at BioLaySumm: An extract-then-summarize approach augmented with Wikipedia knowledge for biomedical lay summarization. In *Proceedings of the 23rd Workshop on Biomedical Natural Language Processing*, pages 132–143, Bangkok, Thailand. Association for Computational Linguistics.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, Toronto, Canada. Association for Computational Linguistics.

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

A Instruction Messages

Table 4 details the system messages used to instruct the model in generating the summaries. We found that including the target domain and grade level contributed to better *readability* scores. The `eLife` summaries were created with version 1, while the `PLOS` summaries were produced with version 2.

B Dataset Statistics

The Public Library of Science (`PLOS`) is a non-profit, open-access publisher launched in 2000 with the goal of providing free access to full-text scientific articles. It currently publishes 14 academic journals in a range of fields such as biology, medicine, and computational biology. `eLife` is likewise a non-profit, peer-reviewed, open-access publisher for articles in the biomedical and life science domains established in 2012. Articles in the two datasets cover various topics and specialties within the biomedical domain. We report length statistics for the `PLOS` and `eLife` datasets in Table 3.

Dataset	# Docs	Doc	Summary	
		# words	# words	# sents
PLOS	27,525	5,366.7	175.6	7.8
eLife	4,828	7,806.1	347.6	15.7

Table 3: Average word and sentence counts for each dataset. Adapted from Goldsack et al. (2022).

C Computational Efficiency

Although using full article texts as model input yielded the highest performance, this approach is significantly more resource-intensive than relying only on extractive summaries or abstracts. This difference is clearly illustrated in Figure 2, which compares average inference runtimes on the `eLife` and `PLOS` datasets. Specifically, inference on full-text inputs required over 30 times the runtime of

#	Message
1	You are a specialist medical communicator responsible for translating biomedical articles into a clear, accurate 1020 sentence summary for non-experts. The summary should be at a FleschKincaid grade level of 1014 and explain any technical terms.
2	You are a specialist medical communicator responsible for translating biomedical articles into a clear, accurate 10 to 20 sentence summary for non-experts. The summary should have a FleschKincaid grade level of 10 to 14, explaining any technical terms in simple language. Ensure factual accuracy by using terminology from the source article, and omit all in-text citations.

Table 4: The two system messages used to generate the abstractive summaries. Generative language models were used to refine the messages.

abstract-only inputs, while providing only a 14.86% improvement in the overall average score.

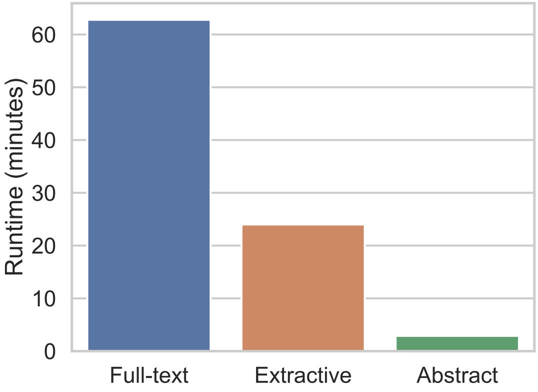

Figure 2: Inference runtime comparison of the summarization model based on different input types: full-text articles, extractive summaries, and abstracts.

D Training Challenges and Workarounds

There is a peculiarity that we would like to mention about our training setup. While University of Washington's high-performance computing cluster Hyak offers powerful hardware, GPU jobs are prone to preemption and can run at most for 8-9 hours before being requeued. However, a full epoch exceeded that limit, sometimes taking over 24 hours. At the time of our experiment, `torchtune` did not support mid-epoch checkpointing, so we had to split the data into smaller sections to ensure each partial epoch could finish within the time limit. The actual split sizes were smaller to accommodate preemption and were dynamically adjusted along with the batch size based on the number and model of the GPU in use. The total number of epochs was set to $\left\lceil \frac{1}{\text{split ratio}} \right\rceil \times$ (number of epochs) to have `torchtune` save the training state between partial epochs. Training processes were killed and restarted after each partial epoch to force `torchtune` to reload the training configuration file with updated data splits. This part is specific to Hyak, and the code will only be included in the `release/class` branch and excluded from the `main` branch and future releases.

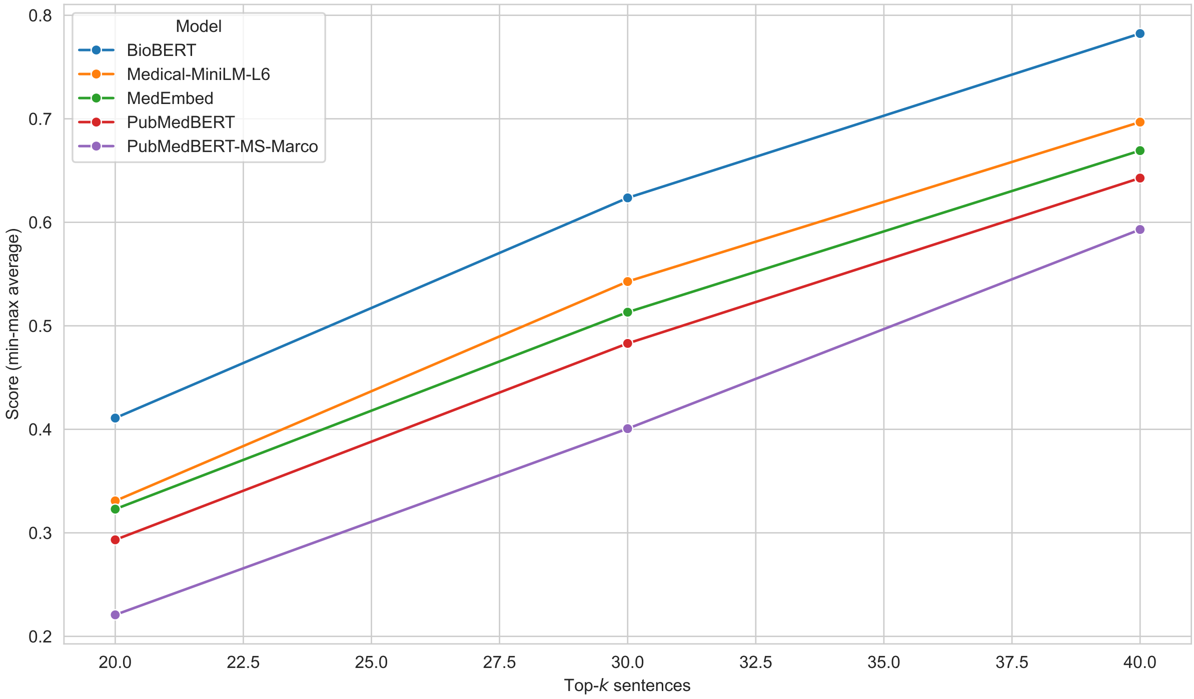

Figure 3: Relative performance of extractive methods on the `eLife` training data, categorized by embedding model and the top-k sentences extracted using TextRank (Mihalcea and Tarau, 2004).

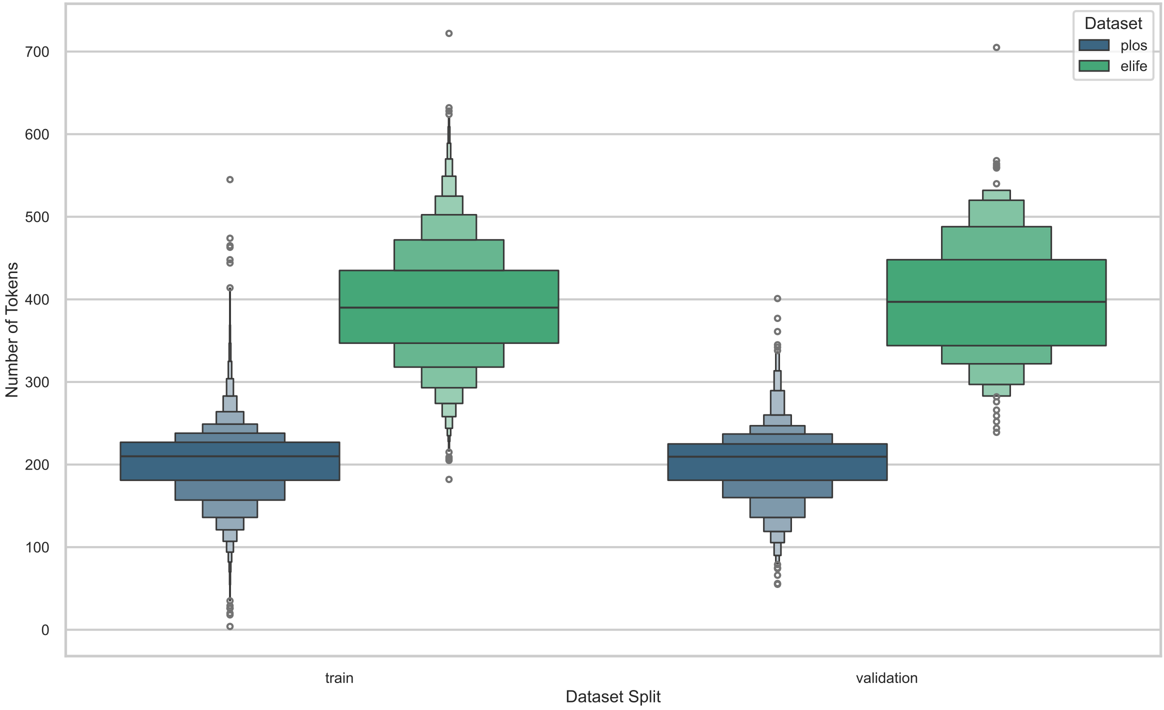

Figure 4: Distribution of token counts across training and validation splits for the `PLOS` and `eLife` datasets.