CocoSciSum: A Scientific Summarization Toolkit with Compositional Controllability

Yixi Ding[†] **Yanxia Qin**[†*] **Qian Liu**[‡] **Min-Yen Kan**[†]
[†]School of Computing, National University of Singapore
[‡] Sea AI Lab
`{yixi.d, qinyx, kanmy}@comp.nus.edu.sg, liuqian@sea.com`

Abstract

We present a novel toolkit for controlled summarization of scientific documents, designed for the specific needs of the scientific community. Our system generates summaries based on user preferences, adjusting key attributes specifically of length and keyword inclusion. A distinguishing feature is its ability to manage multiple attributes concurrently, demonstrating **Co**mpositional **Co**ntrollability for **Sci**entific **Sum**marization (*CocoSciSum*). Benchmarked against the strong Flan-T5 baseline, *CocoSciSum* exhibits superior performance on both the quality of summaries generated and the control over single and multiple attributes. Moreover, *CocoSciSum* is a user-centric toolkit, supporting user preferences expressed in natural language instructions, and accommodating diverse input document formats. *CocoSciSum* is available on GitHub[1] with an introduction video[2].

1 Introduction

Scientific summarization refers to the process of distilling scientific documents such as research papers into shorter versions that capture the key information. It has become increasingly important as it can facilitate quick search results filtering, as seen in Semantic Scholar's TL;DR (Too Long, Didn't Read) feature. However, we argue that controlled summarization for scientific documents, which generates user-customized summaries over various control attributes, can further enhance personalized result filtering and paper comprehension. For example, a reader primarily interested in a paper's relationship to a particular term — say, '*cloze task*' — can instruct the summarization system to focus on the term when generating the summary. Accordingly, we introduce the first toolkit designed for controlled summarization in the scientific domain. It consistently generates high-quality summaries, as validated by various automatic evaluation metrics and human annotators. Aside from improved performance, our system, *CocoSciSum*, makes two key contributions to the summarization landscape.

Contribution 1: Compositional Controllability. We seek a unified solution for all summarization functions, aiming to reduce memory usage and streamline deployment. *CocoSciSum* provides three modes of summarization: (1) vanilla summarization, (2) single-attribute controlled summarization (managing either length or keyword inclusion), and (3) compositionally-controlled summarization, which jointly regulates both length and keywords. It employs instruction-tuned PLMs; specifically adopting FLAN-T5 (Chung et al., 2022) as the backbone model. Figure 1 shows the framework of the summarization model.

To achieve such compositional goals, past summarization systems all have resorted to separate models to control each attribute of interest (Takase and Okazaki, 2019; Saito et al., 2020; Liu et al., 2022; Zheng et al., 2020; Narayan et al., 2021). However, this presents a significant challenge to the pre-trained language model (PLM) finetuning paradigm, due to the scarcity of training data. The necessary creation of training data incurs significant annotation costs, given the complexity of annotating multiple reference summaries for a single document, each conforming to individual or multiple attribute constraints. The disparity between available and desired training data compels us to seek an alternative, as reference summaries for multiple controls are not available and prohibitively expensive to annotate. Instead, we break down the annotation task into simpler sub-tasks, synthesizing summaries constrained for each attribute separately. We hypothesize that by initially training the model on simplified relevant tasks, it can subsequently generalize. That is, we finetune the PLM using

*Corresponding author
[1]`https://github.com/WING-NUS/SciAssist/tree/CocoSciSum`
[2]`https://youtu.be/YC1YDeEjAbQ`

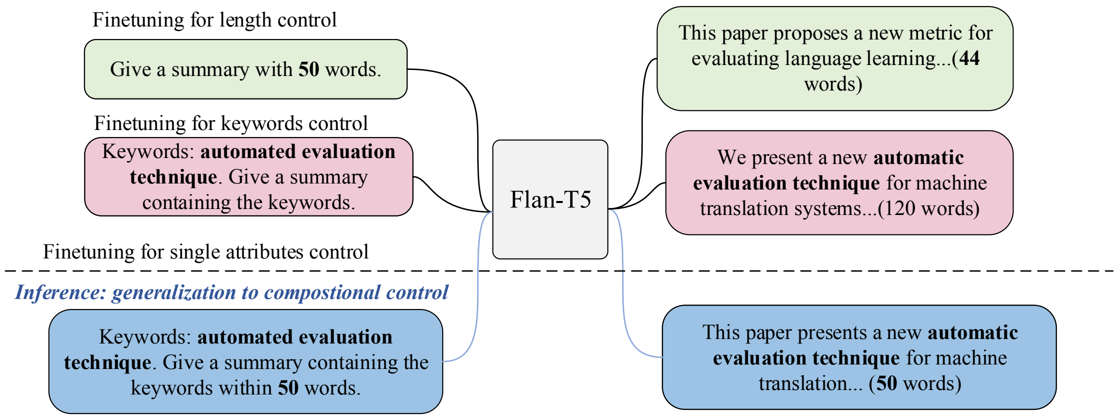

Finetuning for length control
Give a summary with **50** words.
Finetuning for keywords control
Keywords: **automated evaluation technique**. Give a summary containing the keywords.
Finetuning for single attributes control
Flan-T5
This paper proposes a new metric for evaluating language learning...(**44** words)
We present a new **automatic evaluation technique** for machine translation systems...(120 words)
Inference: generalization to compostional control
Keywords: **automated evaluation technique**. Give a summary containing the keywords within **50** words.
This paper presents a new **automatic evaluation technique** for machine translation... (**50** words)

Figure 1: The summarization model with compositional controllability used in *CocoSciSum*. The FLAN-T5 backbone model is separately finetuned for length and keyword controls, and generalizes to the composition task. Control attribute mentions are highlighted in bold.

single-attribute constrained data, expecting its capability to solve composite summarization tasks.

Contribution 2: User-friendly. We build *CocoSciSum* to allow easy uptake for those simply curious as well as serious practitioners. We offer a demonstration page[3] for *CocoSciSum*'s immediate utilization, and an easy-to-install Python package along with comprehensive development documentation[4] for practitioners.

CocoSciSum itself is also user-friendly: it interprets user preferences specified in natural language instructions and supports multiple input document formats. In contrast to previous work that employed opaque vectors (i.e., unreadable to humans) for summarization control, *CocoSciSum* adopts user-friendly natural language to specify instructions, such as "*Give a summary within 50 words and containing the phrase 'automated evaluation technique':* " (Figure 1). Finally, considering the prevalence of portable document format (PDF) for scientific documents (which are not directly machine-readable), we incorporate a specialized PDF text extraction module. Our toolkit also supports other document formats like plain text and structured JSON for wider practical usage.

2 Architecture

We show the architecture of *CocoSciSum* in Figure 2. It consists of training and inference phases.

[3]https://huggingface.co/spaces/wing-nus/SciAssist

[4]https://wing-sciassist.readthedocs.io/en/latest/

2.1 Training

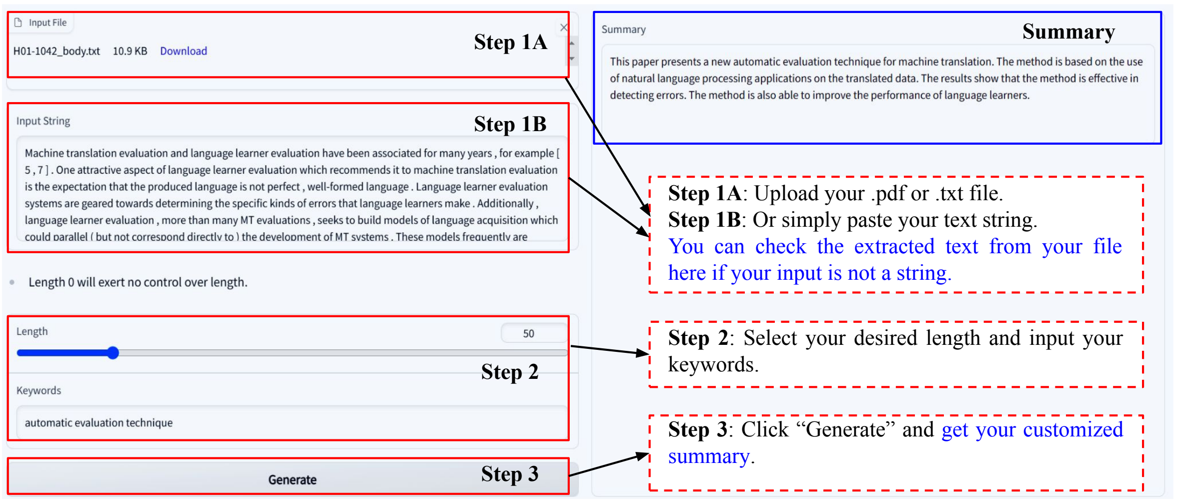
Implementation Framework. PyTorch Lightning[5] is a flexible high-level interface for PyTorch, abstracting complex training logic for rapid prototyping. We adopt Lightning for *CocoSciSum*'s workflow. The workflow of *CocoSciSum* starts with a `LightningDataModule` to download and split datasets and then assemble batched vector data. Subsequently, a `LightningModule` that encapsulates the model and optimizer is used. Simultaneously, a `Trainer` coordinates the above components to automate training.

Configuration. Hydra (Yadan, 2019) is an open-source Python framework that simplifies the management of configurations in applications. In particular, Hydra allows dynamically creating a hierarchical configuration by composition from multiple sources, and overriding it through both the original configuration files and the command line, facilitating specifying hyper-parameters and experimental settings. We follow a deep learning project template[6] for designing Hydra in *CocoSciSum*.

Utility. The Utility module centralizes all data processing procedures, inclusive of tokenizing text-label mixed strings (`tokenize_and_align_labels`), generating the text sequence from a matrix of probabilities over a vocabulary (`prob2text`) and customizing batch collation (`batch_collator`). All data utility functions are encapsulated into a `DataUtils` class,

[5]https://lightning.ai/pytorch-lightning

[6]https://github.com/ashleve/lightning-hydra-template



Input File

H01-1042_body.txt 10.9 KB Download

Step 1A

Input String

Step 1B

Machine translation evaluation and language learner evaluation have been associated for many years , for example [5 , 7] . One attractive aspect of language learner evaluation which recommends it to machine translation evaluation is the expectation that the produced language is not perfect , well-formed language . Language learner evaluation systems are geared towards determining the specific kinds of errors that language learners make . Additionally , language learner evaluation , more than many MT evaluations , seeks to build models of language acquisition which could parallel (but not correspond directly to) the development of MT systems . These models frequently are

Length 0 will exert no control over length.

Length 50

Step 2

Keywords

automatic evaluation technique

Generate

Step 3

Summary

Summary

This paper presents a new automatic evaluation technique for machine translation. The method is based on the use of natural language processing applications on the translated data. The results show that the method is effective in detecting errors. The method is also able to improve the performance of language learners.

Step 1A: Upload your .pdf or .txt file.
Step 1B: Or simply paste your text string.
You can check the extracted text from your file here if your input is not a string.

Step 2: Select your desired length and input your keywords.

Step 3: Click "Generate" and get your customized summary.

Figure 2: *CocoSciSum*'s interface and usage for controlled summarization. It accepts multiple formats for input documents, including PDF, text string, and text files. User preferences on length, keyword inclusion or both can be specified in the control section. The customized summary will be presented in the output summary zone with a click on the Generate button.

which facilitates the use of different datasets in various formats, and enables the seamless application of a single dataset to multiple models.

Adding New Datasets. Integrating additional datasets for use in *CocoSciSum* is simple. We create a new `DataModule` class inheriting from `LightningDataModule`, implementing several key functions to download the data (`prepare_data`), split it into training and validation sets (`setup`), prepare batched vector data (`train_dataloader`). A Hydra file associated with the dataset can be used for easy configuration.

Adding New Models. Users can also add a new model by implementing a model class inherited from `LightningModule`. Such a class can customize the training logic, defined in `training_step` function. A Hydra configuration file can be used to define model settings and hyper-parameters.

2.2 Inference

In addition to supporting various input document formats *CocoSciSum* is designed to be easily deployable on Linux, Windows, and MacOS, further enhancing its usability. Due to the absence of a universal toolkit for all three OS, we use different text extraction toolkits for PDF files on different platforms. Moreover, users can access *CocoSciSum* through its Python package or a demo page 2.3. Accordingly, the system offers both direct summaries and text-summary pairs for different scenarios. On the demo page, we fix the control prompts to be the instructions used for finetuning the model in Section 3, for generating summaries in high quality. Meanwhile, the provided model in the Python package, sourced from Section 3, supports more flexible natural language prompts.

2.3 Demonstration

To illustrate the functionality of *CocoSciSum*, we offer a demo page featuring a user-friendly interface, as shown in Figure 2. The interface is divided into two sections. The left side serves as the user input area, which accepts input documents, user-specified control attributes, and the generation action. The right side shows the system-generated summary.

Users can customize summary length using a slider with increments of 50 or input specific keywords if necessary. The system also displays the text extracted from the input document in a text box, offering users a chance to verify the content, thereby enhancing overall usability.

Our demo page is hosted on the Huggingface platform, which provides various hardware deployment options. We have utilized the free CPU-basic version for our demo, equipped with 2 CPUs and a total of 16 GB RAM. On average, a summary is generated in less than 30 seconds, indicating that *CocoSciSum* can be deployed effortlessly, even on hardware with limited resources. This is mainly due to the small-sized model, which allows for satisfactory performance with a reasonable time cost.

3 Compositionally Controlled Summarization

We now introduce the compositionally-controlled summarization model used in *CocoSciSum*. The model is designed to support all summarization functions, including standard, single-attribute-controlled and multi-attribute-controlled summarization. We first introduce the backbone summarization model and then detail the instruction tuning for compositional controllability.

3.1 Backbone Model

Previous controlled summarization work (He et al., 2022; Zhang et al., 2022) has successfully leveraged the PLM finetuning paradigm, which mostly utilizes text generation PLMs such as BART (Lewis et al., 2019) and T5 (Raffel et al., 2020). However, the lack of labeled data and high annotation costs for the compositionally-controlled summarization motivates us to seek another solution. Drawing inspiration from instruction tuning (Wei et al., 2021; Chung et al., 2022), we utilize an instruction-tuned PLM to harness its zero-shot performance on unseen tasks. Specifically, we instruction-tune a PLM on two single-attribute-controlled summarization tasks, anticipating its zero-shot capability on multi-attribute-controlled summarization.

We employ the FLAN-T5 model, which is based on the T5 model and instruction tuned on 1,836 tasks. As described in the prior work (Chung et al., 2022), this wide-ranging finetuning results in substantially improved zero-shot performance on unseen tasks. In addition, among these instruction tuning tasks, 1,554 of them are natural instruction tasks, further enhancing its comprehension of natural language instructions. We capitalize on this and use natural language to prompt FLAN-T5 for user-desired summaries, such as "`Summarize the text with 50 words:`".

3.2 Finetuning

To further finetuning FLAN-T5 on single-attribute controlled summarization tasks, we synthesize their finetuning data from a generic summarization dataset in the scientific domain.

Length Control. Compared to coarse-grained length control in previous work such as "long" and "short", we aim to generate summaries with exact length control. We propose to use the following prompt: "`Give a summary of the following text, which has less than` n `words:`" to control the length of the system-generated summary, where n is the number of words in the reference summary. To avoid the sparseness of exact n in finetuning, we round it up to the nearest bin of 50, e.g., 167→200.

Keyword Control. The purpose of keyword control is to generate summaries relevant to the keywords of interest. We use the following instruction to guide the model: "`Keywords:` $[k_1, k_2, \cdots]$`. Give a summary of the following text based on these keywords:` ", where k refers to a keyword. Here, a keyword can be a single word or a phrase, used to represent a scientific term. Given a generic summarization data instance including a document–summary pair, we lack the keywords that appeared in the summary. Thus we propose to extract keywords from the reference summary. Again, we take advantage of FLAN-T5 for its zero-shot performance in information extraction tasks and prompt FLAN-T5 to generate keywords in the reference summaries. In particular, we instruct a `FLAN-T5-XL`[7] model with the following prompt "`What keywords does this scientific summary include? [Ref Summary] Keywords:`".

Compositional Control. Given the limitations in available computing resources, we adopt a small-sized PLM, `FLAN-T5-base` with 230 million parameters, to finetune with length-controlled and keyword-controlled data expecting compositional controllability. Due to its limited model capacity, the FLAN-T5-base model is unlikely to keep its first gained ability when finetuned for the second ability (Fu et al., 2023). We validated this phenomenon also occurs in our experiments , which guides us for a different finetuning strategy for compositional control. Inspired by (Cachola et al., 2020), we shuffle two finetuning datasets and then train the model with the shuffled dataset to generalize on both attributes.

4 Experiments

4.1 Experimental Settings

Datasets. We construct datasets using two scientific domain summarization datasets: *MuP* (Cohan et al., 2022) and *SciSumm* (Chandrasekaran et al.,

[7]The FLAN-T5-XL model is used due to better performance.

Model	R-1	R-2	R-L	R-LSum	BScore	MAD↓	PCC↑	SR↑	FACT↑
T5	31.10	5.67	17.98	27.99	0.091	–	–	–	–
FLAN-T5	33.59	6.40	19.34	29.60	0.128	–	–	–	–
Ours	33.97	6.54	*19.48*	30.00	*0.131*	–	–	–	–
FLAN-T5$_{Len}$	33.33	6.49	19.43	29.44	0.129	1.41	-0.01	–	–
Ours$_{Len}$	*35.24*	6.80	**19.92**	*31.12*	**0.133**	**0.36**	**0.85**	–	–
CTRLsum$_{Kw}$	32.00	**7.93**	18.06	28.62	0.090	–	–	**0.61**	–
FLAN-T5$_{Kw}$	33.08	6.53	19.41	29.23	0.130	–	–	0.24	0.44
Ours$_{Kw}$	34.43	7.06	18.97	30.34	0.129	–	–	0.57	0.58
T5$_{Len,Kw}$	31.20	5.56	18.07	28.11	0.085	1.46	0.00	0.23	–
Ours$_{Len,Kw}$	**35.35**	*7.16*	19.45	**31.26**	0.130	*0.60*	*0.83*	*0.58*	**0.65**

Table 1: Experimental results. R-* (ROUGE), BScore (BERTScore), MAD and PCC are calculated on *MuP-dev-1k*, SR and FACT are calculated on *KW-test*. The settings denoted with $_{Len}$ and $_{Kw}$ are attribute-controlled inferences, others are vanilla summarization inferences. Optimal results are highlighted in **bold**. *Italicization* indicates values that are the second best.

2019). A training dataset *Coco-train* and a development dataset *Coco-dev* are created for model finetuning (See Section 3.2). Two test sets *MuP-dev-1k* and *KW-test* are constructed for evaluating summarization quality and length controllability, and keyword controllability, respectively. Details and statistics are presented in Appendix A.

Evaluation Metrics. We evaluate the output summaries for both summarization quality and attribute controllability. We employ ROUGE (Lin, 2004) and BERTScore (Zhang et al., 2020) for quality evaluation. For length controllability evaluation, we adopt the Mean of Absolute Deviation (MAD, Liu et al., 2018) and the Pearson Correlation Coefficient (PCC, Liu et al., 2018) between length codes. For evaluating keyword controllability, we use the Success Rate (SR, Fan et al., 2018; He et al., 2022 and the Factual Correctness (FACT, Krishna et al., 2023). More details are in Appendix B.

Baselines. We choose baseline methods including a Seq2Seq model T5 (Raffel et al., 2020), a controlled summarization method CTRLsum (He et al., 2022), and an instruction-tuned model FLAN-T5. The FLAN-T5 baseline is finetuned with the general summarization dataset *MuP*. The T5 models is finetuned with the same attribute-controlled dataset *Coco-train* as our model, CTRLsum is finetuned with keyword-controlled dataset *KW-data*. We have chosen not to include scientific PLMs, such as SciBERT (Beltagy et al., 2019) and Galactica (Taylor et al., 2022) because a fair comparison is currently not feasible (wrong model size, architecture, and inability for instruction tuning).

4.2 Results

We present the experimental results in Table 1.

Summary Quality. Compared to T5 in the vanilla summarization setting, both FLAN-T5 and our model demonstrate improved performances across all quality evaluation metrics (e.g., 33.59 and 33.97 V.S. 31.1 on R-1). This indicates the superiority of FLAN-T5 over T5. However, our system consistently surpasses the strong FLAN-T5 baseline in both standard and controlled settings, with an average performance increase of 1.21 on R-1, underlining its effectiveness. Furthermore, by comparing our model in three controlled inferences with the vanilla setting, we observe rises in ROUGE scores (an average increase of 1.04 on R-1). This implies that control signals further enhance the quality of the generated summaries. Interestingly, this advantage is exclusive to our model and not observed in either T5 or FLAN-T5.

Length Controllability. From Table 1, the low PCC value (-0.01) of the FLAN-T5 baseline indicates that it has no controllability over the length of summaries, suggesting it does not understand unseen instructions of length control. We observe that T5 haves no controllability over length (PCC 0), even trained with length-controlled data. While our model in two length-controlled inferences ($_{Len}$ and $_{Len,Kw}$) achieve high PCC values (≥0.83, out of 1), showing the presence of strong length controllability in *CocoSciSum*. Interestingly, we observe weaker controllability of our model in the compositional control setting $_{Len,Kw}$, compared with single length control. We assume that multiple control

System	Attribute	Summary (# of Words)
FLAN-T5$_{Kw}$	"Cloze task" ×	This paper proposes a new method for pre-training language models. The method is based on a masked language model (LM) pre-training objective. The masked language models are used to predict the original vocabulary id of the masked word. The authors show that the proposed method can achieve state-of-the-art performance on a range of tasks. (54)
Ours	—	This paper proposes a bidirectional language model pre-training method for NLP tasks. The proposed method is based on a masked language model (MLM) pre-training objective. The masked language models randomly mask some of the tokens from the input $\cdots$ The proposed method achieves state-of-the-art performance on a large suite of sentence-level and token-level tasks. (90)
Ours$_{Len}$	50 words ✓	This paper proposes a bidirectional language model pretraining method. The proposed method is based on a masked language model. The proposed method achieves state-of-the-art performance on several NLP tasks. (**29**)
Ours$_{Kw}$	"Cloze task" ✓	We present a new pre-training method for language representations. The method is inspired by the **Cloze task**, which is a task-specific task that requires pre-trained language models to perform well $\cdots$ model is able to achieve state-of-the-art performance on a large set of sentence-level tasks, and outperforms many task-specific baselines. (125)
Ours$_{Len,Kw}$	50 words ✓; "Cloze task" ✓	This paper proposes a bidirectional pre-training method for language representations. The method is inspired by the **Cloze task**. The method is evaluated on a large suite of sentence-level and token-level tasks. (**31**)

Table 2: Example summaries of BERT, generated by different systems. Attribute denotes the control attribute mentions, "50" means to limit the summary within 50 words, and "Cloze task" is the user-preferred keyword. Evidences of controllability are highlighted in bold. ✓ and × indicate successful and failed control, respectively.

signals are difficult to understand for a relatively small-sized PLM with 230 million parameters compared to those with 7 billion or more parameters.

Keyword Controllability. With the success rate, we evaluate whether the model can generate a summary containing user-predefined keywords following the natural language instruction. A success rate of 0.24 for FLAN-T5 indicates it has only limited controllability for keywords. In contrast, our model achieves a higher success rate of 0.57, offering more than double the probability of generating summaries inclusive of keywords, thus demonstrating our model's superior and more reliable keyword controllability. CTRLsum achieves the best controllability over keywords among all methods, however no length controllability as it is controlled by the number of keywords.

Our approach not only aims to include keywords in the summary but also strives to prevent overfitting to the success rate by generating fictional summaries. Compared to the factual correctness of 0.44 of FLAN-T5, our model yields a higher score of 0.58, further reinforcing its superior controllability. When comparing our system in the compositionally-controlled setting $_{Len,Kw}$ with T5, which show no or random controllability (0 for PCC, 0.23 for SR), it is apparent that FLAN-T5 serves as the foundational model for the compositional controllability of our model.

Case Study. We present several system-generated summaries for the BERT (Devlin et al., 2019) paper in Table 2. All summaries are fluent, grammar error-free, and easy to understand, which shows strong summarization abilities of Flan-T5 models again. More importantly, we observe controllability over both length and keywords from our model through the desired length and keyword inclusion. Another interesting observation is that all summaries include the main contribution ("*a new pre-training method for language representations*") as well as the experimental information ("state-of-the-art performance") of the source scientific document.

5 Conclusion

We present the first scientific toolkit, *CocoSciSum*, which summarizes scientific papers by customizing multiple aspects, including fine-grained length control, keyword inclusion, and compositional control of both. The user preferences are specified in natural language instructions rather than human-unreadable vectors. We provide an easy-to-install Python package and a demo page for different uses. *CocoSciSum* supports multiple input formats of scientific papers, plain text, structured JSON, and the most commonly used PDF. We employ FLAN-T5 as the backbone model, finetuning it on single-attribute-controlled tasks, and prove its capability for multi-attribute control.

In future work, we will explore compositional controllability conflicts and user-specified weighting between multiple attributes.

Acknowledgements

We thank the reviewers for their valuable feedback and suggestions. This research is supported by the Singapore Ministry of Education Academic Research Fund Tier 1 (T1 251RES2216, T1 251RES2037).

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. 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 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel Weld. 2020. TLDR: Extreme summarization of scientific documents. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4766–4777, Online. Association for Computational Linguistics.

Muthu Kumar Chandrasekaran, Michihiro Yasunaga, Dragomir Radev, Dayne Freitag, and Min-Yen Kan. 2019. Overview and results: Cl-scisumm shared task 2019.

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

Arman Cohan, Guy Feigenblat, Tirthankar Ghosal, and Michal Shmueli-Scheuer. 2022. Overview of the first shared task on multi perspective scientific document summarization (MuP). In *Proceedings of the Third Workshop on Scholarly Document Processing*, pages 263–267, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning.

Junxian He, Wojciech Kryscinski, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2022. CTRLsum: Towards generic controllable text summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5879–5915, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kalpesh Krishna, Erin Bransom, Bailey Kuehl, Mohit Iyyer, Pradeep Dasigi, Arman Cohan, and Kyle Lo. 2023. Longeval: Guidelines for human evaluation of faithfulness in long-form summarization.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

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.

Yizhu Liu, Qi Jia, and Kenny Zhu. 2022. Length control in abstractive summarization by pretraining information selection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6885–6895, Dublin, Ireland. Association for Computational Linguistics.

Yizhu Liu, Zhiyi Luo, and Kenny Zhu. 2018. Controlling length in abstractive summarization using a convolutional neural network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4110–4119, Brussels, Belgium. Association for Computational Linguistics.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreferencefor scientific knowledge graph construction. In *Proc. Conf. Empirical Methods Natural Language Process. (EMNLP)*.

Shashi Narayan, Yao Zhao, Joshua Maynez, Gonçalo Simões, Vitaly Nikolaev, and Ryan McDonald. 2021. Planning with learned entity prompts for abstractive summarization. *Transactions of the Association for Computational Linguistics*, 9:1475–1492.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, Atsushi Otsuka, Hisako Asano, Junji Tomita, Hiroyuki Shindo, and Yuji Matsumoto. 2020. Length-controllable abstractive summarization by guiding with summary prototype. *CoRR*, abs/2001.07331.

Sho Takase and Naoaki Okazaki. 2019. Positional encoding to control output sequence length. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3999–4004, Minneapolis, Minnesota. Association for Computational Linguistics.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *CoRR*, abs/2109.01652.

Omry Yadan. 2019. Hydra - a framework for elegantly configuring complex applications. Github.

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

Yusen Zhang, Yang Liu, Ziyi Yang, Yuwei Fang, Yulong Chen, Dragomir Radev, Chenguang Zhu, Michael Zeng, and Rui Zhang. 2022. Macsum: Controllable summarization with mixed attributes.

Changmeng Zheng, Yi Cai, Guanjie Zhang, and Qing Li. 2020. Controllable abstractive sentence summarization with guiding entities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5668–5678, Barcelona, Spain (Online). International Committee on Computational Linguistics.

A Dataset Construction and statistics

In this section, we introduce details about the dataset construction.

Training Data Our training data is derived from two generic summarization datasets in the scientific domain: *MuP* (Cohan et al., 2022) and *SciSumm* (Chandrasekaran et al., 2019). We adopt these two datasets as they both provide reference summaries and are large-scale (18.9k and 1k document–summary pairs, respectively). We construct length-controlled labeled data from the training set of *MuP-train* with the method outlined in Section 3.2, denoted as *LEN-data*. The keyword-controlled data comes from *SciSumm*, denoted as *KW-data*. Finally, *LEN-data* and *KW-data* are merged together and then randomly split into a training set *Coco-train* and a development set *Coco-dev* using a 9:1 ratio.

Test sets We design three test sets for different testing objectives. For calculating ROUGE scores and BERTScore, length controllability metrics MAD and PCC, we randomly sampled 1000 data points from MuP's development set, creating our test set (*MuP-dev-1k*).

To ensure high-quality keywords for calculating keyword controllability evaluation metric Success Rate, we choose SciERC (Luan et al., 2018), a dataset with human-annotated scientific entities, as the data source. From SciERC, we select entities classified as "Method" and "Task", each comprising less than 3 words, to simulate user preferences. This results in a set of 229 (document, keyword) pairs from 55 documents, denoted as *KW-test*.

To facilitate the manual evaluation of Factual Correctness, we randomly sample 15 documents from SciERC and 10 documents from the top 30 most cited papers in ACL Anthology [8] as of June 2023. For ACL papers, we utilize zero-shot Flan-T5-XL to extract keywords. Then we obtain the test set(*FACT-test*) containing 48 instances derived from 25 documents.

The statistics of all datasets are presented in Table 3.

Datasets	**#Instances**
MuP-train	18,934
MuP-dev	3,604
SciSumm	915
Coco-train	17,865
Coco-dev	1,984
MuP-dev-1k	1000
KW-test	229
FACT-test	48

Table 3: Statistics of the Training Set

B Evaluation Metrics

For length controllability evaluation, we adopt (1) the Mean of Absolute Deviation (MAD, Liu et al., 2018) of length codes of system-generated summaries and the references, measuring their length distance; and (2) the Pearson Correlation Coefficient (PCC, Liu et al., 2018) between above length codes to access the summary variations as length

[8]https://aclanthology.org/

signals change. For evaluating keyword controllability, we use (1) the Success Rate (SR, Fan et al., 2018; He et al., 2022)[9] to reflect the fraction of requested keywords actually occurring in the output summaries; and (2) the Factual Correctness (FACT, Krishna et al., 2023) to ensure the generated summaries contain factually accurate information. We introduce the calculation of the evaluation metrics as follows.

Mean of Absolute Deviation (MAD, Liu et al., 2018) is used to evaluate the distance between the target length L_{target} and the length of the system-generated summary L_{sys}. We categorize summaries into bins based on their lengths: bin 1 contains 0-50 words, bin 2 has 50-100 words, and so forth. We calculate MAD using the bin number of corresponding length with the following equation:

$$MAD = \frac{1}{N}\sum_{n}^{N} |L_{sys} - L_{target}|. \qquad (1)$$

Pearson Correlation Coefficient (PCC, Liu et al., 2018), which is a number between –1 and 1 that measures the strength and direction of the relationship between two variables. A number between 0 and 1 means when one variable changes, the other variable changes in the same direction, and vice versa. For each test instance, we generate five summaries with length signals of 50, 100, 150, 200, and 250 words. Subsequently, we compute the PCC between the actual length and the control signal for each summary.

Success Rate (SR, Fan et al., 2018), is the fraction of keywords actually occurring in the output summaries. We calculate SR employing exact matching after stemming.

Factual Correctness (FACT, Krishna et al., 2023), is a fine-grained evaluation by human annotators. We ask annotators to make a binary 0/1 judgment for the correctness of each sentence, based on whether it can be entailed or logically inferred from the provided source document. We use voting to decide the final score of the sentence based on judgments of multiple annotators. The FACT score of a batch of generated summaries is the average number of binary scores of all sentences.

We recruit 3 students from our university for annotation, who are majored in computer science and fluent in scientific document reading in English. These annotators are carefully selected based on their expertise and demonstrated high performance in a trial task, ensuring the quality and reliability of their evaluations. The average annotation time for the three annotators is 18 hours, and we provide reasonable compensation based on their working hours.

C Experimental Settings

All input source documents are truncated to 1024 tokens to fit in the model. To finetune models, we use a learning rate of $5e-4$, applying StepLR for learning rate scheduling. We conduct all experiments with an RTX 3090 GPU with 24GB memory.

[9] Full article success rate in (He et al., 2022)