Abstractive Summarizers are Excellent Extractive Summarizers

Daniel Varab
Novo Nordisk
IT University of Copenhagen
djam@itu.dk

Yumo Xu
School of Informatics
University of Edinburgh
yumo.xu@ed.ac.uk

Abstract

In this paper, we explore the efficacy of modeling extractive summarization with an abstractive summarization system. We propose three novel inference algorithms for sequence-to-sequence models, evaluate them on established summarization benchmarks, and show that recent advancements in abstractive designs have enabled them to compete directly with extractive systems with custom extractive architectures. We show for the first time that a single model can simultaneously produce both state-of-the-art abstractive and extractive summaries, introducing a unified paradigm for summarization systems. Our results question fundamental concepts of extractive systems and pave the way for a new paradigm - generative modeling for extractive summarization.[1]

1 Introduction

Extractive summarization selects a set of salient sentences from the original document(s) and composes them into a summary. Compared to abstractive summaries, made up of words or phrases that do not appear in the input document, extractive summaries are less flexible but avoid inconsistencies and hallucinations. The pipeline for building an extractive summarizer typically consists of two separate stages: *sentence labeling* and *extractive modeling*. Since few summarization datasets come with gold labels indicating which document sentences are summary-worthy, the first step is to create *oracle* sentence labels (Nallapati et al., 2017). The task is commonly *modeled* with a sequence labeling architecture (Cheng and Lapata, 2016) where a salience score is estimated for each document sentence, and top-ranked sentences are selected for summary inclusion. Recent work has expanded extractive modeling to higher-order sentence selection to account for complex label dependencies, via extracting sentences stepwise (Narayan et al., 2020), or reranking a small set of summary candidates (Zhong et al., 2020; An et al., 2022).

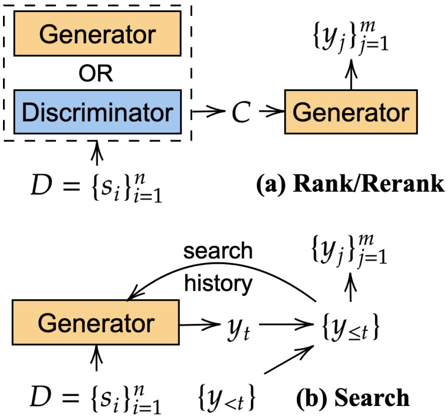

Figure 1: Proposed inference methods for GenX. We show (a) a two-stage approach with generative ranking/reranking, which creates a set of candidate summaries C from document sentences D, and (b) a single-stage inference method, generative search, which extracts summary sentences y_t autoregressively.

In this work, we revisit these fundamental concepts in extractive summarization. Specifically, we highlight that heuristically-derived sentence labels can be highly suboptimal (Narayan et al., 2018b; Xu and Lapata, 2022b), and that customized neural architectures for extractive modeling prevent taking advantage of independent improvements. We recognize that generative modeling with a neural encoder-decoder architecture (Bahdanau et al., 2015; Sutskever et al., 2014), the *de facto* choice for abstractive summarization (Nallapati et al., 2017; Zhang et al., 2020; Lewis et al., 2020), constitutes a promising direction for extractive summarization. In particular, such models learn directly from abstractive references and do, therefore, not require sentence labeling, while also embodying the extractive capabilities previously enabled by specialized neural architectures. Existing literature

[1] We distribute the code to replicate the results presented in the paper at https://github.com/danielvarab/GenX.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 330–339
July 9-14, 2023 ©2023 Association for Computational Linguistics

has established varied and many connections between abstractive and extractive modeling such as copy mechanism (See et al., 2017), content selection (Kedzie et al., 2018; Gehrmann et al., 2018), and generation guidance (Dou et al., 2021). These connections, however, are mostly *abstract-centric* which are identified or constructed to improve abstractive summarization. In contrast, there are few studies from an *extract-centric* point of view.

In this work, we propose a new summarization paradigm that unifies extractive and abstractive summarization with generative modeling, *without* compromising abstractive performance. To this end, we treat extractive summarization as an *inference*-time task and explore methods for adapting a pre-trained abstractive system for extractive summarization without further optimization. We hypothesize that an abstractive system can be used as a summary evaluator for not only abstracts but extracts as well. A model optimized on abstractve references should be able to provide an accurate quality estimation for an extractive candidate summary when conditioned on the input document. A straightforward approach to validate this assumption is to search for the best document extract with an abstractive model for candidate evaluation. However, performing an exhaustive search over a combinatorial space of all eligible summary candidates is computationally intractable. To tackle this challenge, we propose GenX, **Gen**erative e**X**tractive summarization, which introduces a set of inference algorithms (shown in Figure 1) to reduce the search complexity via various approximations of the entire search space, at either sentence- or summary-level.

Experiments show that GenX achieves competitive or superior performance compared to custom systems developed for extractive summarization on the CNN/DM benchmark without compromising its ability to generate abstracts. Particularly, for one-stage summarization the proposed method shows superior results to custom extractive state-of-the-art systems. GenX also exhibits high robustness in zero-shot transfer: on XSum, its zero-shot performance surprisingly surpasses its fully supervised counterpart. We further conduct an extensive analysis of GenX's properties, providing potential directions for future research on generative modeling for extractive summarization.

2 Generative Modeling for Extracts

Given a generative model θ trained on summarization data comprising documents and abstractive references, at inference time, for an input document D and a summary sequence Y, we estimate the length-normalized log probability of Y, following the standard practice in neural text generation (Cho et al., 2014):

$$p_\theta(Y|D) = \frac{1}{|Y|}\sum_{t=1}^{|Y|} \log p_\theta(Y_t|D, Y_{<t}) \quad (1)$$

As θ is optimized at the token level, we evaluate both *complete* and *partial* summaries with $p_\theta(Y|D)$.

The candidate summary space for a document $D = \{s_i\}_{i=1}^{n}$ of n sentences is combinatorial, consisting of $|\mathbb{C}(D)| = C\binom{m}{n}$ candidate summaries of length m. To sidestep the computational intractability, we introduce three inference algorithms that reduce the search complexity via approximations. The first two (ranking and reranking) construct a candidate summary set, using either a discriminative or generative model (see Figure 1(a)), while the last approach searches directly over the partial summary candidate space (see Figure 1(b)).

Generative Ranking We employ a pre-trained generative model at both sentence- and summary-level for hierarchical ranking. Specifically, we input each document sentence s into a generator and evaluate its summary-worthiness independently via its likelihood. We then rank all document sentences, and any subset of size m of the top-k sentences is considered as a candidate summary c. The sequence-to-sequence generator then evaluates and ranks all candidate summaries, and the highest-ranked one is selected as the extractive hypothesis:

$$y = \underset{c \subseteq \text{top-k}\, p_\theta(s|D)}{\operatorname{argmax}} \; p_\theta\left(\oplus(c)|D\right) \quad (2)$$

where $\oplus$ concatenates the selected document sentences in c, ordered by their rank.

Generative Reranking Instead of using the same generative model for both sentence and summary evaluation, we assume access to an existing discriminative model $p_\phi(s|D)$ for sentence evaluation and ranking. Following Zhong et al. (2020), we adopt BERTSUMEXT (Liu and Lapata, 2019) to score each document sentence and then build candidate summaries as the combinations of top-scoring

Model	**R-1**	**R-2**	**R-L**
Lead-3	40.42	17.62	36.67
Oracle	52.59	31.23	48.87
One-Stage Systems			
BERTSumExt	*42.73*	*20.13*	*39.20*
RoBERTaSumExt	*42.99*	20.60	*39.21*
Stepwise ETCSum	***43.84***	***20.80***	*39.77*
GenX (Search)	43.57	20.55	**40.01**
Two-Stage Systems			
BertSumExt+TRB	*43.18*	*20.16*	*39.56*
RoBERTaSumExt+TRB	*43.30*	*20.58*	*39.48*
MatchSum	***44.41***	**20.86**	***40.55***
Posthoc Rank	*39.77*	*18.51*	*36.00*
GenX (Rank)	*42.90*	*19.99*	*39.09*
GenX (Rerank)	43.76	20.82	40.02

Table 1: Results on CNN/DM test set. We bold **highest** scores, and italicize scores of one-stage and two-stage systems that are *outside the 95% confidence interval* of GenX (Search) and GenX (Rerank), respectively (with 95% confidence interval via bootstrap resampling (Davison and Hinkley, 1997)).

Model	**R-1**	**R-2**	**R-L**
BertSumExt (ZS)	20.54	2.93	15.55
BertSumExt+TRB (ZS)	20.62	2.95	15.62
MatchSum (ZS)	20.90	**3.07**	15.75
GenX (Search; Supervised)	17.90	2.79	13.36
GenX (Search; ZS)	**20.94**	2.96	**15.92**

Table 2: Results on XSum test set. We highlight **highest** scores. ZS denotes zero-shot performance for models trained on CNN/DM while Supervised uses XSum for training.

sentences. In this case, the role of generative modeling is a summary-level reranker $p_\theta(\oplus(c)|D)$.

Generative Search Instead of ranking, we consider constructing a summary by searching directly over the *sentence* space, i.e., without first composing candidate summaries from the input document. We propose a novel search algorithm that autoregressively selects a sentence until a stopping criterion is satisfied. Specifically, at each search step t, we evaluate and select a sentence as:

$$y_t = \operatorname*{argmax}_{s \in D} p_\theta(y_{<t} \oplus s|D) \tag{3}$$

where $\oplus$ concatenates the selected sentences $y_{<t}$ and a candidate sentence s. The selected sentence y_t is then concatenated with $y_{<t}$ to form the selection history for the next step, as shown in Figure 1(c). We follow common practice in non-autoregressive extractive summarization (Liu and Lapata, 2019; Zhong et al., 2020) and assume a fixed number of sentences in the summary hypothesis, leading to a fixed number of search steps. Narayan et al. (2020) introduced a stepwise model which employs a special stop-token where the search stops when the token is generated. To explore this we additionally experiment with a dynamic stopping criterion where search over sentences continues until the end of the sequence token, EOS , provides a higher summary likelihood than adding an additional sentence:

$$\text{s.t.} \max_{s \in D} p_\theta(y_{<t} \oplus s|D) > p_\theta(y_{<t} \oplus \texttt{EOS}|D). \tag{4}$$

3 Experimental Setup

We perform supervised experiments on CNN/DM (Hermann et al., 2015) and zero-shot experiments on XSum (Narayan et al., 2018a). We evaluate summaries with ROUGE (Lin and Hovy, 2003). Details for our experimental settings and datasets can be found in Appendix A.

As there is no established baseline for extractive summarization with generative modeling, we construct **Posthoc Rank**, a posthoc method for direct comparison with GenX. The baseline first generates an abstract using the abstractive model. Then, the generated abstract is used to query document sentences and m sentences are retrieved with BM25 as the summary while applying tri-gram blocking.

4 Results

Supervised Summarization Table 1 shows the results of various systems trained and evaluated on CNN/DM. The first block presents the performance of the Lead-3 baseline which considers the first 3 sentences in a document as the summary and an Oracle baseline which serves as an upper bound.

The second block reports the performance of one-stage summarization systems. Stepwise ECTSum (Narayan et al., 2020) is a state-of-the-art autoregressive system that learns to score partial summaries by selecting which sentence is a summary sentence iteratively. Different from GenX, it is a highly-customized extractive architecture optimized with extractive oracle summaries. As can be seen, GenX performs on par with Stepwise ETCSum, and outperforms BERTSumExt (Liu and Lapata, 2019) and RoBERTaSumExt (Narayan et al., 2020). Two popular extractive systems based on sequence labeling.

Model	R-1	R-2	R-L
GenX (Search)	43.57	20.54	40.01
BART	↓5.11	↓4.12	↓5.08
Dynamic Stopping	↓0.11	↓0.08	↓0.10
Trigram Blocking	↓0.16	↓0.26	↓0.18

Table 3: Ablation study in CNN/DM test set.

The third block presents the results of two-stage systems. TRB denotes an additional stage for sentence selection with Trigram Blocking, an effective method for reducing redundancy. MatchSum (Zhong et al., 2020) is a state-of-the-art extractive system that takes top-ranked sentences from BERTSumExt and then re-ranks the summary candidates composed by them with a model based on a Siamese-BERT architecture. As can be seen, GenX models improve over the one-stage BERTSumExt and RoBERTaSumExt, i.e., with or without BERTSumExt as a sentence-level ranker. Its reranking variant also outperforms BERTSumExt+TRB and RoBERTaSumExt+TRB, showing that generative summary-level evaluation is more effective than heuristically-derived selection criteria. Note, the performance of GenX still falls short of state-of-the-art MatchSum. This is all achieved while the design allows the base generative model to retain its ability to produce abstractive summaries. This is not applicable to any existing extractive systems except Posthoc Rank, which shows significantly inferior performance.

Zero-Shot Summarization We also examine the generalization capability of extractive systems in a *zero-shot* setting.[2] As shown in Table 2, GenX generalizes to a different dataset robustly, outperforming strong one- and two-stage systems. It is generally perceived that a model's zero-shot performance is inferior to the supervised performance. Surprisingly, GenX performs substantially better in the zero-shot setting than its supervised counterpart. One potential reason for this is that despite the discrepancy between training and inference, CNN/DM is a more extractive dataset than XSum (Liu and Lapata, 2019), and therefore contains more extract-specific knowledge. Compared to existing systems, GenX is more capable of transferring the extractive ability learned from CNN/DM to XSum. This shows that treating extractive summarization as an *inference* task can significantly reduce the risk of overfitting to one specific dataset, shedding light on a new direction for knowledge transferring in zero-shot summarization.

5 Ablation Study

We further assessed GenX with an ablation study. Replacing BRIO (trained with MLE and Contrastive Loss) with Bart (trained with MLE) leads to the largest performance drop. With the augmentation of contrastive learning, the abstractive system is competent in the dual role of both a generation and evaluation model, emphasizing the importance of calibrating a generative model on its summary-level probability, even for its extractive inference.

The dynamic stopping mechanism introduced in Equation (4) performs on par with fixed-step search, showing that learning directly from abstracts is a promising way to teach models *when to stop* for summary extraction. GenX is also shown to be able to search for extractive summaries of less redundancy as its performance can *not* be further improved by incorporating Trigram Blocking.

6 Efficiency

We have shown that abstractive systems are capable extractive summarizers, however, it is important to highlight that the proposed method exhibits different computational requirements than that of contemporary extractive designs. Unlike extractive designs that compute a single score for a candidate sentence or summary (via a classification token), abstractive systems produce scores for all individual tokens in a candidate summary[3]. Computing these extra tokens causes approaches such as *ranking* and *reranking* with GenX more computationally demanding. However, when combined with search GenX stands as an efficient solution to searching through an otherwise intractable candidate summary space. This is enabled by an abstractive system's ability to sequentially score text (see Equation 1) and boils down to the complexity of beam search. This is a clear improvement in computational efficiency over systems like MatchSum which only supports scoring complete summaries and must exhaustively recompute different permutations in the candidate summary spaces. To make this strategy computationally tractable these models resort to heavy pruning which limits the

[2] We do not include zero-shot results of Stepwise ETCSum as there are no publicly available code or models.

[3] For the sake of generality we ignore computational costs related to encoding as this varies across models but emphasize that it can have sizable practical implications.

expressiveness that high-order modeling otherwise enables.

7 Related Work

There is a plethora of work on controlling different aspects of summarization, from content (Xu and Lapata, 2022a; Ahuja et al., 2022) to formats (Zhong et al., 2022). In this work, we offer efficient and effective control over the summary type (extract versus abstract) during inference. Recent work also investigates how to treat discriminative tasks such as information extraction and retrieval with generative modeling and its effectiveness for entities (De Cao et al., 2020) and string identifiers (Bevilacqua et al., 2022). Others have suggested delegating extractive inference to the encoder of a generative model (An et al., 2022). Despite the resemblances, extractive summarization with generative modeling remains under-explored and stands as a promising research direction with the surge of innovations in large language models.

8 Conclusion

In this paper, we explored the possibility of modeling extractive summarization with an abstractive system. We proposed three novel inference algorithms which allow an abstractive model to perform the extractive task. Our results showed that not only is extractive summarization feasible, but recent systems are directly competitive with contemporary extractive systems. This work shows that extractive and abstractive paradigms can be unified through a sequence-to-sequence design, removing the need for oracle summary labels and custom extractive model architectures.

9 Limitations

One potential way to improve the extractive performance of a generative system is to explicitly model the likelihood of *extracts* during training. Driven by this intuition, we investigate creating a mixture of extractive and abstractive candidates for contrastive learning in BRIO. Specifically, we obtain extractive candidates with beam labeling proposed in Xu and Lapata (2022b), while the abstractive ones are from the original BRIO training data. Nevertheless, as we can see, this mixing method hurts both BRIO's extractive and abstractive performance. However, it is noteworthy that extractive summary is important in a wider context, as shown in Section 4: reference summaries in CNN/DM are highly extractive and optimizing a model on these summaries therefore may have provided it with the task instruction needed for extractive summarization, albeit implicitly. We leave the study of a more effective extract-aware learning strategy for future study.

Furthermore, we emphasize that the conclusions drawn in this paper are based on results produced on English datasets from the news domain. Even though these datasets are established benchmark datasets for summarization it is imaginable that other domains and languages may have produced different evidence. Despite this, the results remain insightful as the results show that extractive summarization is in fact feasible with modern abstractive systems. In future research, we look forward to shedding light on the possibilities and limitations of the proposed methods in a broader context.

References

Ojas Ahuja, Jiacheng Xu, Akshay Gupta, Kevin Horecka, and Greg Durrett. 2022. ASPECTNEWS: Aspect-oriented summarization of news documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6494–6506, Dublin, Ireland. Association for Computational Linguistics.

Chenxin An, Ming Zhong, Zhiyong Wu, Qin Zhu, Xuanjing Huang, and Xipeng Qiu. 2022. CoLo: A contrastive learning based re-ranking framework for one-stage summarization. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5783–5793, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. In *Advances in Neural Information Processing Systems*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 484–494, Berlin, Germany.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Anthony Christopher Davison and David Victor Hinkley. 1997. *Bootstrap methods and their application*. 1. Cambridge university press.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. In *International Conference on Learning Representations*.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 1693—-1701, Cambridge, MA, USA.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium.

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

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 71–78, Edmonton, Canada.

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

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. BRIO: Bringing order to abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2890–2903, Dublin, Ireland. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 3075–3081, San Francisco, California, USA.

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

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759, New Orleans, Louisiana.

Shashi Narayan, Joshua Maynez, Jakub Adamek, Daniele Pighin, Blaz Bratanic, and Ryan McDonald. 2020. Stepwise extractive summarization and planning with structured transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4143–4159, Online. Association for Computational Linguistics.

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

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Yumo Xu and Mirella Lapata. 2022a. Document summarization with latent queries. *Transactions of the Association for Computational Linguistics*, 10:623–638.

Yumo Xu and Mirella Lapata. 2022b. Text summarization with oracle expectation. *arXiv preprint arXiv:2209.12714*.

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

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

Ming Zhong, Yang Liu, Suyu Ge, Yuning Mao, Yizhu Jiao, Xingxing Zhang, Yichong Xu, Chenguang Zhu, Michael Zeng, and Jiawei Han. 2022. Unsupervised summarization with customized granularities. *arXiv preprint arXiv:2201.12502*.

A Implementation Details

We show detailed data statistics in Table 4. For our GenX experiments, we use the BRIO system (Liu et al., 2022) as our underlying abstractive model. To replicate the BRIO system we run the published code repository associated with the paper. Specifically, we initialize a BART model with the Huggingface Models Hub checkpoint `facebook/bart-large-cnn` and fine-tune it with the provided configuration using the training scheme presented in the paper on both the CNN/Dailymail, and XSum dataset using the data distributed in said repository. We train the model with full precision on a single machine with four Tesla V100 GPUs for 30 hours and choose the checkpoint with the lowest cross-entropy (generative) loss term on a held-out validation set. Interestingly choosing the checkpoint with the lowest contrastive term produces poor results. Also, using mixed precision training doesn't appear to work.

To run the inference algorithms we initialize a BART system with different weights, either obtained through the above training procedure (BRIO) or the baseline `facebook/bart-large-cnn` checkpoint. The hyperparameter m is identical to the desired length of the generated summary. m was tuned on the validation set and set to 3 for the CNN/DM dataset, and 2 for the XSum. k was set to 5, following MatchSum system. We studied the effects of various length penalties in Equation 1 and did not find our approach sensitive to its choice and omitted it from the equation. For this computation we run the model under fp16 mixed precision to save memory, however, casting the model entirely to half-precision for inference does not appear to work.

Datasets	CNN/DM	XSum
Language	En	En
Domain	Newswire	Newswire
#Train	287,084	203,02
#Validation	13,367	11,273
#Test	11,489	11,332
#Sentences in Extract	3	2

Table 4: Data statistics for extractive summarization.

We used standard parameter settings for all experiments: ROUGE-1.5.5.pl -c 95 -m -r 1000 -n 2 -a.

B License Information

The datasets used in this work, CNN/DM (Hermann et al., 2015) and XSum (Narayan et al., 2018a), are both released under the MIT License.

C System Output

Document: We spend a third of our lives asleep, but most of us don't pay attention to what our mind and body actually need during these resting hours in order to feel refreshed every day. The Sleep Health Foundation have released a study reporting that 30 percent of Australians complain about their lack of sleep on a daily basis. According to Chair Professor David Hillman, those misplaced hours of sleep must be paid back in order to be functional for the entire week. A study has outlined that 30 percent of Australians complain about their lack of sleep on a daily basis. The average adult needs around eight hours of sleep per night with a range of seven to nine. The average amount of sleep for an adult is around eight hours, with a range of seven to nine, the ABC have reported. Any less than six hours or any more than 10 hours is unusual for the standard person. Professor Hillman added that our sleep pattern is influenced by how much we are willing to compromise from the work week. 'A lot of us pay back a bit of that debt on the weekend but I think it's possible to exist in a sort of tolerable, sleep-restricted state,' he said. 'In other words you're not optimal, but you're still functional.' Pushing these sleep-debt boundaries can lead to micro sleeps in certain people. Therefore, the hours must be paid back to avoid an error rate in alertness tasks. Any less than six or any more than ten hours is unusual for the standard person. If power napping, it is important to get no more than 20 minutes or inertia will set in. In relation to a sleep schedule, Professor Hillman said the eight hours per night does not necessarily need to be consecutive. 'Interestingly enough, your slow wave sleep, is in the first four hours,' he said. 'Most adults, the most convenient way our particular society is organised is to have your eight hours in a continuous block overnight but that's not a necessary thing.' If choosing to break up your eight hours of sleep, napping throughout the day is the answer. Professor Hillman advises 20 minute power naps to avoid falling into deep sleep and suffering from inertia which makes you feel temporarily worse off. 'The longer naps, you get the sleep inertia but ultimately once you've got up, they sustain you better,' he said. Professor Hillman has also advised that if you are waking up tired and fatigued it could be due to sleep apnoea which is often associated with snoring.

Reference Summary: The Sleep Foundation study has shown that adults need 8 hours of sleep. According to the study, 30 percent of Australians say they lack sleep daily. Professor David Hillman said it's important to pay back our sleep debts. He also says sleep can be broken up as long as you get the first 4 hours. Power naps should not be longer than 20 minutes or inertia will set in.

BertSumExt: The Sleep Health Foundation have released a study reporting that 30 percent of Australians complain about their lack of sleep on a daily basis. The average adult needs around eight hours of sleep per night with a range of seven to nine. Any less than six hours or any more than 10 hours is unusual for the standard person.

MatchSum: The Sleep Health Foundation have released a study reporting that 30 percent of Australians complain about their lack of sleep on a daily basis. The average adult needs around eight hours of sleep per night with a range of seven to nine.

GenX (Search): A study has outlined that 30 percent of Australians complain about their lack of sleep on a daily basis. The average adult needs around eight hours of sleep per night with a range of seven to nine. According to Chair Professor David Hillman, those misplaced hours of sleep must be paid back in order to be functional for the entire week.

Table 5: Examples of system output on the CNN/DM test set. BertSumExt adds an unnecessary sentence highlighted in red. MatchSum removes this sentence, while GenX adds an additional sentence, highlighted in blue, which is reflected in the reference summary but missing from the other two system outputs.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 9.

☒ A2. Did you discuss any potential risks of your work?
Section 9.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 3 and 4.

☑ B1. Did you cite the creators of artifacts you used?
Section 3-4 and Appendix A-B.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Appendix B.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Appendix A.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
We used existing benchmarks as they are for fair comparisons.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Appendix A.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix A.

C ☑ Did you run computational experiments?

Section 4.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.