Extractive Summarization via Fine-grained Semantic Tuple Extraction

Yubin Ge[1], **Sullam Jeoung**[1], **Jana Diesner**[1, 2]
[1]University of Illinois Urbana Champaign, USA
[2]Technical University of Munich, Germany
{yubinge2,sjeoung,jdiesner}@illinois.edu

Abstract

Traditional extractive summarization treats the task as sentence-level classification and requires a fixed number of sentences for extraction. However, this rigid constraint on the number of sentences to extract may hinder model generalization due to varied summary lengths across datasets. In this work, we leverage the interrelation between information extraction (IE) and text summarization, and introduce a fine-grained autoregressive method for extractive summarization through semantic tuple extraction. Specifically, we represent each sentence as a set of semantic tuples, where tuples are predicate-argument structures derived from conducting IE. Then we adopt a Transformer-based autoregressive model to extract the tuples corresponding to the target summary given a source document. In inference, a greedy approach is proposed to select source sentences to cover extracted tuples, eliminating the need for a fixed number. Our experiments on CNN/DM and NYT demonstrate the method's superiority over strong baselines. Through the zero-shot setting for testing the generalization of models to diverse summary lengths across datasets, we further show our method outperforms baselines, including ChatGPT.

1 Introduction

The objective of automatic text summarization is to condense the content of an original document while preserving its essential information. Existing summarization techniques can be categorized into two main approaches: extractive and abstractive methods (Ge et al., 2023b). Abstractive methods aim to generate new sentences, often referred to as paraphrased sentences, to compose a summary (Widyassari et al., 2020), while extractive techniques generate summaries by selecting and extracting salient sentences directly from the source text (Kasture et al., 2014).

In this study, we focus on extractive summarization, primarily formulated as sentence-level classification. This task typically involves a greedy method to derive binary labels for sentences in a source document, indicating their inclusion or exclusion in the summary (Nallapati et al., 2017). Nevertheless, previous research (Zhou et al., 2020) demonstrates the drawbacks of this sentence-centric granularity for extraction as it can introduce redundancy and unnecessary information into the output.

Besides, during inference, a fixed-length cutoff or threshold is often applied to restrict the sentence length of the output summary. This practice is inherently limited as it fails to accommodate the varying characteristics of different documents, which may necessitate extractive summaries of different lengths. For instance, a long document may need more sentences to comprehensively cover its salient information, whereas a short document may suffice with a more concise representation. Additionally, in real-world applications, expecting users to specify the exact number of sentences to be extracted when utilizing a summarization system may not be always feasible or practical.

Motivated by the shortcomings outlined above, we present a new fine-grained autoregressive approach for extractive summarization via semantic tuples extraction. To this end, we exploit the inherent interdependence between information extraction (IE) and text summarization as both tasks share a common objective: extracting accurate information from unstructured texts in alignment with a user's specific requirements and presenting the extracted information in a concise manner (Grishman et al., 1999). While summarization aims to present this information in natural language sentences, IE aims to transform relevant information into structured representations (Ji et al., 2013).

To effectuate this integration, we first use an IE tool to convert each sentence into a semantic meaning representation based on predicate-argument structures (Surdeanu et al., 2003), which we call

Proceedings of the 17th International Natural Language Generation Conference, pages 121–133
September 23–27, 2024. ©2024 Association for Computational Linguistics

semantic tuples in this work. We identify these semantic tuples corresponding to the target summary as the objective of extraction. Leveraging a Transformer-based autoregressive model (Vaswani et al., 2017), we train the model to extract the target semantic tuples from each source document. This can encourage the model to concentrate on salient information at a more granular level compared to conventional approaches that perform extraction at the sentence level. During inference, we introduce a greedy strategy to select source sentences that cover the extracted semantic tuples, avoiding the requirement to specify a fixed number of sentences for extraction.

By following standard evaluation protocols, we demonstrate that our proposed method outperforms competitive baselines on CNN/DM and NYT. Furthermore, to highlight the advantage of our approach, we examine the impact of fixed sentence extraction requirements on model generalization under a zero-shot setting. This involves assessing the model's performance on a different dataset, where the anticipated summary lengths deviate from those in the training data. In contrast to baselines that consistently output summaries of the same length for different documents, our method excels due to its capacity to dynamically extract sentences to cover the identified semantic tuples.

We also compare the proposed approach to using ChatGPT (Brown et al., 2020). To do this, we provide ChatGPT with a prompt without specifying the number of sentences to extract. The results reveal the low performance of ChatGPT in this task —a revelation consistent with recent work (Zhang et al., 2023). Upon manual examination of the extractive summaries output by ChatGPT, we discovered that ChatGPT tends to optimize recall by selecting more sentences than expected. While ChatGPT has demonstrated commendable capabilities across a diverse spectrum of tasks, our observations suggest that current fine-tuning approaches on smaller models may still present promising avenues for enhancing extractive summarization performance.

Our contributions can be summarized as follows:

- We introduce a new, fine-grained, autoregressive method for extractive summarization by using semantic tuples extraction.

- Leveraging the extracted semantic tuples, we present a greedy strategy for selecting sentences to construct extractive summaries. Notably, our approach avoids the convention of necessitating a predetermined number of sentences for extraction.

- Through extensive experiments, we empirically demonstrate the superior efficacy of our method over competitive baselines. Our approach excels under the demanding zero-shot setting.

- We test ChatGPT for extractive summarization and uncover that ChatGPT's performance is inferior in this task. Our findings signify the ongoing significance of exploring mainstream fine-tuning approaches for future research.

2 Related Work

2.1 Extractive Summarization

Extractive summarization, an NLP task with decades of exploration, has been approached with a wide array of methods. Sequential neural models, which use diverse encoders such as recurrent neural networks (Cheng and Lapata, 2016; Nallapati et al., 2017; Xiao and Carenini, 2019), and pre-trained language models (Zhou et al., 2018; Egonmwan and Chali, 2019; Liu and Lapata, 2019) are frequently adopted for this task. Another trajectory in research conceptualizes extractive summarization as a node classification task and solves it by leveraging graph neural networks to model inter-sentence relationships (Wang et al., 2020; Zhang et al., 2022). Despite the sophistication of these approaches, they are formulated as sentence-level predictions and require the specification of a fixed quantity of sentences for extraction. Alternatives to the sentence-centric focus are text matching (Zhong et al., 2020; An et al., 2022) and reinforcement learning (Narayan et al., 2018b; Bae et al., 2019), which have been explored through summary-level formulations. Our approach departs from these prior undertakings by honing in on a more refined granularity. Specifically, we extract semantic tuples, which we consider as semantic representations of textual content.

2.2 Text Summarization and Information Extraction

Previous studies of the relationship between information extraction (IE) and text summarization have demonstrated advantages of integrating IE methods into text summarization, including the capacity to enhance the overall quality of summarization outcomes in different domains (McKeown and Kan,

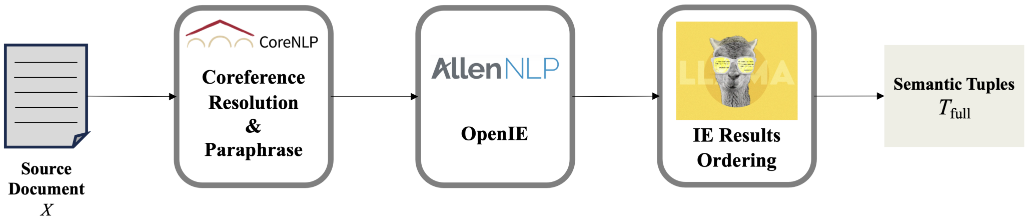

Figure 1: An overview of the pipeline for semantic tuples extraction from a document.

1999). Furthermore, incorporating IE has improved the coherence of multi-document abstract summarization (Ji et al., 2013; Li, 2015; Venkatachalam et al., 2020). In line with our current approach, Litvak and Last introduced a graph-based IE method for summarization. Their work represents text documents as an order-relationship graph, where nodes correspond to discrete words and edges encapsulate the sequential precedence of terms within the text. Our approach diverges from theirs by leveraging predicate-argument structures, which accommodate varying numbers of arguments. This stands in contrast to graph-based representations, which are characterized by a fixed number of elements within each triplet and are limited in representing the nuanced semantic meaning of textual content.

2.3 Flexible Extractive Summarization

The inference of extractive summarization models conventionally entails the extraction of the top-k most significant sentences from a given document, determined by predicted sentence scores. Nevertheless, employing a fixed value k for all documents tends to yield summaries of uniform length, thereby constraining the diversity in summary lengths. Although a few recent investigations (Jia et al., 2020; Zhong et al., 2020) have sought to generate summaries of variable lengths, their techniques either necessitate an additional phase of hyperparameter optimization on validation datasets to identify an appropriate threshold or frame the problem as a selection of a subset from the top-k sentences. Conversely, our approach relies on the extraction of semantic tuples, which are subsequently matched to sentences to ensure coverage in a greedy manner. Therefore we effectively eliminate both the pre-specification of summary lengths and conducting hyperparameter search.

3 Fine-grained Semantic Tuples Construction

In this section, we introduce the process of converting sentences from text into *semantic tuples*, which in our case are fine-grained semantic representations based on predicate-argument structures (Surdeanu et al., 2003). The overall pipeline is shown in Figure 1. This is different from conventional approaches for extractive summarization, which rely on sentences as the primary granularity.

To extract semantic tuples from a given source document, we employed Stanford CoreNLP (Manning et al., 2014) to first perform coreference resolution, thereby replacing identified mentions (e.g., pronouns) with their corresponding entity names. Subsequently, an IE tool was employed to extract fine-grained semantic information from the sentences: we conducted a comparative analysis of different IE systems, including AllenNLP OpenIE (Stanovsky et al., 2018), Stanford CoreNLP OpenIE (Angeli et al., 2015), knowledge base-based OpenIE (Huguet Cabot and Navigli, 2021), and AMR (Zhou et al., 2021). Our selection was based on factors such as system accessibility and IE performance on summarization datasets. Ultimately, we chose the OpenIE tool provided by AllenNLP, which enables us to extract a list of propositions from each sentence, effectively yielding semantic tuple candidates. Each semantic tuple is composed of a single predicate and a variable number of arguments. To ensure the data's integrity, we excluded any semantic tuples with arguments exceeding 20 tokens. Moreover, we associated each predicate with its arguments based on predicted argument roles, adhering to the conventions established by Surdeanu et al., where 'arg0' denotes the agent, "arg1" refers to the direct object, and "arg2" represents the indirect object.

However, upon inspecting the results, we noted that the extracted semantic tuples exhibited certain inaccuracies in the predicted argument roles, potentially leading to semantic ambiguities. Considering the high performance of LLMs in various tasks(Ge et al., 2023a), we leveraged an LLM to identify the most plausible semantic tuples from all candidates to address this concern. Specifically, for each semantic tuple, we generated permutations by exploring all possible argument role assignments, i.e., “arg0" to “arg2", and concatenated each candidate accordingly to form a text representation. For instance, one candidate semantic tuple $\{$*became*, arg1: *Evnika Saadvakass*, arg2: *a YouTube sensation*$\}$ would have been transformed into "*became Evnika Saadvakass a YouTube sensation*".

To find the most appropriate semantic tuple, we input all candidate texts into an LLM[1], calculating their perplexity. The candidate with the lowest perplexity was regarded as aligning best with the language model, thus warranting selection as the final semantic tuple. Continuing with the previous example, after querying the language model with all different combinations, we obtain $\{$arg0: *Evnika Saadvakass*, *became*, arg1: *a YouTube sensation*$\}$ as the ultimate result. This pipeline enables us to enhance the accuracy and reliability of the extracted semantic tuples, ultimately contributing to a more robust knowledge representation.

4 Methodology

The overview of the proposed method is shown in Figure 2. Given a source document $X = \{x_1, x_2, \cdots, x_{|X|}\}$ consisting of a sequence of sentences x_i, we consider each sentence x_i to have a semantic meaning representation in the form of predicate-argument structures (Surdeanu et al., 2003), namely semantic tuples. The process of extractive summarization entails the following steps:

1. Given the source document X and its comprehensive set of semantic tuples denoted as T_{full}, we first extract the subset T_{sub} from T_{full}, which corresponds to the target summary.

2. Subsequently, having identified the subset T_{sub}, we next select the minimum number of sentences x_i from the original source document X whose corresponding semantic tuples cover the subset T_{sub}, thereby constituting the final output summary.

4.1 Semantic Tuples Extraction

Inspired by the great success of applying Transformer-based generative model in various IE and semantic parsing tasks (De Cao et al., 2020; Bai et al., 2022; Josifoski et al., 2022), we present an end-to-end autoregressive formulation of semantic tuple extraction.

4.1.1 Model Training

During the training phase, we initially adopted the widely-used greedy approach (Nallapati et al., 2017) to acquire sentence-level ground-truth labels for a given source document X. These labels indicated which sentences should be extracted as target sentences to form the summary. Consequently, we identified semantic tuples corresponding to these target sentences, which constitute the target subset denoted as T_{sub}. Our goal was to extract T_{sub} from the complete set of semantic tuples T_{full}, which corresponds to the source document X.

To prepare T_{sub} for end-to-end training and linearize it as a target sequence, we introduced a special token <sep> to connect each predicate with its respective arguments. For instance, the semantic tuple $\{$arg0: *Evnika Saadvakass*, *became*, arg1: *a YouTube sensation*$\}$ was transformed into "*Evnika Saadvakass* <sep> *became* <sep> *a YouTube sensation*". Additionally, we introduced another special token <et> at the end of each semantic tuple sequence to connect and form the target sequence, denoted as y.

We used BART (Lewis et al., 2020) as our generative model. The primary objective of the model training was to learn the conditional probability of generating the output sequence y given the input document X in an autoregressive manner: $p_\theta(y|X) = \prod_{i=1}^{|y|} p_\theta(y_i|y_{<i}, X)$, where θ represents the model's parameters. During training, the aim was to maximize the conditional log-likelihood of the target sequences using the cross-entropy loss, and label smoothing was applied as a regularization technique (Szegedy et al., 2016).

4.1.2 Constrained Decoding with Local Tries

One challenge with common generative models, such as BART, is that they generate unrestricted, free-form text without explicit constraints. Consequently, the trained model may generate invalid semantic tuples that do not correspond to any semantic tuples present in the complete set T_{full}. To overcome this issue, previous work in generative IE and entity retrieval (De Cao et al., 2020; Josifoski et al., 2022) has resorted to constrained beam search, establishing constraints through the use of a prefix tree (aka trie) (Cormen et al., 2022). Specifically, two distinct tries are constructed in those prior studies based on all entity names and all relations. Each node in the trie represents a token from a predefined vocabulary, and its children encompass all allowable continuations stemming from

[1] We adopted *openlm-research/open_llama_3b* specifically.

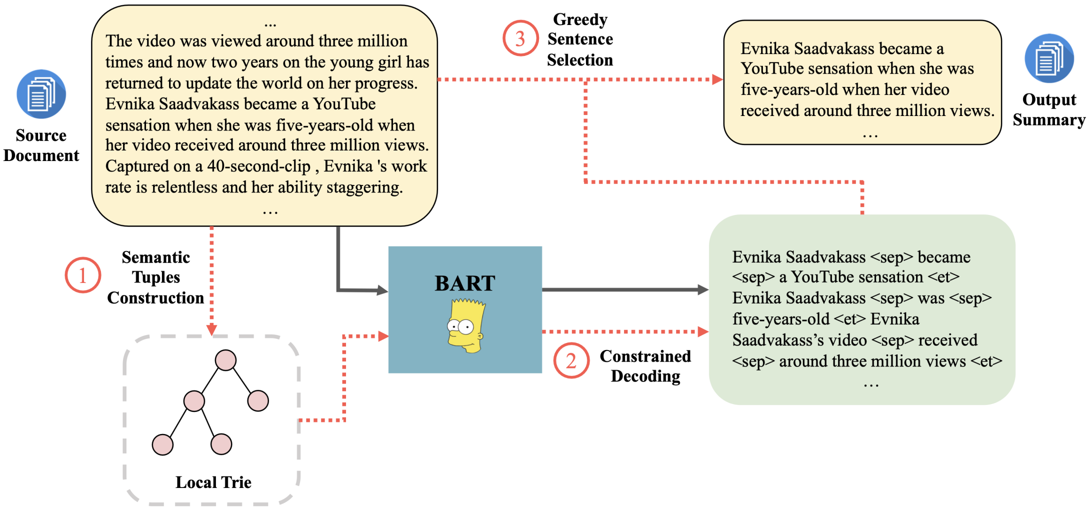
Source Document
...
The video was viewed around three million times and now two years on the young girl has returned to update the world on her progress. Evnika Saadvakass became a YouTube sensation when she was five-years-old when her video received around three million views. Captured on a 40-second-clip , Evnika 's work rate is relentless and her ability staggering.
...
3 Greedy Sentence Selection
Evnika Saadvakass became a YouTube sensation when she was five-years-old when her video received around three million views.
...
Output Summary
1 Semantic Tuples Construction
BART
Evnika Saadvakass <sep> became <sep> a YouTube sensation <et> Evnika Saadvakass <sep> was <sep> five-years-old <et> Evnika Saadvakass's video <sep> received <sep> around three million views <et>
...
2 Constrained Decoding
Local Trie

Figure 2: An overview of the proposed method. Grey solid arrows indicate the data flow during training. Red dashed arrows represent the additional data flow during inference. The inference consists of three steps: (1) construct semantic tuples from a source document and build a local trie; (2) run constrained decoding based on the built local trie to ensure extracted semantic tuples are valid; (3) select sentences from the source document to cover extracted semantic tuples in a greedy manner.

the prefix defined by traversing the trie from the root. Using a similar mechanism for our case can ensure that a traversal from the root to a leaf node guarantees the generation of a valid predicate or argument.

Nonetheless, directly applying the aforementioned strategy cannot ensure the accuracy of generated semantic tuples for our case. This limitation arises due to the inherent independence and static nature of the two pre-built tries, which we refer to as **global tries**. Consequently, during the generation process, the model remains susceptible to producing invalid semantic tuples comprising disconnected predicates and arguments. For instance, the model may generate a tuple like { arg0: *Chicago*, *helps*, arg1: *dog* }, wherein the model switches between two independent tries. To address this concern effectively, we propose the dynamic construction of a **local trie** in real time. Specifically, to generate an extractive summary for a source document X, we create a trie that stores all semantic tuples present in T_{full}. Traversing this trie from the root to a leaf node guarantees the generation of a valid and complete semantic tuple. Subsequently, we incorporate the constructed tries into the constrained beam search, following previous work (De Cao et al., 2020; Josifoski et al., 2022).

4.2 Source Sentence Extraction

During the inference phase, upon identifying T_{sub}, the task at hand involves mapping T_{sub} back to sentences within the source document X to generate an extractive summary. To achieve this objective, we have devised a pragmatic and flexible approach, inspired by the idea of deriving sentence-level ground-truth labels (Nallapati et al., 2017). Importantly, our proposed approach does not impose a fixed number of sentences to be extracted, as is commonly seen in prior methodologies.

Specifically, we adopt a greedy strategy to iteratively select one sentence x_i at a time, gradually building a summary. This selection is guided by the criterion that the semantic tuples of the chosen sentence x_i exhibit the most significant overlap with the elements in T_{sub}. After one optimal sentence is selected at a time, we remove the semantic tuples that correspond to the selected sentence from T_{sub}. This process is repeated until T_{sub} becomes empty, signifying that the final summary has encompassed all the identified semantic tuples within T_{sub}.

5 Experiments and Results

We introduced our experimental settings and results in this section, and included the implementation details in Appendix Sec. A. Additionally, we follow previous work in text summarization and related tasks (Zhang et al., 2023; Ge et al., 2021) to mainly report ROUGE-1 (unigram), ROUGE-2 (bigram), and ROUGE-L (longest common subsequence) scores (Lin, 2004) for evaluation.

5.1 Datasets

We performed the evaluation on two widely recognized benchmark datasets: CNN/DM (Hermann et al., 2015; Nallapati et al., 2016) and the New York Times Annotated Corpus (NYT) (Sandhaus, 2008):

- **CNN/DM** comprises news articles from both CNN and Daily Mail. The summaries are constructed from highlighted bullet points. We used the non-anonymized version and the provided training, validation, and testing splits.

- **NYT** consists of 110,540 articles published by the New York Times. This dataset also includes summaries authored by library scientists. We processed the dataset as in previous work (Durrett et al., 2016; Liu and Lapata, 2019) to obtain training, validation, and testing splits.

Additionally, to show that fixing the number of sentences to extract can influence models' generalization even in the same domain, we designed zero-shot experiments, where we trained models on CNN/DM and tested their performance on XSum(Narayan et al., 2018a).

- **XSum** is designed for single-sentence news summarization, with each summary formulated as an answer to the question "What is the article about?". The summaries in this dataset are professionally written and often authored by the original document's author(s).

5.2 Baselines

We compared our model with several competitive baseline methods:

- **HIBERT** (Zhang et al., 2019) is a hierarchical Transformer-based model pre-trained on unlabeled data.

- **PNBERT** (Zhong et al., 2019) combines LSTM Pointer with the pre-trained BERT.

- **BERTSum** (Liu and Lapata, 2019) builds the extractive model based on BERT.

- **BERTEXT** (Bae et al., 2019) augments BERT with reinforcement learning to maximize summary-level ROUGE scores.

- **MATCHSUM** (Zhong et al., 2020) conceptualizes extractive summarization as a semantic text matching problem. It generates candidate summaries and then finds the optimal summary that is the most semantically similar to the source document.

- **COLO** (An et al., 2022) is a contrastive, learning-based re-ranking framework based on a proposed online sampling approach.

Model	R1	R2	RL
ORACLE	52.59	31.24	48.87
LEAD-3	40.42	17.62	36.67
HIBERT (2019)	42.37	19.95	38.83
PNBERT (2019)	42.69	19.60	38.85
BERTEXT (2019)	42.76	19.87	39.11
BERTSum (2019)	43.85	20.34	39.90
MATCHSUM (2020)	44.22	20.62	40.38
COLO (2022)	44.10	20.97	40.19
Ours	**44.91**	**21.54**	**40.61**

Table 1: Experimental results on CNN/DM.

We also included the results of an extractive **ORACLE** as an upper bound, and **LEAD-3** baseline (which selects the first three sentences in a document).

5.3 Experimental Results on CNN/DM

The results on CNN/DM are presented in Table 1. The average number of sentences in our generated extractive summaries is 4.87 with a variance of 1.83. Notably, our proposed method demonstrates superior performance compared to other competitive baselines. This superiority can be attributed to our model's ability to effectively concentrate on fine-grained semantic information embedded within sentences. By leveraging this capability, our approach is capable of discerning and extracting salient structured information, a feature that plays a vital role in the summarization process.

Moreover, it is evident that our novel formulation of extractive summarization, revolving around the extraction of semantic tuples, holds significant relevance for Information Extraction (IE) tasks: Traditional IE tasks typically involve extracting structured semantic information from sentences, while our task takes a step further, aiming to extract salient structured information specifically corresponding to target summaries.

We find inspiration in the remarkable achievements and state-of-the-art performances observed in performing IE and semantic parsing through autoregressive methods (De Cao et al., 2020; Josifoski et al., 2022; Bai et al., 2022). Consequently,

Model	**R1**	**R2**	**RL**
ORACLE	49.18	33.24	46.02
LEAD-3	39.58	20.11	35.78
BERTSum (2019)	46.66	26.35	42.62
MATCHSUM (2020)	46.32	26.07	42.17
Ours	**47.87**	**26.70**	**42.83**

Table 2: Experimental results on NYT. For MATCHSUM, we used the released BERTSum checkpoint to generate candidates, and then trained the matching model on NYT.

Model	**R1**	**R2**	**RL**
ORACLE	25.62	7.62	18.72
LEAD-2	14.40	1.46	10.59
BERTSum‡	22.86	4.48	17.16
BERTSum†	20.04	2.97	16.77
MATCHSUM†	21.50	3.47	16.98
Ours (trained on CNN/DM)	**23.07**	**4.53**	**17.18**

Table 3: Zero-shot testing results on XSum. ‡ represents we trained the model on XSum and † indicates we trained the model on CNN/DM. For MATCHSUM, we used the released BERTSum checkpoint to generate candidates.

our decision to adopt the autoregressive model further contributes to the performance improvement observed in our model. By building upon the capabilities of autoregressive modeling, our approach capitalizes on the strengths of this technique, enabling enhanced summarization outcomes and underscoring the potential of this approach in extractive summarization.

5.4 Experimental Results on NYT

The experimental results obtained on NYT are displayed in Table 2. Our method generates extractive summaries of different lengths, with an average sentence length of 4.01 and a variance of 1.35. Once again, our model outperforms the considered baselines, reaffirming the efficacy and potential of our proposed method. Note that all the baselines rely on fixed numbers of sentences to be extracted. However, in more realistic scenarios, users may not always have prior knowledge of how many sentences to extract when presented with a new document.

5.5 Zero-shot Experiments on XSum

To explore the impact of fixed sentence extraction requirements on the generalization of extractive models, we formulated zero-shot testing. This set of experiments enables an investigation of how the training on one dataset, characterized by certain target summary lengths, may impact the performance of the trained model during testing on a different dataset with different target summary lengths, even within the same domain. Based on this idea, we trained models on CNN/DM, where the expected number of sentences for extraction is 3, and subsequently tested on XSum, which is expected to extract only 2 sentences.

The results are presented in Table 3. We observed that the baseline BERTSum, trained on CNN/DM, achieved inferior performance compared to its performance when trained on XSum. This discrepancy in performance highlights the challenge of generalization under the zero-shot setting and can potentially be attributed to the different number of sentences that should be extracted for the two datasets.

In contrast, our model, trained on CNN/DM, outperformed the baselines trained on CNN/DM. We attribute this improvement to the new formulation of extractive summarization adopted in our approach. Unlike traditional extractive summarization, our approach encourages the model to focus on more fine-grained and semantic-structured information in the form of semantic tuples. This allows the model to effectively identify salient semantic tuples and subsequently map flexible numbers of sentences to cover these identified elements, enhancing the overall performance.

Furthermore, our model's performance is better than that of BERTSum trained on XSum, which further underscores our model's generalization capability. This might be particularly useful in real-world applications where users may not know the optimal number of sentences to be extracted. Our approach offers a solution to this problem, addressing a crucial aspect often overlooked in previous work.

5.6 Comparison with ChatGPT

We created a prompt (Appendix Sec. B) to task ChatGPT[2] to generate an extractive summary for a given source document. Unlike the prompts used by Zhang et al., our prompt does not specify the number of sentences to extract, allowing for a meaningful comparison with our method in scenarios where the number of extracted sentences is not predetermined.

[2] We used *gpt-3.5-turbo* specifically.

Model	R1	R2	RL
CNN/DM			
ChatGPT-Ext(2023)	39.25	17.09	25.64
ChatGPT-Ext(ICL)(2023)	42.38	17.27	28.41
ChatGPT	30.23	12.90	19.75
Ours	**44.51**	**21.03**	**40.41**
XSum			
ChatGPT-Ext(2023)	19.85	2.96	13.29
ChatGPT-Ext(ICL)(2023)	17.49	3.86	12.94
ChatGPT	10.50	1.22	4.33
Ours	**23.07**	**4.93**	**17.18**

Table 4: Comparison results with ChatGPT-based approaches on CNN/DM and Xsum. ICL refers to in-context learning.

Model	relevance	faithfulness
MATCHSUM	1.41	1.83
Ours	**1.74***	**1.87**

Table 5: Human evaluation results on samples from CNN/DM. $^*p < 0.05$

The outcomes are presented in Table 4. The performance of ChatGPT exhibits notable deficiencies on both CNN/DM and XSum. Notably, in comparison to the findings of Zhang et al., ChatGPT's performance diminishes when the number of sentences to extract was left unspecified. This observation underscores the susceptibility of ChatGPT's performance to fixed sentence extraction requirements, emphasizing the influence of such constraints on model generalization. Furthermore, incorporating strategies such as in-context learning (Brown et al., 2020) has been noted to marginally enhance performance, although still falling behind existing baselines.

Inspecting the generated extractive summaries (for an example see Appendix Sec C), we observed that ChatGPT demonstrates a proclivity to select an excessive number of sentences, surpassing the expected number. For instance, on average, ChatGPT extracts approximately 8 sentences for CNN/DM, whereas the expected length is 3 sentences. This suggests a potential bias of ChatGPT towards optimizing recall at the expense of precision, contributing to its suboptimal performance. This unexpected outcome underscores the imperative for future research into more effective strategies to leverage ChatGPT for extractive summarization.

5.7 Human Evaluation

We performed a human evaluation based on our model's outputs and those released by MATCHSUM. We randomly sampled 50 test instances from CNN/DM and focused on two critical aspects: **relevance** (whether the output summary is relevant to the source document) and **faithfulness** (indicating the degree to which the output summary faithfully represents the source document). Three proficient English-speaking students scored them on a scale ranging from 0 (poor) to 2 (excellent), and averages were computed for each aspect. The outcomes are presented in Table 5. We observe that our method reaches a notably higher relevance score, with both methods exhibiting comparably high levels of faithfulness. This outcome further substantiates the efficacy of our proposed method in extractive summarization.

6 Conclusion

This study introduces an innovative, fine-grained, and autoregressive technique for extractive summarization via the extraction of semantic tuples. Diverging from conventional strategies that focus on sentence-level extraction, our approach operates at a more nuanced and semantically-structured granularity. During the inference process, we use a greedy approach to select sentences to cover the extracted semantic tuples, eliminating the necessity to predefine a fixed number of sentences for extraction. Empirical assessments conducted on CNN/DM and NYT establish the superior efficacy of our method compared to competitive baselines. Furthermore, our investigation into the generalization capabilities of our approach within zero-shot settings highlights its remarkable adaptability across diverse summary lengths, outperforming baseline models and achieving better generalization. In addition, we explored the suitability of prominent large language models for the task of extractive summarization by evaluating ChatGPT's performance in generating extractive summaries. We found ChatGPT to underperform relative to baseline models, emphasizing the potential of fine-tuning-centric methodologies for enhancing summarization performance.

7 Limitations

Our work has the following limitations. First, our extraction process is based on the output from information extraction (IE). Therefore the performance and type of IE tools can impact the downstream semantic tuple extraction. With better and better performance achieved by SOTA IE, we believe our approach can also be improved.

Furthermore, our evaluation of LLMs for extractive summarization only involved ChatGPT, specifically *gpt-3.5-turbo*. To make the conclusion and findings more robust, we plan to extend the current work by including other more recent and powerful LLMs, such as Llama 2(Touvron et al., 2023).

References

Chenxin An, Ming Zhong, Zhiyong Wu, Qin Zhu, Xuan-Jing 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.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354.

Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sang-goo Lee. 2019. Summary level training of sentence rewriting for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 10–20.

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for amr parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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: Long Papers)*, pages 484–494.

Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2022. *Introduction to algorithms*. MIT press.

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

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1998–2008, Berlin, Germany. Association for Computational Linguistics.

Elozino Egonmwan and Yllias Chali. 2019. Transformer-based model for single documents neural summarization. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 70–79.

Yubin Ge, Ly Dinh, Xiaofeng Liu, Jinsong Su, Ziyao Lu, Ante Wang, and Jana Diesner. 2021. Baco: A background knowledge-and content-based framework for citing sentence generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1466–1478.

Yubin Ge, Devamanyu Hazarika, Yang Liu, and Mahdi Namazifar. 2023a. Supervised fine-tuning of large language models on human demonstrations through the lens of memorization. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Yubin Ge, Sullam Jeoung, Ly Dinh, and Jana Diesner. 2023b. Detection and mitigation of the negative impact of dataset extractivity on abstractive summarization. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada. Association for Computational Linguistics.

Ralph Grishman, Jerry Hobbs, Eduard Hovy, Antonio Sanfilippo, and Yorick Wilks. 1999. Cross-lingual information extraction and automated text summarization. *Multilingual information management: current levels and future abilities*, page 14.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Heng Ji, Benoit Favre, Wen-Pin Lin, Dan Gillick, Dilek Hakkani-Tur, and Ralph Grishman. 2013. Open-domain multi-document summarization via information extraction: Challenges and prospects. *Multi-source, multilingual information extraction and summarization*, pages 177–201.

Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. 2020. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Martin Josifoski, Nicola De Cao, Maxime Peyrard, Fabio Petroni, and Robert West. 2022. GenIE: Generative information extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4626–4643, Seattle, United States. Association for Computational Linguistics.

NR Kasture, Neha Yargal, Neha Nityanand Singh, Neha Kulkarni, and Vijay Mathur. 2014. A survey on methods of abstractive text summarization. *Int. J. Res. Merg. Sci. Technol*, 1(6):53–57.

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.

Wei Li. 2015. Abstractive multi-document summarization with semantic information extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1908–1913.

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.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Coling 2008: Proceedings of the workshop multi-source multilingual information extraction and summarization*, pages 17–24.

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.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR).*

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Kathleen McKeown and Min-yen Kan. 1999. Information extraction and summarization: Domain independence through focus types.

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 AAAI conference on artificial intelligence*, volume 31.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

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.

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.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS).*

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. 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 885–895.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 8–15.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Swathilakshmi Venkatachalam, Lakshmana Pandian Subbiah, Regan Rajendiran, and Nithya Venkatachalam. 2020. An ontology-based information extraction and summarization of multiple news articles. *International Journal of Information Technology*, 12(2):547–557.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuan-Jing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6209–6219.

Adhika Pramita Widyassari, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Abdul Syukur, Affandy Affandy, et al. 2020. Review of automatic text summarization techniques & methods. *Journal of King Saud University-Computer and Information Sciences*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP)*.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. 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 3011–3021.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2022. Hegel: Hypergraph transformer for long document summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10167–10176.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2023. Extractive summarization via chatgpt for faithful summary generation. *arXiv preprint arXiv:2304.04193*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

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*, Online. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, Young-Suk Lee, Radu Florian, and Salim Roukos. 2021. Structure-aware fine-tuning of sequence-to-sequence transformers for transition-based AMR parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qingyu Zhou, Furu Wei, and Ming Zhou. 2020. At which level should we extract? an empirical analysis on extractive document summarization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5617–5628.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663.

A Implementation Details

Models are implemented using Pytorch (Paszke et al., 2019) and Huggingface transformers (Wolf et al., 2020). We initialized BART with *facebook/bart-base* and trained the model with AdamW (Loshchilov and Hutter, 2018). We set the learning rate to $3e-5$, gradient clipping to 0.1, and weight decay to 0.01. The learning rate was updated using a polynomial decay schedule with an end value of 0. We set the warm-up step to 1000, the total training steps to 40000, and the batch size to 14. During inference, we used Constrained Beam Search (Anderson et al., 2017) and restricted the max length for the input and the output sequence to be 768 and 512, respectively. We normalized the log probabilities by sequence length. The training was performed on 8 NVIDIA V100

GPUs and it took about 30 minutes for one training run.

B Prompt Design

The prompt utilized for querying ChatGPT is presented in Table 6. Different from the approach of Zhang et al. (2023), we omitted the specification of the number of sentences to be extracted. This deliberate exclusion facilitates a direct comparison with our proposed method under equivalent experimental conditions.

> The extractive summary consists of exact sentences from a given document, and those sentences can serve as the summary of the given document. Give the extractive summary for the following article, and represent the extracted sentences in a list format.
>
> {source document }

Table 6: The designed prompt template for querying ChatGPT to generate an extractive summary for a given source document.

C Case Study

We provided an example generated by both our proposed method and ChatGPT using the same source document in Table 7. The prompt employed for ChatGPT, as detailed in Table 6, intentionally omitted the specification of the number of sentences to be extracted. As discussed in Section 5.6, ChatGPT tends to select more sentences than the expectation, thereby trading off or sacrificing precision for recall. In contrast, our method adopts an IE-based approach that initially extracts fine-grained semantic tuples. Subsequently, source sentences are chosen to cover the extracted semantic tuples, which eliminates the a priori predetermination of the exact number of sentences for extraction.

Document	Tony Mowbray will be named as the new manager of Coventry on Tuesday. Mowbray has been out of work since leaving Middlesbrough in 2013 but has agreed to return to the dugout with the League One strugglers. Steven Pressley was sacked by the Sky Blues last week after a run of seven games without a win. Former Middlesbrough manager Tony Mowbray will take over as the new boss of struggling Coventry. Neil MacFarlane and Dave Hockaday took charge of Saturday's win over MK Dons and look set to remain in charge for Tuesday's game at Barnsley but Mowbray could be in the stands at Oakwell. Coventry are just above the relegation zone after their poor run of form and will look to Mowbray to turn things around quickly. Steven Pressley was sacked by the League One club after they became mired in a relegation battle.
Reference	Tony Mowbray will take over as manager of League One Coventry City. He will replace Steven Pressley who was sacked last month. Mowbray has been out of work since 2013 when he left Middlesbrough. He also had spells in charge of Hibs, West Brom and Celtic .
Output Semantic Tuples	Tony Mowbray<sep>been<sep>out of work<et>Tony Mowbray<sep>agreed<sep>to return to the dugout<sep>Tony Mowbray<sep>leaving<sep>Middlesbrough<et>Tony Mowbray <sep>take<sep>as the new boss of Coventry<et>Steven Pressley<sep>sacked<sep>by the Sky Blues<et>
Our Output Summary	Mowbray has been out of work since leaving Middlesbrough in 2013 but has agreed to return to the dugout with the League One strugglers. Former Middlesbrough manager Tony Mowbray will take over as the new boss of struggling Coventry. Steven Pressley was sacked by the Sky Blues last week after a run of seven games without a win.
ChatGPT	Tony Mowbray will be named as the new manager of Coventry on Tuesday. Mowbray has been out of work since leaving Middlesbrough in 2013. Steven Pressley was sacked by the Sky Blues last week after a run of seven games without a win. Former Middlesbrough manager Tony Mowbray will take over as the new boss of struggling Coventry. Neil MacFarlane and Dave Hockaday took charge of Saturday's win over MK Dons and look set to remain in charge for Tuesday's game at Barnsley. Mowbray could be in the stands at Oakwell. Coventry are just above the relegation zone after their poor run of form. The League One club sacked Steven Pressley as they became mired in a relegation battle.

Table 7: Case study of different settings