Generating and Evaluating Long Story Summaries with Knowledge Graphs

Anonymous ACL submission

Abstract

Summarizing long stories is a challenging task due to their narrative complexity and the context length limits of language models. We propose a method that integrates knowledge graph retrieval with the summarization process to provide global context. We construct a knowledge graph containing entity descriptions and relations from the entire story, then retrieve relevant information from it to aid summary generation. Additionally, we propose a novel metric, KGScore, which evaluates summaries by comparing the similarity of knowledge graphs extracted from generated and reference summaries. Experimental results demonstrate that our knowledge graph retrieval method outperforms the baselines in terms of our KGScore metric and that KGScore is a reliable measure of factual consistency.

1 Introduction

Language models based on the transformer architecture (Vaswani et al., 2017) have been successfully trained to summarize short texts (Liu and Lapata, 2019; Zhang et al., 2020a, 2022a). However, understanding and summarizing longer documents, such as entire books, remains a challenge, largely due to the context length limit imposed by the quadratic complexity of the attention mechanism (Cao and Wang, 2023). Furthermore, stories pose unique problems as their summaries are highly abstractive in nature and require the navigation of a mix of narration and dialogue, with complex dependencies interspersed throughout the text (Kryściński et al., 2021). The difficulty is amplified by the fact that narrative texts often employ the technique of "show, don't tell": instead of explicit descriptions or stating of facts, the author relies on implicit information conveyed through dialogue or character actions. As a result, long stories, with their dual hurdles of extensive length and narrative intricacy, present a particularly daunting task for summarization.

Previous research on the topic ranges from divide-and-conquer strategies that produce a summary of summaries from split-up story segments (Wu et al., 2021; Kashyap, 2022), to approaches that generate an abstractive summary of extractive samples (Hardy et al., 2022). The ability of these methods to produce factually consistent summaries is limited by the lack of a global context.

We propose the use of knowledge graphs to address this issue. Knowledge graphs represent descriptions of entities and relationships between them in a structured form and have been used to successfully improve performance on a variety of natural language generation tasks (Fan et al., 2019; Andrus et al., 2022). We frame the problem as a chapter summarization task, where the model is given the chapter text and a knowledge graph of the entire book. The model retrieves information from the knowledge graph to augment its understanding of the story and generate a chapter summary.

To generate the knowledge graph, we split the book text into small chunks and instruct a large language model to identify named entities in the text, which become the nodes of the graph. We then extract the graph edges by prompting the model to generate entity descriptions and relations. We additionally follow a series of steps to ensure that the information in the knowledge graph is accurate and relevant. During summarization, the knowledge graph edges are ranked based on their semantic similarity to a set of keywords, then retrieved and prepended to the chapter text in linearized form to be given to the summarization model.

We also propose a new metric for evaluating generated summaries, which we name KGScore. It is designed to address the limitations of existing metrics in evaluating factual consistency. The metric computes precision, recall, and F1 scores based on the cosine similarity of knowledge graph edge embeddings extracted from generated and reference summaries.

We evaluate the effectiveness of our approach through experiments on the BookSum Chapters dataset (Kryściński et al., 2021), which contains chapter texts and their summaries. We find that our proposed method outperforms the baseline in terms of our KGScore metric. Through an additional experiment, we also verify that KGScore is a valid measure of factuality.

2 Related Work

2.1 Long Document Summarization

Summarizing long documents using transformer-based language models is challenging because of the quadratic computational and memory requirements. Existing approaches to overcoming this problem can be broadly classified into three categories: divide and conquer, efficient attention, and extractive-abstractive summarization.

The divide-and-conquer strategy breaks down the task of summarizing a long document into smaller tasks of summarizing short sections of the document that can fit into a language model's context. Summaries for each section are combined to produce the summary for the full document (Gidiotis and Tsoumakas, 2020; Zhang et al., 2022b). This segmentation can result in reduced coherence due to a lack of global context. To address this problem, Cao and Wang (2023) introduce an external memory mechanism. Pang et al. (2023) propose a variant form of divide and conquer, where they combine a bottom-up pass using local self-attention on chunks of text with a top-down correction step to capture long-range dependencies.

There have also been efforts to improve the attention mechanism itself instead of working around its limitations, reducing the time and memory complexities to subquadratic levels for long sequences. This makes it feasible to fit long inputs into the model (Huang et al., 2021). On top of this, Phang et al. (2022) incorporate a pretraining step on long texts to further improve performance.

Extractive-abstractive summarization is a group of methods consisting of two steps: extracting relevant parts of the input document, then using a language model to generate an abstractive summary from the extracted snippets (Pilault et al., 2020; Zhao et al., 2020). Large language models such as OpenAI's ChatGPT have been used for the abstractive step (Lu et al., 2023).

Solutions for the more specific problem of summarizing long stories have also been explored. Wu et al. (2021) and Kashyap (2022) use techniques based on divide and conquer, while Hardy et al. (2022) propose an extractive-abstractive approach. Our method can be considered a form of divide and conquer; we additionally incorporate knowledge graphs as a way of providing global context.

2.2 Knowledge Graphs for Text Generation

Knowledge graphs can be used with text-generation tasks to supplement models with additional information. Here, we discuss knowledge graphs extracted on the fly from input documents. Prior research, such as ASGARD (Huang et al., 2020), typically employs graph attention to encode the graph data (Zhu et al., 2021; Chen et al., 2023). They focus on tasks that involve synthesizing information from multiple documents (Fan et al., 2019), including the summarization of multiple news articles (Lakshika et al., 2020).

The application of these methods has largely been confined to factual content, such as news articles or academic papers, and not stories. While the Stanford OpenIE system (Angeli et al., 2015) is a popular choice for extracting relational data from documents to construct knowledge graphs, rule-based systems like this often struggle to generate meaningful knowledge graphs from narrative texts. Andrus et al. (2022) do use the OpenIE system for story completion and question answering tasks, but integrate it with GPT-3 (Brown et al., 2020). In our approach, we forgo the OpenIE system entirely, directly using a large language model for knowledge graph construction.

2.3 Metrics for Summarization Evaluation

One of the most commonly used metrics for evaluating summaries is ROUGE (Lin, 2004), which measures the overlap of n-grams between generated and reference summaries. BERTScore (Zhang et al., 2020b) is another widely-used metric and involves computing the similarity of contextual embeddings. Many of these existing metrics have been shown to correlate poorly with human judgments of quality (Novikova et al., 2017), especially for assessing factuality (Maynez et al., 2020).

Methods have been proposed to evaluate the factual consistency of generated summaries (Kryscinski et al., 2020; Xie et al., 2021), including QuestEval (Scialom et al., 2021), which uses question generation and answering for this purpose. However, these methods are often impractical to use with long documents as they involve the use of models with limited context sizes. Our proposed metric bypasses this limitation by adopting a source-free approach that compares the generated summary with the reference summary instead of the original document.

Some recent metrics make use of large language models such as ChatGPT and GPT-4 (OpenAI, 2023), guiding a model through prompts to produce evaluations (Gao et al., 2023; Liu et al., 2023). While our metric also incorporates a large language model as part of the process, we do not rely on it fully; its use is limited to the knowledge graph extraction step.

Metrics specifically targeting summaries of long documents, including long stories, have also been proposed. LongDocFACTScore (Bishop et al., 2023) is a framework that enables the extension of any preexisting metric to accommodate long documents. SNaC (Goyal et al., 2022) and BooookScore (Chang et al., 2023) are both reference-free and source-free metrics that identify errors by focusing exclusively on the content of the generated summaries. Our metric is source-free but not reference-free; it compares predicted summaries with reference summaries.

3 Methods

3.1 Chapter Summarization Task

The task is formulated as follows. Given the full text of a book $\mathcal{B} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_n\}$ consisting of n chapters, and a chapter index k $(1 \leq k \leq n)$, the model $\mathbb{M}$ must learn to generate the chapter summary $\mathcal{Y}_k$ corresponding to the chapter text $\mathcal{C}_k$:

$$\mathbb{M} : (\mathcal{B}, k) \rightarrow \mathcal{Y}_k \ (1 \leq k \leq n) \qquad (1)$$

The model may utilize information from all parts of the book as needed. However, we assume that the full book text is too long to be given to the model as input in its original form, while individual chapters are not. This task can be considered the first half of a divide-and-conquer approach, where the chapters are segments of the book that can fit into the model's context size. The second half of the process would be to combine the generated summaries for each chapter into a summary for the full book, but we do not focus on that part here.

3.2 Summarization with Knowledge Graph Retrieval

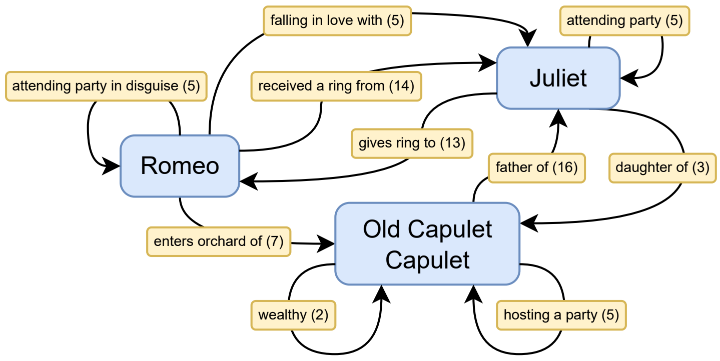
In our method, we generate a knowledge graph containing information from the entire book text. Each node in the graph represents a named entity in the story, such as a character, organization, or location, and each directed edge represents a <subject, predicate, object> triple (*e.g.*, <Romeo, is in love with, Juliet>), where the source node and target node correspond to the subject and object, respectively. As an exception, in self-loops (*i.e.*, edges with the same source and target node), the object is ignored and the edge represents a <subject, predicate> pair (an entity description or an action with no object; *e.g.*, <Romeo, is in love>). Multiple edges with different predicates can exist between a single (subject, object) pair. Figure 1a shows an example of a generated knowledge graph.

During training and inference, we retrieve information from the knowledge graph and provide it to the summarization model along with the full chapter text. This additional information acts as global context that is lacking when only the chapter text is provided. For example, if a character that was introduced in a previous chapter reappears in the current chapter, it could be difficult for the model to determine the identity of the character by only examining the current chapter. Information from previous chapters would be helpful global context that helps the model "remember" who the character was, and any details about the character in the generated summary would be more likely to be correct.

3.3 Book Knowledge Graph Generation

Knowledge Extraction We use a large language model to extract the knowledge graph nodes and edges from the book text. We split the text at paragraph boundaries so that each segment, when inserted into the prompt, fits into the model's context size. The prompt begins with an instruction to identify named entities and knowledge graph edges. This is followed by an example containing a story excerpt and lists of corresponding entities and edges. Finally, the book segment of interest is given as the task for the model. The complete prompt can be found in Appendix E. For our experiments, we use OpenAI's `gpt-3.5-turbo-0613` model.

Names Graph Before building the knowledge graph, we parse the named entities from the model



(a) Knowledge graph.

<subject>Romeo
<predicate>attending party in disguise
<object>Juliet
<predicate>falling in love with
<predicate>received a ring from
<object>Capulet
<predicate>enters orchard of
<subject>Juliet
<predicate>attending party
<object>Romeo
<predicate>gives ring to
<object>Capulet
<predicate>daughter of
<subject>Capulet
<predicate>hosting a party
<predicate>wealthy
<object>Juliet
<predicate>father of

(b) Linearized knowledge graph.

Figure 1: (a) Part of a knowledge graph generated from William Shakespeare's *Romeo and Juliet*, with three named entities. "Old Capulet/Capulet" is a single entity with two names. The numbers in parentheses following the predicates are chapter numbers. (b) The same graph in linearized form.

responses to generate a *names graph*. The model returns a list of names (aliases or name variations) for each entity that appears in a given story segment. The purpose of the names graph is to consolidate this information and keep track of names that refer to the same entity over the span of the entire book. For example, names *A* and *B* could be identified as aliases of an entity in one section of the book, while the same is done for names *B* and *C* in a different section; the names graph will indicate that names *A*, *B*, and *C* are all names of the same entity. A node is created for each distinct name, and undirected edges are created between nodes that represent names of the same entity. By repeating this process, we obtain a graph of all identified names in the book. If two nodes are connected (*i.e.*, there exists a path of edges between them), their names refer to the same entity.

Knowledge Graph Initialization We initialize the knowledge graph by giving each name in the names graph its own node in the knowledge graph, regardless of whether they belong to the same entity. We parse the knowledge graph edges from the model responses and perform a processing step to ensure that both the subject and object are named entities. If there is no object, we repeat the subject as the object to create a self-loop. Each processed edge is added to the knowledge graph as a directed edge from the subject node to the object node.

Node Merging At this point, the knowledge graph may contain multiple nodes representing the same entity under different names. In this step, we merge these nodes into one node to obtain a graph with one node per entity, with each node having a list of names for its entity. For each edge in the names graph (connecting two names that refer to a single entity), we identify the nodes in the knowledge graph that contain the names that the edge connects. If the names belong to two different nodes, we merge the nodes by combining their name lists and transferring the edges of one node to the other.

One problem is that the names graph occasionally contains incorrect connections (*i.e.*, edges between names of distinct entities). We employ two heuristics to prevent the merging of entity nodes when this occurs. First, if the two nodes have an edge between them (representing a <subject, predicate, object> triple), we do not merge the nodes. It is unlikely that a triple would have the same entity as the subject and object while also referring to it using different names. Second, we check the degree (the number of edges entering or leaving the node) of each of the two nodes, excluding any self-loops from the count. We do not merge the nodes if both edge counts exceed a maximum threshold value. The reasoning is that if both nodes have a high degree, they are both important entities and are likely to be different. In our experiments, we set the threshold to 3.

Node Removal The final step is to remove nodes in the knowledge graph with a degree that is less than a minimum threshold value (excluding self-loops). This ensures that only the most relevant entities remain in the graph. This can also eliminate erroneously identified entities. The removal of a node and its edges can affect the edge counts of other nodes, so we repeat the process of counting and removing until no more nodes are removed. We use a minimum degree of 2 in our experiments.

3.4 Knowledge Graph Edge Retrieval

Edge Ranking Instead of providing the entire knowledge graph to the model with the chapter text, we retrieve a subset of the graph that would be the most helpful for generating a summary. We experimentally select a set of keywords and corresponding weights to score and rank the knowledge graph edges. Each keyword is designed to focus on an important narrative aspect, such as character relationships (with the "relation" keyword) and events ("happen"). The full set of keywords and weights can be found in Appendix A.

For a book knowledge graph G and the set of nodes N_k of entities mentioned in the current chapter text $\mathcal{C}_k$, we obtain the induced subgraph $G[N_k]$ that only contains the nodes in N_k and edges between them. We further judge that future information is mostly unnecessary for generating a chapter summary and remove edges from $G[N_k]$ that were extracted from later in the story than the current chapter. We define the remaining set of edges as E_k.

We use Sentence-BERT (Reimers and Gurevych, 2019) to compute similarity scores between embeddings for each keyword in the set of keywords Q and the predicate portion of each <subject, predicate, object> edge in E_k. Let $s_{ij} = \text{cos\_sim}(p_i, q_j)$ be the cosine similarity score between the embedding of the i-th predicate p_i $(1 \leq i \leq |E_k|)$ and the embedding of the j-th keyword q_j $(1 \leq j \leq |Q|)$. Then the normalized similarity score $\tilde{s}_{ij}$ is computed as:

$$\tilde{s}_{ij} = \frac{s_{ij} - \mu_j}{\sigma_j} \tag{2}$$

where μ_j and σ_j are the mean and standard deviation, respectively, of the scores for the j-th keyword across all edges. This normalization is required to fairly weight the scores, as some keywords may have generally higher or lower scores than others. The weighted score W_{ij} for each edge and keyword is then:

$$W_{ij} = \tilde{s}_{ij} \cdot w_j \tag{3}$$

where w_j is the weight associated with the j-th keyword. The final aggregated score S_i for the i-th edge is the sum of its weighted scores across all keywords:

$$S_i = \sum_j W_{ij} \tag{4}$$

This score is used to rank the edges in E_k.

Edge Linearization We provide the retrieved knowledge graph edges to the summarization model as a linearized string of text prepended before the chapter text. Starting from the highest-ranked edge (*i.e.*, the edge with the highest score), we gather edges in E_k until the context length limit is reached, taking into account the length of the chapter text. We then arrange the gathered edges by their subject entities so that edges with a common subject are grouped together. In each group, we categorize the edges by their objects. Additionally, we sort the subjects and objects by the total number of appearances in the chapter text in decreasing order. This puts entities important to the chapter near the front of the linearized text. Self-loops, which are treated as edges with no object, are placed before other edges with the same subject.

The linearization format depends on the level of access that is available for the summarization model. For models that are able to be finetuned for the task, we use a format that includes three new special tokens: `<subject>`, `<object>`, and `<predicate>`. The model is expected to learn the meaning of these tokens during the finetuning process. Each group of edges sharing a subject is preceded by the `<subject>` token and the name of the subject. If the entity has multiple names, the name that appears the most frequently in the chapter text is used. Inside each group, every new object is marked by the `<object>` token and the object name (again, the most commonly used name). This part is omitted for edges with no object. Finally, a `<predicate>` token and the predicate text are appended for each predicate. An example is shown in Figure 1b. The input to the summarization model is the string of linearized edges, followed by an additional `<chapter>` special token and the chapter text.

If the summarization model is one that must be used without finetuning (*e.g.*, a model that can only be accessed through an inference API), the linearization format simply consists of each edge on its own line, with the subject, predicate, and object separated by a single space and semicolon (*e.g.*, `Juliet; gives ring to; Romeo`); no additional special tokens are used. The subjects and objects for each edge are always specified, even when they are the same as those in the previous edge. The intent is to present the edge information in a format that the model can understand without further training.

3.5 Knowledge Graph Similarity Metric (KGScore)

Existing metrics such as ROUGE and BERTScore may not be good measures of the quality of abstractive summaries (Novikova et al., 2017), especially when it comes to factuality and faithfulness to the original text (Maynez et al., 2020). As we believe the main role of the knowledge graphs is to provide global context and enhance factual consistency, a metric that explicitly checks for adherence to established facts would be helpful for evaluating the effect of the knowledge graphs. For stories, many of these facts have to do with character descriptions, actions taken by characters, and relationships between them, all of which can be represented in a knowledge graph.

We propose a novel metric that we call KGScore, which measures the quality of a summary by computing the similarity between two knowledge graphs. Let G_g and G_r represent knowledge graphs extracted from a generated and reference summary, respectively. They contain edges E_g and E_r. For an edge $e_g \in E_g$ with subject s_{e_g} and object o_{e_g}, let E_{r,e_g} be the subset of edges in E_r that have the same subject s_{e_r} and object o_{e_r} as e_g:

$$E_{r,e_g} = \{e_r \in E_r \mid s_{e_r} = s_{e_g} \wedge o_{e_r} = o_{e_g}\} \quad (5)$$

E'_g is the subset of edges in E_g for which E_{r,e_g} is not empty:

$$E'_g = \{e_g \in E_g \mid E_{r,e_g} \neq \varnothing\} \quad (6)$$

Then the precision KGScore P_{KG} is defined as follows:

$$P_{\text{KG}} = \frac{1}{|E_g|} \sum_{e_g \in E'_g} \max_{e_r \in E_{r,e_g}} \text{cos\_sim}(p_{e_g}, p_{e_r}) \quad (7)$$

where $\text{cos\_sim}(p_{e_g}, p_{e_r})$ is the cosine similarity between the embeddings of the predicates of e_g and e_r. The precision score is roughly equivalent to a measure of the proportion of the information in the generated summary graph that also exists in the reference summary graph. For the recall score R_{KG}, the direction is reversed:

$$R_{\text{KG}} = \frac{1}{|E_r|} \sum_{e_r \in E'_r} \max_{e_g \in E_{g,e_r}} \text{cos\_sim}(p_{e_g}, p_{e_r}) \quad (8)$$

where E_{g,e_r} and E'_r are defined in the same way as E_{r,e_g} and E'_g, but with the roles of the two graphs swapped. Finally, the F1 score F_{KG} is the harmonic mean of the precision and recall scores:

$$F_{\text{KG}} = \frac{2 \cdot P_{\text{KG}} \cdot R_{\text{KG}}}{P_{\text{KG}} + R_{\text{KG}}} \quad (9)$$

To generate the two knowledge graphs G_g and G_r, we follow a process similar to the one for book knowledge graph generation, with some modifications. Instead of using a single prompt and model to identify both named entities and knowledge graph edges, we split the process into two steps.

First, we use spaCy (Montani et al., 2023) to find named entities in the reference summary (we use version 3.7.3 of the `en-core-web-trf` pipeline). This is faster than the previous approach of using a large language model, but it is limited in that it cannot identify aliases or name variations. We consider this an acceptable compromise, as summaries are generally short and it is unlikely that multiple names are used for a single character. Consequently, we skip the steps of generating a names graph and merging nodes that represent the same entity.

For the next part of the process, we opt for a locally run model instead of an OpenAI model and pair it with the Guidance library[1] to constrain the model output to text that can be parsed into valid knowledge graph edges. This is to increase accuracy and lower costs at the expense of longer generation times. More specifically, we use Mixtral 8x7B (Jiang et al., 2024), a mixture-of-experts model with 13 billion active parameters. The prompt includes the named entities from the reference summary, as well as three few-shot examples of extracting edges from summaries, formatted as a multi-turn conversation. The full prompt is in Appendix F. We use the same entities for both the reference and generated summaries to maximize the overlap between entities in the two sets of edges, which is important for computing reliable KGScore values. We omit the original final step of removing nodes with few edges because the number of edges is generally small for a summary.

[1] https://github.com/guidance-ai/guidance

4 Experiments

4.1 Dataset

For our experiments, we choose the BookSum Chapters dataset (Kryściński et al., 2021), which contains chapter texts and their summaries from over 200 English-language books, including novels, plays, and short stories. We filter the dataset to improve its quality and better align it with our purposes, and we are left with 7255 examples in the training set and 1155 examples in the validation set; details are in Appendix B.

4.2 Effectiveness of Knowledge Graph Retrieval

To verify the effectiveness of our knowledge graph retrieval method, we finetune two LongT5 (Guo et al., 2022) models for summarization on the filtered BookSum Chapters dataset: a baseline model trained with only the chapter texts as input (LongT5-No-KG) and a model trained using our proposed method (LongT5-KG). Finetuning details and parameters are included in Appendix C.

We additionally apply our method to OpenAI's `gpt-4-1106-preview` model with no finetuning, using the simple edge linearization format as described in Section 3.4. We employ Chain of Density (Adams et al., 2023), an iterative prompting technique that produces entity-dense summaries, to maximize the effect of the entity information contained in the knowledge graph edges. We label the baseline results without our knowledge graph retrieval method as GPT-4-No-KG and the results from our method as GPT-4-KG.

The results are summarized in Table 1. Due to resource constraints, we randomly select 100 chapters from the test set of the BookSum Chapters dataset and report evaluation results on this smaller subset. Results for three sets of metrics are included: ROUGE, BERTScore, and our proposed KGScore. Details on the evaluation procedure are in Appendix D. For the LongT5 models, the model using our knowledge graph retrieval method (LongT5-KG) achieves higher scores across all metrics compared to its baseline counterpart (LongT5-No-KG). For the GPT-4 results, our method outperforms the baseline in terms of KGScore while receiving slightly lower scores for ROUGE and BERTScore.

4.3 Validity of KGScore

We perform an additional experiment to verify the hypothesis that our proposed KGScore metric is a valid measure of factual consistency. We filter the training set of the BookSum Chapters dataset for summaries whose word counts fall within the range of 300 to 450 words, then gather pairs of summaries of the same book chapter. We select the 100 most similar summary pairs and use them as the baseline dataset for our experiment. Although all of these summaries are human written, we randomly select one summary from each pair as a "prediction" ("generated") summary and the other as a "reference" summary for the purpose of calculating evaluation metrics.

Next, we create a modified version of the dataset by identifying the named entities in each prediction summary using spaCy (Montani et al., 2023) and shuffling their locations, ensuring that entities only get swapped with other entities of the same type (*e.g.*, person or location). Much of the factual information included in this new summary is likely to be inaccurate. While the baseline dataset consists of pairs of summaries containing similar information (as they are summaries of the same chapter), the altered prediction summaries in the entity-shuffled dataset factually deviate from the reference summaries. Therefore, a reliable factuality metric should produce a significantly lower score for the entity-shuffled dataset in comparison to the baseline dataset.

The results of evaluating the two datasets on ROUGE, BERTScore, and KGScore are shown in Table 2. The decrease in metric values from the baseline to the entity-shuffled dataset is substantially greater in KGScore compared to ROUGE and BERTScore, which exhibit relatively small reductions.

5 Analysis

Combining and examining the results of the experiment in Section 4.2, where our knowledge graph retrieval method attains better KGScore results than the baselines, and the results of the entity shuffle experiment in Section 4.3, which show that KGScore is significantly more sensitive to variations in factual accuracy than ROUGE and BERTScore, we claim that our method successfully improves factual consistency in summaries as intended. This improvement may not always be detectable through traditional metrics, as evident in the GPT-4 re-

Model	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore F1	P_{KG}	R_{KG}	F_{KG}
LongT5-No-KG	28.80	5.48	13.96	53.82	22.54	15.69	17.73
LongT5-KG	**30.07**	**5.91**	**14.51**	**54.58**	**23.07**	**16.59**	**18.34**
GPT-4-No-KG	**25.06**	**3.76**	**14.03**	**56.28**	23.26	18.67	20.00
GPT-4-KG	24.14	3.60	13.76	55.96	**25.57**	**20.17**	**21.75**

Table 1: Evaluation results on a subset of the BookSum Chapters test set. P_{KG}, R_{KG}, and F_{KG} are average KGScore values as defined in Section 3.5. The best scores among each model category (LongT5 and GPT-4) are in bold.

	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore F1	P_{KG}	R_{KG}	F_{KG}
Baseline	50.45	13.15	23.51	63.62	26.03	24.67	24.93
Shuffled entities	50.34	11.79	21.59	60.44	16.10	14.33	14.78
Change (%)	-0.2	-10.3	-8.1	-5.0	**-38.1**	**-41.9**	**-40.7**

Table 2: Results of the entity shuffle experiment. P_{KG}, R_{KG}, and F_{KG} are average KGScore values as defined in Section 3.5. Changes in metric values after the entity shuffling are shown, with KGScore results in bold.

sults, where the application of our method leads to slightly worse ROUGE and BERTScore values than the baseline.

5.1 KGScore Trends

In Tables 1 and 2, the precision KGScore (P_{KG}) is higher than the recall KGScore (R_{KG}) in all cases. This is because the named entities are identified from the reference summary and used for extracting knowledge graph edges in both the reference and generated summaries, as described in Section 3.5. All entities contained in the edges can be found in the reference texts, while some are missing in the generated texts. This imbalance could be removed by gathering named entities from both summaries, but this could potentially introduce incorrectly hallucinated entities from the generated summaries.

Another related observation is that the KGScore values are low overall, ranging in the 10s and 20s out of a theoretical maximum of 100 (%). This could also be the symptom of an entity-matching problem, as a single entity may sometimes appear under different names in the prediction and reference summaries. Adding a step in the metric computation process to identify these cases could help, but if overly eager deductions are made about which names refer to the same entity, a new problem could arise where entities that should be kept separate are merged into one.

5.2 Qualitative Evaluation

To verify the interpretation that our knowledge graph retrieval method improves factual consistency among entities, we qualitatively evaluate a small sample of generated chapter summaries. Examples of these summaries are in Appendix G.

For the finetuned LongT5 models, we find that the summaries generated by both the LongT5-No-KG model and the LongT5-KG model are of low quality. They are similar to extractive summaries, repeating large sections of story text verbatim, and they often contain errors. This basic deficiency in summarization performance makes it difficult to determine the effects of using our method.

In comparison, we perceive the GPT-4 summaries to be higher-quality, which is corroborated by the higher KGScore values in Table 1. It is interesting to note that the ROUGE-1 and ROUGE-2 scores are much lower than those of the LongT5 models; this could be another indication that ROUGE does not align well with human judgments.

Although GPT-4-KG receives better KGScore results than GPT-4-No-KG, we find it challenging to identify specific examples of retrieved knowledge graph edges improving summary quality. It could be that the degree of improvement in this case is too subtle for humans to readily notice.

6 Conclusion

We present a method of summarizing long stories that employs knowledge graph retrieval to provide global context. We also propose a novel metric, KGScore, which evaluates summaries based on knowledge graph similarity. Experimental results indicate that our approach may enhance factual consistency and that our KGScore metric is an effective measure of it.

Limitations

Besides the improvement in KGScore, we have not presented additional evidence that our knowledge graph retrieval method enhances summary quality, such as a large-scale human evaluation.

The LongT5 model chosen for finetuning is a relatively old model with limited performance, as described in Section 5.2. More recent models, such as LLaMA 2 (Touvron et al., 2023), could be better suited for future experiments.

It can be argued that the observed score improvements using our method are not very significant. To achieve greater improvements, methods to more effectively provide information to the summarization model through the knowledge graph edges could be explored. For example, providing temporal information along with the edges, either in the form of chapter numbers or relative positions in the story, could be beneficial. The model can then gain a sense of time by placing events and details in relation to each other and the current chapter. This can be especially helpful if significant changes occur to an entity over the course of the story (*e.g.*, a death or a relocation). Another potential point of enhancement could lie in the way the retrieved edges are provided to the model. For instance, instead of simply prepending a linearized string to the context, a graph neural network could be used to process the knowledge graph information.

Ethics Statement

We do not anticipate any ethical issues. Our work deals with producing summaries of existing stories, so malicious applications would be limited, although the generated summaries could be inaccurate or reflect harmful content in the source text.

All tools used are open source, including the scripts to download the BookSum Chapters dataset, which are released under the BSD 3-Clause License. Considering the nature of the dataset, which consists of book texts and their summaries, its content was not inspected for offensive material or personally identifiable information. Where specified, usage of tools and datasets was done in accordance with their intended use.

References

Griffin Adams, Alex Fabbri, Faisal Ladhak, Eric Lehman, and Noémie Elhadad. 2023. From Sparse to Dense: GPT-4 Summarization with Chain of Density Prompting. In *Proceedings of the 4th New Frontiers in Summarization Workshop*, pages 68–74.

Berkeley R. Andrus, Yeganeh Nasiri, Shilong Cui, Benjamin Cullen, and Nancy Fulda. 2022. Enhanced Story Comprehension for Large Language Models through Dynamic Document-Based Knowledge Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10436–10444.

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.

Jennifer A. Bishop, Qianqian Xie, and Sophia Ananiadou. 2023. LongDocFACTScore: Evaluating the Factuality of Long Document Abstractive Summarisation. *arXiv preprint arXiv:2309.12455*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Shuyang Cao and Lu Wang. 2023. AWESOME: GPU Memory-constrained Long Document Summarization using Memory Mechanism and Global Salient Content. *arXiv preprint arXiv:2305.14806*.

Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2023. BooookScore: A systematic exploration of book-length summarization in the era of LLMs. *arXiv preprint arXiv:2310.00785*.

Tong Chen, Xuewei Wang, Tianwei Yue, Xiaoyu Bai, Cindy X. Le, and Wenping Wang. 2023. Enhancing Abstractive Summarization with Extracted Knowledge Graphs and Multi-Source Transformers. *Applied Sciences*, 13(13):7753.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019. Using Local Knowledge Graph Construction to Scale Seq2Seq Models to Multi-Document Inputs. 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 4186–4196.

Mingqi Gao, Jie Ruan, Renliang Sun, Xunjian Yin, Shiping Yang, and Xiaojun Wan. 2023. Human-like Summarization Evaluation with ChatGPT. *arXiv preprint arXiv:2304.02554*.

Alexios Gidiotis and Grigorios Tsoumakas. 2020. A Divide-and-Conquer Approach to the Summarization of Long Documents. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:3029–3040.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. SNaC: Coherence Error Detection for Narrative Summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 444–463.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient Text-To-Text Transformer for Long Sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736.

Hardy Hardy, Miguel Ballesteros, Faisal Ladhak, Muhammad Khalifa, Vittorio Castelli, and Kathleen McKeown. 2022. Novel Chapter Abstractive Summarization using Spinal Tree Aware Sub-Sentential Content Selection. *arXiv preprint arXiv:2211.04903*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient Attentions for Long Document Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436.

Luyang Huang, Lingfei Wu, and Lu Wang. 2020. Knowledge Graph-Augmented Abstractive Summarization with Semantic-Driven Cloze Reward. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5094–5107.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. *arXiv preprint arXiv:2401.04088*.

Prerna Kashyap. 2022. COLING 2022 Shared Task: LED Finteuning and Recursive Summary Generation for Automatic Summarization of Chapters from Novels. In *Proceedings of The Workshop on Automatic Summarization for Creative Writing*, pages 19–23.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the Factual Consistency of Abstractive Text Summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. BookSum: A Collection of Datasets for Long-form Narrative Summarization. *arXiv preprint arXiv:2105.08209*.

M. V. P. T. Lakshika, H. A. Caldera, and W. V. Welgama. 2020. Abstractive Web News Summarization Using Knowledge Graphs. In *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 300–301.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. *arXiv preprint arXiv:2303.16634*.

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.

Guang Lu, Sylvia B. Larcher, and Tu Tran. 2023. Hybrid Long Document Summarization using C2F-FAR and ChatGPT: A Practical Study. *arXiv preprint arXiv:2306.01169*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.

Ines Montani, Matthew Honnibal, Matthew Honnibal, Adriane Boyd, Sofie Van Landeghem, and Henning Peters. 2023. spaCy: Industrial-strength Natural Language Processing in Python.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why We Need New Evaluation Metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252.

OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

Bo Pang, Erik Nijkamp, Wojciech Kryscinski, Silvio Savarese, Yingbo Zhou, and Caiming Xiong. 2023. Long Document Summarization with Top-down and Bottom-up Inference. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1267–1284.

Jason Phang, Yao Zhao, and Peter J. Liu. 2022. Investigating Efficiently Extending Transformers for Long Input Summarization. *arXiv preprint arXiv:2208.04347*.

Jonathan Pilault, Raymond Li, Sandeep Subramanian, and Chris Pal. 2020. On Extractive and Abstractive Neural Document Summarization with Transformer Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9308–9319.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. 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 3982–3992.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization Asks for Fact-based Evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 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. In *Advances in Neural Information Processing Systems*, volume 30.

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*, pages 38–45.

Jeff Wu, Long Ouyang, Daniel M. Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. 2021. Recursively Summarizing Books with Human Feedback. *arXiv preprint arXiv:2109.10862*.

Yuexiang Xie, Fei Sun, Yang Deng, Yaliang Li, and Bolin Ding. 2021. Factual Consistency Evaluation for Text Summarization via Counterfactual Estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 100–110.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *ICML'20*, pages 11328–11339.

Mengli Zhang, Gang Zhou, Wanting Yu, Ningbo Huang, and Wenfen Liu. 2022a. A Comprehensive Survey of Abstractive Text Summarization Based on Deep Learning. *Computational Intelligence and Neuroscience*, 2022:e7132226.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. BERTScore: Evaluating Text Generation with BERT. *arXiv preprint arXiv:1904.09675*.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022b. Summ^N: A Multi-Stage Summarization Framework for Long Input Dialogues and Documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1592–1604.

Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. SEAL: Segment-wise Extractive-Abstractive Long-form Text Summarization. *arXiv preprint arXiv:2006.10213*.

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2021. Enhancing Factual Consistency of Abstractive Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 718–733.

A Keywords for Edge Retrieval

Table 3 lists the keywords and weights for knowledge graph edge retrieval. The weight for each keyword is empirically chosen based on the perceived effectiveness at retrieving useful edges.

Keyword	Weight
relation	30
happen	15
conflict	10
desire	10
emotion	10
role	10
think	5
location	5
personality	5

Table 3: Keywords and weights for knowledge graph edge retrieval.

B BookSum Chapters Dataset Filtering

We remove books that are found in both the training set and either the validation or test sets (book IDs 1130, 1526, 1783, and 1798). We also exclude books that are not narrative texts (61, 1232, 1404, 3207, 3420, 3755, 4320, 7370, 11224, 13434, and 34901) and books that are collections of multiple stories (221, 416, 610, 1429, and 15859). For some summaries, the script provided to build the BookSum dataset either fails to download the summary (because it is no longer available on the web) or downloads an incomplete version of it (empty or just a few words); we do not use those. Finally, in order to accommodate the context length limit of our model, we set chapter length and summary length limits of 16384 and 1024 tokens, respectively, and only use text pairs that fall under both limits.

C LongT5 Finetuning

We use the Hugging Face Transformers library (Wolf et al., 2020) and begin finetuning from the pretrained `long-t5-tglobal-base` checkpoint[2], which has 250 million parameters. For computing similarity scores with Sentence-BERT, we use the `all-MiniLM-L6-v2` model[3]. We train on 8 NVIDIA A100 GPUs for approximately 70 hours total for the two models, using the following parameters: Adam optimizer (Kingma and Ba, 2015), cosine learning rate scheduling with 2e-4 maximum learning rate and 1 epoch linear warm-up, batch size 128, 48 epochs. The hyperparameters were selected manually without tuning due to resource constraints.

[2]https://huggingface.co/google/long-t5-tglobal-base

[3]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

D Evaluation

For the finetuned LongT5 models, we evaluate the checkpoints with the lowest validation losses during training, with 4 beams and `no_repeat_ngram_size` set to 3. We use version 0.1.2 of the `rouge-score` Python package[4] and version 0.3.13 of the `bert-score` package[5] for ROUGE and BERTScore, respectively, with `microsoft/deberta-xlarge-mnli`[6] as the BERTScore model.

[4]https://pypi.org/project/rouge-score

[5]https://pypi.org/project/bert-score

[6]https://huggingface.co/microsoft/deberta-xlarge-mnli

E Prompt for Book Knowledge Graph Generation

Read part of a story, then identify named entities and generate knowledge graph edges.

[Begin story excerpt]
"Christmas won't be Christmas without any presents," grumbled Jo. "It's so dreadful to be poor!" sighed Meg, looking out the window at the snow-covered streets of Concord. "I don't think it's fair for some girls to have plenty of pretty things, and other girls nothing at all," added little Amy, with an injured sniff. "We've got Father and Mother, and each other," said Beth contentedly from her corner. The four young faces brightened at the cheerful words, but darkened again as Jo said sadly, "We haven't got Father, and shall not have him for a long time." She didn't say "perhaps never," but each silently added it, thinking of Father far away, where the fighting was.
As young readers like to know 'how people look', we will take this moment to give them a little sketch of the four sisters. Margaret March, the eldest of the four, was sixteen, and very pretty, with large eyes, plenty of soft brown hair, a sweet mouth, and white hands. Fifteen-year-old Jo March was very tall, thin, and brown, and never seemed to know what to do with her long limbs. Elizabeth, or Beth, as everyone called her, was a rosy, smooth-haired, bright-eyed girl of thirteen, with a shy manner, a timid voice, and a peaceful expression which was seldom disturbed. Amy, the youngest, was a regular snow maiden, with blue eyes, and yellow hair curling on her shoulders.
The clock struck six and, having swept up the hearth, Beth put a pair of slippers down to warm. Somehow the sight of the old shoes had a good effect upon the girls, for Mother was coming, and everyone brightened to welcome her. Jo sat up to hold the slippers nearer to the blaze. "They are quite worn out. Marmee must have a new pair." "I thought I'd get her some with my dollar," said Beth. "No, I shall!" cried Amy. "I'll tell you what we'll do," said Beth, "let's each get her something for Christmas, and not get anything for ourselves." "Let Marmee think we are getting things for ourselves, and then surprise her. We must go shopping tomorrow afternoon," said Jo, marching up and down.
"Glad to find you so merry, my girls," said a cheery voice at the door, and the girls turned to welcome a tall, motherly lady. She was not elegantly dressed, but the girls thought the gray cloak and unfashionable bonnet covered the most splendid mother in the world. As they gathered about the table, Mrs. March said, with a particularly happy face, "I've got a treat for you after supper." A quick, bright smile went round like a streak of sunshine. Beth clapped her hands, and Jo tossed up her napkin, crying, "A letter! A letter! Three cheers for Father!" "Yes, a nice long letter. He is well, and he sends all sorts of loving wishes for Christmas, and an especial message to you girls," said Mrs. March, patting her pocket as if she had got a treasure there. "I think it was so splendid in Father to go as chaplain when he was too old to be drafted, and not strong enough for a soldier," said Meg warmly, proud of her father's work with the Union Army.
[End story excerpt]

Named entities (include all aliases and name variations):
Jo / Jo March
Meg / Margaret / Margaret March
Amy
Beth / Elizabeth
March sisters
Mrs. March / Marmee / Mother
Father
Concord
Union Army

Knowledge graph edges (select up to 15 most important, `subject(s); predicate; [object(s)]` format, named entities only, predicate: five words max):
1. Jo, Meg, Amy, Beth; in; March sisters
2. March sisters; daughters of; Mrs. March, Father

3. Mrs. March; mother of; March sisters
4. Father; father of; March sisters
5. March sisters, Mrs. March; living in; Concord
6. Father; away fighting in war
7. Father; chaplain in; Union Army
8. Meg; sixteen years old
9. Jo; fifteen years old
10. Beth; thirteen years old
11. Beth; shy
12. Amy; youngest among; March sisters
13. March sisters; complained about not getting presents
14. March sisters; decided to buy presents for; Mrs. March
15. Mrs. March; brought home a letter from; Father

[Begin story excerpt]
{story excerpt}
[End story excerpt]

F Prompt for Summary Knowledge Graph Generation

Your task is to create a list of knowledge graph edges from a chapter summary. Here are the rules you must follow:
- A knowledge graph edge is in the format "<subject(s)>; [None] or <object(s)>; <predicate>".
- Subjects and objects must be named entities.
- If there are multiple subjects or objects, separate them with commas.
- Predicates describe a relationship or action between the subjects and objects.
- Predicates should not contain names.
- Keep the predicates short (four words max).
- If the knowledge graph edge is a description or action of a single entity with no object, use "[None]" in place of the object(s).
- Only include information explicitly provided in the summary.
- Find a diverse set of edges, never repeating similar edges.
- Order the edges by importance rather than appearance in the summary, with the most important edges first.

Summary:
The trio continues their arduous journey toward Canyon B, 300 km east of their home colony. Tensions rise as Janek and AC-293 clash over the best route to take. The Narrator, who has been silent for the past few days, finally speaks up and suggests that they take a shortcut through the mountains. Janek and AC-293 are skeptical, but they agree to try it. The Narrator leads them through a narrow pass, and they soon find themselves in an open valley. The Martian landscape is breathtaking, and the trio sets up camp for the night. The next morning, they continue their journey through the valley. Suddenly, they hear a loud noise and see a cloud of dust rising in the distance. They realize that a sandstorm is approaching, and they must find shelter quickly. They spot a cave in the distance and rush toward it. In the haste, Janek trips and falls, injuring his leg. The Narrator and AC-293 help him to his feet, and they make it to the cave just in time. They wait out the storm and emerge the next morning to find that their rover has been buried in sand. They dig it out and continue their journey toward Canyon B.

- Narrator, Janek, AC-293; Canyon B; continues journey toward
- Janek; AC-293; argues with
- AC-293; Janek; argues with
- Narrator; [None]; suggests shortcut through mountains
- Narrator, Janek, AC-293; [None]; camps in valley

- Narrator, Janek, AC-293; [None]; notices approaching sandstorm
- Narrator, Janek, AC-293; [None]; rushes toward cave
- Janek; [None]; falls and injures leg
- Narrator, AC-293; Janek; helps to feet
- Narrator, Janek, AC-293; [None]; waits out storm
- Narrator, Janek, AC-293; [None]; digs out rover
- Janek, AC-293; [None]; skeptical of shortcut
- Janek, AC-293; [None]; agrees to shortcut
- Narrator; [None]; leads through narrow pass
- Narrator, Janek, AC-293; [None]; makes it to cave
- Canyon B; [None]; east of home colony

Summary:
This chapter focuses on the dazzling Grand Acorn Gala at Furrington Grove. Sir Reginald, the host of the gala, is a wealthy hedgehog who has a reputation for being a bit of a snob. He is also a collector of rare artifacts, and he has invited the guests to bring their own treasures to the gala. Xander, a young fox, and his father, Yorick, are among the guests. Yorick is a famous explorer, and he has brought a rare artifact to the gala, a golden leaf. Xander is eager to show off his father's treasure, but when he approaches Sir Reginald, the host dismisses the golden leaf as common and uninteresting. Undeterred, Xander explores the gala and encounters Penelope, a wise owl, who shares a secret about the golden leaf: it was once part of a magical tree that grew in the forest. Xander is intrigued by the story, and he decides to investigate further. He discovers that the tree was destroyed by a terrible storm, and the golden leaf was the only thing that survived. Xander is determined to find the tree and restore it to its former glory.

- Sir Reginald; Grand Acorn Gala; host of
- Sir Reginald; [None]; artifact collector
- Xander; [None]; young fox
- Yorick; Xander; father of
- Yorick; [None]; brings golden leaf
- Sir Reginald; [None]; dismisses golden leaf
- Penelope; Xander; shares secret with
- Xander; [None]; investigates golden leaf
- Xander; [None]; wants to restore tree
- Yorick; [None]; famous explorer
- Xander; Sir Reginald; approaches
- Xander; Penelope; encounters
- Penelope; [None]; wise owl
- Sir Reginald; [None]; wealthy
- Sir Reginald; [None]; snob
- Grand Acorn Gala; Furrington Grove; takes place at

Summary:
In this chapter, Amelia, a seasoned investigator with the Justice & Integrity Taskforce, dives into a high-stakes art theft case at the Metropolitan Museum of Art in New York City. The missing painting leads her to Marco Santos, an enigmatic artist tied to the Paris Art Collective. Crossing the Atlantic, Amelia unravels a web of clues in the underground European art scene, exposing the ruthless European Art Syndicate behind the theft. As Amelia delves deeper, she unearths Marco's troubled past, connecting him to his controversial exhibitions at the Louvre and the Tate Modern. An unexpected alliance forms between them as they race to unveil the truth behind the theft. They are not the sole seekers of the stolen masterpiece, however. Dr. Harold Blackwood, CEO of Blackwood Enterprises, lurks in the shadows, manipulating the chaos for his personal gain. With a cunning mind and powerful connections, Blackwood poses a hidden threat. His motives remain veiled in mystery as he holds secret meetings in Paris, London,

and New York. Amelia and Marco navigate a treacherous path, not only against the European Art Syndicate but also against the calculated machinations of Dr. Harold Blackwood, who seeks to outsmart them and claim the stolen artwork for himself.

- Amelia; Metropolitan Museum of Art; investigates theft at
- European Art Syndicate; [None]; behind theft
- Amelia; European Art Syndicate; exposes
- Amelia; Marco Santos; forms alliance with
- Marco Santos; Amelia; forms alliance with
- Amelia, Marco Santos; [None]; races for truth
- Dr. Harold Blackwood; Amelia, Marco Santos; tries to outsmart
- Dr. Harold Blackwood; [None]; seeks stolen artwork
- Amelia; Justice & Integrity Taskforce; investigator at
- Marco Santos; [None]; enigmatic artist
- Marco Santos; Paris Art Collective; has ties to
- Amelia; Marco Santos; discovers past of
- Marco Santos; Louvre, Tate Modern; held controversial exhibition at
- European Art Syndicate; [None]; ruthless
- Dr. Harold Blackwood; Blackwood Enterprises; CEO of
- Dr. Harold Blackwood; [None]; cunning
- Dr. Harold Blackwood; Paris, London, New York; holds meetings in
- Metropolitan Museum of Art; New York City; located in

Summary:
{summary}

G Examples of Generated Summaries

The following are summaries generated from Chapter 8 of *Tess of the d'Urbervilles* by Thomas Hardy. KGScore F1 values (F_{KG}) are also provided. The reference summary is omitted due to copyright.

LongT5-No-KG ($F_{\text{KG}} = 35.54$):
Having mounted beside her, Alec d'Urberville drove rapidly along the crest of the first hill, chatting compliments to Tess as they went, the cart with her box being left far behind. Rising still, an immense landscape stretched around them on every side; behind, the green valley of her birth, before, a gray country of which she knew nothing except from her first brief visit to Trantridge.

LongT5-KG ($F_{\text{KG}} = 35.51$):
D'Urberville and Tess are on their way to Trantridge. She is afraid that she will be thrown out of the carriage if she touches his arm. She tries to hold on to him, but he insists that she never touch his arm again. They reach the summit of another declivity, and he asks her to put one kiss on those holmberry lips, or even on that warmed cheek, and she will stop. He says that he will break both their necks, but she refuses to move. He tells her that she is mighty sensitive for a cottage girl, and that she should be made sorry for the kiss he has given her. She says that she would not have come if he had known. He swore at her, and then he hems her in between the gig and the hedge. He laughs heartily at the sight of his misdemeanour, and says he would never do it any more against his will now.

GPT-4-No-KG ($F_{\text{KG}} = 26.82$):
Tess, in white muslin, dreads Alec's driving and his horse Tib's temperament. Her defiance peaks when she wipes away Alec's forced kiss with her handkerchief. Despite her involuntary hold on him during their descent near Melbury Down and Wingreen, she asserts her resolve, opting to walk to Trantridge, her strategic silence marking her commitment to her family's welfare.

GPT-4-KG ($F_{\text{KG}} = 31.32$):
The emotional stakes are high as Tess's big tear and flushed face betray her distress during the tumultuous ride with d'Urberville, whose injured tone and passionate outbursts oscillate between anger and feigned concern. Tess's strategic silence and defiance, coupled with her evasion of his kiss, underscore her determination to resist his control. D'Urberville's laugh at her resistance and his role as a passionate companion add complexity to their dynamic. As they near Trantridge, Tess's resolve is clear, despite the emotional toll of the journey.