Extracting Summary Knowledge Graphs from Long Documents

Zeqiu Wu Rik Koncel-Kedziorski Mari Ostendorf Hannaneh Hajishirzi
University of Washington, Seattle, WA, USA
{zeqiuwu1, kedzior, ostendor, hannaneh}@uw.edu

Abstract

Knowledge graphs capture entities and relations from long documents and can facilitate reasoning in many downstream applications. Extracting compact knowledge graphs containing only salient entities and relations is important but challenging for understanding and summarizing long documents. We introduce a new text-to-graph task of predicting summarized knowledge graphs from long documents. We develop a dataset of 200k document/graph pairs using automatic and human annotations. We also develop strong baselines for this task based on graph learning and text summarization, and provide quantitative and qualitative studies of their effect.

1 Introduction

Knowledge graphs are popular representations of important entities and their relationships. Compact, interpretable knowledge can graphs facilitate human data analysis as well as empower memory-dependent knowledge-based applications. This makes them ideal for modeling the content of documents. Document-level information extraction which captures relations across distant sentences can be used to construct knowledge graphs of documents (Jia, Wong, and Poon 2019; Yao et al. 2019). These techniques focus on extracting all entities and relations from a document, which for long and dense documents such as scientific papers may be hundreds or thousands. This poses a new challenge: how do we determine the most important entities in a paper and the key relationships between them?

Automatic summarization (Liu and Lapata 2019; Yasunaga et al. 2019) addresses the problem of identifying salient information in a document, but introduces the additional challenge of discourse structuring (and in the abstractive case, text generation as well). Summarizing entities and relations directly as a first step could decouple the mixed burdens on models and help assure the factual correctness of a summary in line with recent trends in evaluation for summarization (Wang, Cho, and Lewis 2020; Durmus, He, and Diab 2020; Zhang et al. 2020).

In this work, we introduce the task of extracting from a scientific document a compact knowledge graph that represents its most important information. Figure 1 illustrates the situation: a large knowledge graph can be extracted from the document, but only a portion of entities and relations characterize its main ideas (colored nodes and thick edges), while the rest play a more minor role. Our task emphasizes finding this salient subgraph. We support this task with a dataset of 200k scientific document/graph pairs that integrates automatic and human annotations from existing knowledge resources in the scientific domain. We outline an evaluation paradigm that balances accuracy against redundancy while admitting the variability of textual reference to an entity.

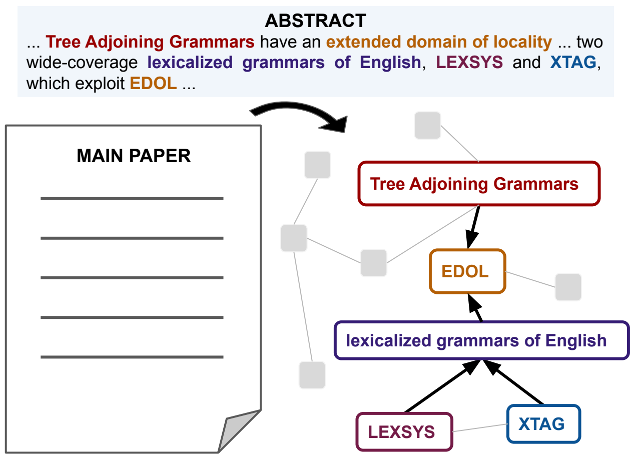
ABSTRACT
... Tree Adjoining Grammars have an extended domain of locality ... two wide-coverage lexicalized grammars of English, LEXSYS and XTAG, which exploit EDOL ...
MAIN PAPER
Tree Adjoining Grammars
EDOL
lexicalized grammars of English
LEXSYS
XTAG

Figure 1: We introduce the task to extract a summary knowledge graph from a long document (e.g. a scientific paper). This is an example from our dataset, where the target summary graph should only contain entities and relations that are salient enough to be included in the abstract of the paper. The entities or relations (shown in light grey) not found in the abstract should be removed. We omit entity and relation types for simplicity.

We develop and investigate two competitive baselines based on text summarization and graph learning models, and compare to two simple frequency-based methods. We provide an analysis of their tradeoffs, and of the general challenges posed in the proposed dataset. For example, we observe that missing entities and entity coreference errors in the predicted graphs have a large impact on relation accuracy. Our hope is that this task and data will facilitate research into future models that can better capture these challenging but important textual relations.

2 Background / Related Work

2.1 Information Extraction (IE)

Most IE work focuses on extracting entities and their relational facts from a single sentence (Zhang et al. 2017; Stanovsky et al. 2018). More recent work addresses document-level IE which aims to capture relations across distant sentences (Jain et al. 2020; Jia, Wong, and Poon 2019; Yao et al. 2019) or to fill a pre-defined metadata table with entities (Hou et al. 2019). Jia, Wong, and Poon (2019), Yao et al. (2019) and Hou et al. (2019) formulate the task as classifying the relation type between each pair of ground-truth entities expressed in the document. We do not assume the existence of ground-truth entities and extract document-level relations directly from text.

Our work complements this trend of applying IE for long document understanding while addressing the need for focused, compact knowledge representations. The closest work to our is Jain et al. (2020), who separately explore the idea of identifying salient entities related to experimental results using weak supervision provided by the Papers with Code dataset.[1] In contrast, we focus on identifying salient entities from the paper based on weak supervision from its abstract. This framing generalizes to a wider variety of documents and domains, and supports diverse tasks including multi-document summarization or building scientific knowledge bases. Besides entity salience, our task also requires models to identify the salience of relations, which we show to be challenging.

2.2 Text Summarization

Document summarization models create summaries by identifying the most important sentences from documents (Nallapati, Zhai, and Zhou 2017; Narayan, Cohen, and Lapata 2018) or using a decoder to generate abstractive summaries (Rush, Chopra, and Weston 2015; Celikyilmaz et al. 2018). Although text summarization tasks (Liu and Lapata 2019; Yasunaga et al. 2019) share our objective of distilling crucial information from documents, they mix this objective with the goal of producing fluent natural language text.

We argue that summarizing entities and relations directly as the first step could decouple the mixed burdens on models and help models to check the factual correctness of a summary. These advantages can benefit other text generation tasks that rely on long document understanding and representation, such as generation grounded on long text. An increasing number of recent works (Wang, Cho, and Lewis 2020; Durmus, He, and Diab 2020; Zhang et al. 2020) have proposed automatically evaluating summarization models by applying information extraction or question answering models to match entities or relations between generated and reference summaries. These newly proposed measures are found to have much higher correlation with human judgements than standard measures.

Recent applications of large pretrained language models such as Ribeiro et al. (2020) and Kale (2020) show the promise of generating fluent and accurate text from knowledge graphs, highlighting the need for identifying correct underlying knowledge representations. In addition, summarized knowledge graphs from multiple documents can be naturally merged by collapsing shared entity nodes to bring even richer information. And such summarized structures can be more easily leveraged to facilitate downstream tasks.

Another line of work that is closely related to ours is graph-based summarization, which leverages graph structures of documents to facilitate the summary generation (Erkan and Radev 2004; Tan, Wan, and Xiao 2017; Yasunaga et al. 2019; Huang, Wu, and Wang 2020; Xu et al. 2020). These works try to leverage graphs that capture relations between sentences or discourse units. Wang et al. (2020a) incorporate graphs between entities extracted by sentence-level IE systems without considering entity or relation salience.

2.3 Graph Summarization

In general, graph summarization work can be categorized according to whether its goal is optimizing for memory or computational resources needed for processing, or improving analysis (Liu et al. 2018). Knowledge graph summarization (Safavi et al. 2019) or node estimation (Park et al. 2019) are most relevant to our work, but they normally take a huge knowledge graph for pruning based on a given query, without any document context. Unlike these works, our task requires the model to handle different document contexts at inference, where each document can contain a completely different knowledge graph with unique entities. Our work is also similar to Falke and Gurevych (2017), who collect a small corpus (30 data instances) of concept map annotations from OpenIE tuples for summarizing sets of documents. We choose a science-specific annotation scheme which provides more structure as our target.

2.4 Scientific Document Understanding

Although our proposed idea of summarized knowledge graphs can be applied to documents in any domain, we focus on scientific papers in this work. Existing works toward understanding scientific documents include but are not limited to information extraction (Luan et al. 2018; Wadden et al. 2019), summarization (Cohan et al. 2018; Collins, Augenstein, and Riedel 2017; Yasunaga et al. 2019), fact verification (Wadden et al. 2020), and citation generation (Luu et al. 2020; Xing, Fan, and Wan 2020). Recently, attention to research in the scientific domain in the NLP community has grown even more (Wang et al. 2020b; Esteva et al. 2020) due to the urgent need for mitigating the COVID-19 global pandemic.

3 Task Formulation

We introduce a text-to-graph task of extracting a succinct, structured knowledge graph which contains the most salient entities and relations from a document. More specifically, such a summarized knowledge graph should meet these conditions: 1) it contains only the most important entities from the document; 2) it includes relations between these entities only if they are crucial to understanding the main ideas of the text; and 3) each salient entity is only represented by a

[1]Papers with Code: `paperswithcode.com`

single node in the graph. These conditions are evaluated as entity salience, relation salience, and entity duplication rate, respectively (evaluation details in Section 5). In our dataset (see Section 4) where long documents are scientific papers, the most salient entities and relations are defined to be those that can be included in paper abstracts.

Figure 1 shows an example of this task. Here, only entities and relations that appear in the abstract should be included in the summary knowledge graph. Those in grey should be removed even though they are mentioned in the paper, because they are not necessary enough to describe or understand the main idea of the paper.

Formally, we define the problem as: Given a document D, a pre-defined entity type set T_v and a relation type set T_R, predict a summarized knowledge graph $\mathcal{G} = (V, E)$, where each $v_i \in V$ represents a salient entity with entity type $t_i \in T_v$ mentioned in D. Each $(v_i, v_j, r_{ij}^k) \in E$ represents an important edge from v_i to v_j with relation type $r_{ij}^k \in T_R$. We note there can be multiple edges between v_i and v_j, but $r_{ij}^k = r_{ij}^l \iff k = l$. Each v_i consists of a cluster of n_i string names $\{m_i^1, m_i^2, ..., m_i^{n_i}\}$ (from co-referent entity mentions).

4 Our Dataset: SciGraphSumm

We construct a text-to-graph dataset from a corpus of scientific papers. Our textual data consists of roughly 200k computer science research papers taken from S2orc, a corpus of 8 million full-text research papers and abstracts (Lo et al. 2020). We leverage abstracts to create summarized knowledge graphs for full papers. Abstracts effectively contain summarized information from full documents, and there are existing human annotation and information extraction (IE) systems that enable constructing relational graphs from abstracts.

Due to the expense and difficulty of annotation, we only have access to a small number of human-annotated summary graphs, which we use to judge system performance. We call this data the "human test set". For training and development, we take a weakly supervised approach using automatically extracted summary graphs. We use a scientific IE system to extract summary graphs from 196k abstracts and pair them with full papers from S2orc. Entity-relation graphs are also extracted for the full papers. We divide these graph/paper tuples into train, dev and automatic test sets. The dev set can be used for parameter tuning purposes, while the auto test set can be used to compare systems. Randomly sampling 115 examples, we observe that over 90% of extracted target entities for abstracts in the automatic test set are meaningful. Moreover, we will show in Section 7 that similar system performance trends are observed in the "human test set" and "automatic test set". Table 1 gives statistics about the number and size of the textual documents, as well as the summary graphs, for all data splits. The data collection and graph construction details are described in the sections that follow. The automatic test set is much larger than the human test set, which reduces the problems of noise in automatic annotation.

	Train	Dev	Auto Test	Human Test
# examples	190k	1k	5k	234
# doc tokens	6.4k	6.5k	6.4k	3.9k
# graph entities	13.4	13.6	13.4	11.2
# graph relations	10.9	11.1	11.0	9.1

Table 1: Statistics of each data split.

4.1 Manual Summarized Graph Annotation

We leverage the SciERC scientific IE dataset (Luan et al. 2018) for the human-labeled test data. SciERC consists of 500 expert-annotated paper abstracts, labeled with entities (6 types: Task, Method, Metric, Material, Other Scientific Term and Generic), co-reference, and relations (7 types: Compare, Part-of, Conjunction, Evaluate-for, Feature-of, Used-for, Hyponym-of). We construct knowledge graphs from these annotations by collapsing coreferent mentions into a single node and linking all nodes via the annotated relations. Of the 500 abstracts in SciERC, 300 have full texts available in S2orc. In order to guarantee information richness, We discard pairs where annotated graphs have fewer than 5 predicted relations. After such filtering, our "human test set" consists of the knowledge graphs and full text of 234 of these documents.

4.2 Automatic Summarized Graph Annotation

To facilitate model training and model development under a weakly supervised setting, we automatically create target knowledge graphs for the remaining papers from their abstracts. We leverage DyGIE++, a state-of-the-art scientific IE system that extracts entities, relations and co-references simultaneously (Wadden et al. 2019). We do not re-train DyGIE++, instead we use the pretrained model on SciERC for all processing and modeling steps in this work. We construct knowledge graphs from the IE output by again collapsing coreferences to create entities, which we associate with the list of coreferential text mentions. IE relations between mentions become edges between the corresponding entities in the graph. The same sample filtering is applied.[2]

4.3 Automatic Full Graph Construction

Our task is designed to build a summarized graph directly from a document, and in order to perform the task, models do not have to use specific information extraction tools to build a full graph first. However, we provide the full knowledge graphs that we constructed from documents as part of our dataset for reproducibility and to encourage future exploration on graph learning models for our task. We process each full document text with DyGIE++ in overlapping 300-token windows (reduce computation memory) with each two consecutive chunks having one overlapped sentence to preserve cross-sentence co-references.[3] We collapse coreferen-

[2] We additionally discard pairs with abstracts that are longer than 500 tokens (rare) to avoid memory limitations of DyGIE++.

[3]We discard sentences longer than 150 tokens to guarantee one overlapped sentence between each consecutive chunks (fewer than 1% sentences discarded).

tial mentions as previous steps, and then collapse coreference clusters from different windows with matching unique, non-generic mentions into a single graph node. A non-generic entity mention is a string (excluding pronouns and determiners) with more than one token, or a unigram with an inverse document frequency (IDF) in the training data that is higher than an empirically chosen threshold, selected for high precision in identifying generic mentions. A generic entity mention is not clustered unless the model predicts it to be coreferent with some other entity mention.

5 Evaluation Metrics

The goal is to evaluate the correctness of the predicted summary knowledge graph compared with the ground truth summary graph. We first align entity nodes in the predicted graph to nodes in the target graph. After the entity alignment step, we measure 3 qualities of the predicted graphs – entity salience, relation salience, and duplication rate – under a relaxed alignment condition, described next.

5.1 Entity Alignment

In the "human test set", we found 30% annotated entity mentions do not have exact string match in the main paper text. Further analysis showed that many such cases are due to minor paraphrasing, hyphenation differences or typos caused by OCR parsers that are 1commonly used to process papers in PDF format. For example, *in-domain monolingual corpus* in the abstract and *in domain monolingual corpus* in the paper are equivalent but do not have an exact match due to the hyphen difference. In addition, as we do not assume any specific information extraction tools being used, a similar issue for exact name matching may occur, potentially due to different entity mention names being extracted by different models. Therefore, exact string match does not yield a good alignment, and we instead use a relaxed alignment method that we found to be reasonably accurate for evaluation.

Another issue in aligning entities between two graphs is that the same entity can be referred to with multiple strings, as each entity node represents a cluster of co-referent entity mentions. To align a predicted node with a target node where either can have a cluster of mention types, we find the maximum similarity over all possible pairs. The similarity score between a target entity $v_i = \{m_i^1, m_i^2, ..., m_i^{n_i}\}$ and a predicted entity $\hat{v}_j = \{\hat{m}_j^1, \hat{m}_j^2, ..., \hat{m}_j^{n_j}\}$ is calculated as:

$$s_{i,j} = \max_{s,t} \text{sim}(m_i^s, \hat{m}_j^t) \tag{1}$$

where we employ "gestalt pattern matching" (Ratcliff and Metzener 1988) to calculate string similarity based on common substrings. Each predicted node is aligned with the target node that gives the highest similarity score, subject to a minimum score λ. $\lambda = 0.7$ is selected such that, in a set of 200 relaxed (but not exact) alignments, 90% of them are manually inspected to be acceptable.

The 200 manually inspected samples fall into the following categories:

- **Paraphrases of Target Nodes (50%):** We consider relaxed alignment examples to be good if differences only involve typo, hyphen, item order or other paraphrases. For example, *log-linear and linear interpolation* versus *linear and log-linear interpolation*
- **Different Specificity Level (40%):** We consider aligned entities with different specificity level as relevant alignments. For example, *speaker's intention prediction modules* versus *intention prediction modules*.
- **Alignments with Error (10%):** We consider entities being aligned that have distinct meanings to be bad alignments. For example, *two-dimensional analog of sorting* versus *one-dimensional notion of sorting*.

In both human and auto test sets, applying the relaxed alignment from full graph nodes to target nodes increases the percentage of aligned target nodes from 60% (by exact matching) to 80%.

5.2 Salience and Duplication Measures

To calculate entity salience, we align each predicted node with up to one target graph node, collapsing multiple predicted nodes that map to the same target entity into a single node for calculating precision, recall and F1 scores. In other words, if multiple predicted entities are aligned to the same target entity, it is only counted once when calculating all scores. These metrics can be computed either with matching or ignoring entity types (typed vs. untyped evaluation). As each entity should only have one entity type, we adopt the dominating type among all mentions to be the entity type. Since this process does not penalize predicted graphs where multiple nodes are aligned to a single target node, we also calculate the duplication rate as the average number of predicted nodes which are aligned to each target node.

Based on entity alignment, a target relation edge (v_i, v_j) can be aligned to a predicted relation for ($\hat{v}_k$, $\hat{v}_l$) if the corresponding nodes align (i.e., v_i aligns to $\hat{v}_k$ and v_j aligns to $\hat{v}_l$). We evaluate relation salience based on such relation alignments, allowing for multiple relation types between each pair of entities. We report precision, recall and F1 scores for relation prediction, with or without considering relation type and direction matching (typed vs. untyped evaluation). When evaluating without relation type and direction, we merge relations (if multiple) between an entity pair into a single edge.

6 Baseline Models

We develop two baseline models for the graph summarization problem: one using a text summarization model that extracts summary sentences from which we extract entities and relations, and one that first builds a full-document graph and then applies a graph learning model to do graph pruning.

6.1 Text-Text-Graph (TTG)

This model first produces a text summary of the full document text using the extractive summarizer BertSumExt (Liu and Lapata 2019), and subsequently uses entities and relations from the text summary to form a summary knowledge graph. We re-train the original model on our dataset, by replacing the pre-trained BERT with SciBERT (Beltagy, Lo, and Cohan 2019) and increase the sequence length from 512 to 1024. DyGIE++-predicted entities and relations that appear within the text summary are used as the summarized graph.

6.2 Graph-to-Graph (G2G)

This model predicts a summary subgraph from the full graph extracted by DyGIE++ (described in Section 4).

We formulate subgraph selection as a node classification problem: we encode the full graph with a GAT (Veličković et al. 2018) and use the resulting node representations to make a binary salience prediction.

In the original GAT, a node v_i is embedded with a learnable feature vector and contextualized via multi-headed attention with its graph neighbors $\mathcal{N}(v_i)$ in each graph attention layer. At each graph attention layer, a vertex v_i with neighborhood $\mathcal{N}(v_i)$ is contextualized as:

$$\hat{\mathbf{v}}_i = \mathbf{v}_i + \sum_{j \in \mathcal{N}(v_i)} \alpha_{ij} \mathbf{W}_V \mathbf{v}_j \tag{2}$$

$$\alpha_{ij} = \frac{\exp((\mathbf{W}_K \mathbf{v}_j)^\top \mathbf{W}_Q \mathbf{v}_i)}{\sum\limits_{z \in \mathcal{N}(v_i)} \exp((\mathbf{W}_K \mathbf{v}_z)^\top \mathbf{W}_Q \mathbf{v}_i)} \tag{3}$$

Here $\hat{\mathbf{v}}_\mathbf{i}$, $\mathbf{v}_\mathbf{i} \in \mathbb{R}^h$ denote contextualized and original vector representations of v_i. $\mathbf{W}_V$, $\mathbf{W}_K$, $\mathbf{W}_Q \in \mathbb{R}^{h \times h}$ are model parameters, and α_{ij} are attention weights computed from the vertex representations. The formulation above was extended using multi-head attention and layered with non-linearities to produce the Graph Attention Network.

Since the original GAT does not consider different relation types between neighboring vertices, to incorporate relation types into the model, we use separate heads for different relation types in T_R; that is, the head corresponding to relation type $R \in T_R$ is used to attend v_i over those $v_j \in \mathcal{N}_R(v_i)$ where v_i and v_j are connected by an edge with label R, i.e. $(v_i, v_j, R) \in E$. As we have 7 different relation types in our dataset, we use 7 heads in our GAT. The representations from all heads are concatenated and transformed via non-linearity between model layers.

At the *node embedding layer*, we use four features to embed each entity node v_i: the number of mentions in the document n_i, the section id of the entity's first appearance in the document s_i, the most frequent entity type among all mentions as predicted by DyGIE++ t_i, and the pooled output representation from SciBERT (Beltagy, Lo, and Cohan 2019) of the longest mention string z_i, to encode each node as follows:

$$\mathbf{v}_i = n_i \mathbf{n} + \mathbf{W_s s_i} + \mathbf{W_t t_i} + \mathbf{W_e} S(z_i) \tag{4}$$

where $\mathbf{n} \in \mathbb{R}^h$ is a learnable unit feature vector for n_i, $\mathbf{s_i} \in \mathbb{R}^{n_s}$ and $\mathbf{t_i} \in \mathbb{R}^{n_t}$ are the one-hot vectors of s_i and t_i respectively, with n_s and n_t as the number of unique section ids and node (entity) types in the dataset. $S(z_i) \in \mathbb{R}^{h_e}$ is the hidden representation at the first token ([CLS]) from the final layer of SciBERT. $\mathbf{W}_s \in \mathbb{R}^{h \times n_s}$, $\mathbf{W_t} \in \mathbb{R}^{h \times n_t}$, $\mathbf{W}_e \in \mathbb{R}^{h \times h_e}$ are trained model parameters.

Following the node embedding layer, we contextualize each node representation with 6 *GAT layers* and pass each node through a final binary classification layer to predict salience.

To supervise the training of this model, we apply the relaxed alignment method (in Section 5) to align full graph entities and target graph entities. All full graph entities that can be aligned are treated as positive examples, and all others as negative. Finally, we use a negative log likelihood loss function using all positive (labeled as salient) nodes and a negative sampling ratio of 3 for training.

We include all full graph relations between predicted entities in the output summary graph. We leave better relation prediction for future study.

Implementaion Details We manually tune the hyperparameters of GAT based on dev set entity F1 performance curve versus training steps. We fix most of the model parameters (e.g. vector dimension h = 16; number of layers = 6; batch size = 10). The only parameters being tuned are learning rate, dropout rate and negative sampling ratio. But we only manually change each parameter value if we observe performance instability on the dev set for the first 1000 training steps. The average number of tuning trials for each parameter is fewer than 5 times. Finally we set negative sampling ratio = 3, dropout rate = 0.2 and learning rate = 5e-5 for all experiments with our G2G model. We use Adam optimizer. We do not finetune SciBERT (the base model) used in GAT. We run each experiment on a single TITAN RTX. We select the model checkpoint based on its typed relation F1 score performance on valid set.

7 Experiments

7.1 Evaluated Systems

We compare the **TTG** and **G2G** models in Section 6 with two frequency-based baselines: **PageRank (PR)**: the top K most "authorized" entities in the full document graph with the highest PageRank scores (Page et al. 1999), where each edge weight is initialized by the number of relation mentions between the entity pair; **TopK-Freq (TKF)**: the top K most frequent entities. K is selected to be 18 for both of the models, which is the average number of full graph nodes aligned to target nodes in the training set. In both cases, relations from the full document graphs between the selected entities are added to generate the predicted summary graph. We also report performance of **Gold Entity (GE)**, which provides the performance upper bound when relying on the entities and relations that can be extracted from the full text with DyGIE++. GE picks the full graph node with the highest similarity score for each target node (with a lower threshold of 0.7 for inclusion). Again, the predicted graph includes all relations found in the full graph between these entities.

7.2 Quantitative Results

Table 2 shows Precision, Recall and F1 scores for both untyped and typed entity and relation prediction, as well as entity duplication rates for the automatic test set. We see similar evaluation trends with untyped and typed evaluation, with lower scores when type matching is required, as expected. Both G2G and TTG outperform other baselines in

	Untyped						**Typed**						
	Ent P	Ent R	Ent F1	Rel P	Rel R	Rel F1	Ent P	Ent R	Ent F1	Rel P	Rel R	Rel F1	E Dup
PR	23.6	34.3	26.8	6.6	8.1	6.7	17.2	24.8	19.5	4.8	6.5	5.0	1.30
TKF	24.6	35.4	27.9	7.8	8.1	7.3	17.8	25.5	20.1	5.5	6.4	5.3	1.34
TTG	**30.2**	29.7	28.3	**11.2**	6.6	7.5	**22.7**	22.3	21.2	**9.5**	5.6	6.4	**1.18**
G2G	29.7	**43.8**	**32.7**	9.1	**12.6**	**9.2**	21.6	**31.8**	**23.7**	6.5	**10.2**	**6.9**	1.55
GE	100.0	81.2	88.8	44.5	18.9	24.3	77.6	63.4	69.2	34.8	16.2	20.0	1.0

Table 2: Full results for **untyped** and **typed** entity / relation evaluation (P, R for Precision and Recall) and entity duplication rate (E Dup) on **auto test** set. All scores are in %, except entity duplication rate. Note that entity duplication rates are expected to be the same for both untyped and typed evaluation as entity alignment only considers string name matching.

	Untyped						**Typed**						
	Ent P	Ent R	Ent F1	Rel P	Rel R	Rel F1	Ent P	Ent R	Ent F1	Rel P	Rel R	Rel F1	E Dup
PR	22.8	38.7	27.8	6.6	9.3	7.2	17.2	29.1	21.0	4.7	7.3	5.4	1.30
TKF	24.3	40.7	29.4	8.9	9.4	8.4	18.0	30.0	21.8	6.1	7.1	5.9	1.41
TTG	33.9	35.3	32.7	**13.7**	9.3	9.9	26.1	27.0	25.1	**11.5**	7.9	**8.3**	**1.17**
G2G	**36.9**	**42.6**	**36.9**	11.3	**11.8**	**10.1**	**28.0**	**32.1**	**27.8**	8.4	**9.3**	7.6	1.42
GE	100.0	79.4	87.6	44.5	20.3	25.8	79.9	64.0	70.3	34.8	17.3	21.3	1.0

Table 3: Full results for **untyped** and **typed** entity / relation evaluation (P, R for Precision and Recall) and entity duplication rate (E Dup) on **human test** set. All scores are in %, except entity duplication rate.

	Human test		**Auto test**	
	TTG	G2G	TTG	G2G
Task	21.9	24.4	14.3	19.1
Method	24.4	29.4	23.8	25.6
Metric	6.5	9.7	6.3	8.9
Material	13.9	14.0	8.0	10.3
Other Scientific Term	17.4	20.6	17.0	20.5
Generic	14.5	15.6	12.7	14.9

Table 4: Entity relaxed match F1 by entity type.

both entity and relation evaluations. In particular, G2G consistently performs the best in F1 scores, though TTG has higher precision. When evaluating entity duplication rate, TTG consistently performs the best. However, all systems' performances are still far from GE, which shows significant room for improvement on this task in the future.

Table 3 gives the same results for the human test set. Most of the trends observed with the automatic test set hold for the human test set. The exception is that G2G shows better performance on entities, but is less effective on typed relations. Therefore, we argue that the automatic test set is reasonable to be used as an extra set to test systems during development.

7.3 Qualitative Analysis

We looked at a handful of abstracts in the human test set to analyze the scoring criteria proposed for this task. The target graphs were further hand-annotated to identify the most important entities (among all salient entities in the abstract), eliminating some "generic" nodes (e.g. "method"), non-essential "other scientific term" nodes, and occasional duplicated nodes. In some cases, we added entities that were not in the gold reference but were identified by one or more of the automatic algorithms and deemed appropriate. Entities identified by the automatic algorithms were hand-aligned to the reduced target set. For untyped entities, the recall is higher on the reduced set for G2G and Pagerank, suggesting that these algorithms may be better capturing the most salient entities. Errors in entity types often involved unclear cases. Trends in precision on the reduced graph were consistent with the automatic scoring. What we also noticed is that some of the aligned predicted nodes contain unrelated entities due to coreference errors from IE systems, which in part explains the low relation scores together with the impact of missing/inserted entities.

We investigate the causes of TKF performing poorly compared with G2G and TTG. In specific, we analyze why frequency may not be a good indicator of salient entities in some cases. We first calculate the average length of mention string names of predicted salient entities, and find out TKF has an average length of 2.1 while both TTG and G2G have 2.5 on average. Furthermore, we find out that some important entities tend to have long string names, especially when paper authors start introducing some specific tasks or models. These entities tend to be split into smaller segments in later parts of the paper for more detailed explanations. Such smaller segments tend to be mentioned more frequently and thus predicted by TKF, although sometimes they are not comprehensive enough to qualify as salient entities. For example, a key entity *Bayesian semi-supervised Chinese word segmentation model* can be detected as salient by both G2G and TTG while TKF only predicts *Chinese words* and *word segmentation* as the closest salient entities. Another example is that TKF predicts *KL-One systems* as a salient entity for a paper, while the gold entity is *KL-One-like knowledge representation systems* which is predicted by both G2G and TTG.

As noted in Table 1, the sizes of target graphs and full papers are different in the human and automatic test sets. We observe that TTG produces graphs of similar size for papers in the two test sets: about 12 nodes and 7 edges. By contrast, G2G produces graphs of different sizes for different document lengths, averaging 14 nodes and 10 edges for human test set but 22 nodes and 17 edges for automatic test set, where documents are longer (and target graphs bigger).

We observe interesting trends regarding the sections where entities first appear. GE entities appear in the first section of the full paper only 55% of the time, for both auto and human test sets. About 25% and 20% of them have their first mention in middle 5 sections and final sections respectively. These numbers are also consistent with both test sets. This observation highlights the fact that extracting the main idea of the paper needs the understanding of the full paper. However, partly due to the sequence length limitation of BertSumExt, TTG is extremely biased towards entities in the first section (85%). G2G is less vulnerable to such bias (68%), but still often fails to include entities from later paper sections in its summaries.

Table 4 shows the entity F1 score based on relaxed match for each different entity type. We calculate these scores by comparing subgraphs of the predicted and target graphs that contain entities of a certain type only. Entities of type "Metric" are the hardest to predict. This correlates with the fact that "Metric" entities are least likely to appear in the first section of a paper (36.7% vs 54.9% overall in human test set). Another possible reason for this is that "Metric" is the least frequent salient entity type. Only 5% and 6% of all target salient entities have the type "Metric", in auto and human test set respectively.

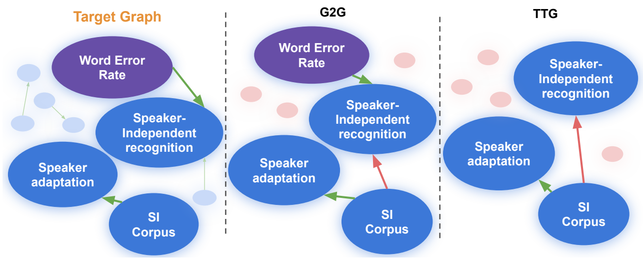

Figure 2: Sample output from G2G and TTG.

Figure 2 shows an example where a target "Metric" entity first appearing in the second section of a paper is correctly predicted by G2G, but missed by TTG. The red (incorrect) relation edges show that only relying on full graph relations limits relation prediction performance. This is evidenced in Tables 3 and 2, where even GE gives low relation prediction scores due to the absence of gold target relations in the full document graphs.

8 Conclusions and Future Work

We have described a new text-to-graph task for constructing summary knowledge graphs from full text documents, including a standard, preprocessed open-access dataset and evaluation techniques to facilitate further research. We have investigated graph classification and text summarization techniques for this task, and detailed some of their qualities in our analysis.

As we show that relation salience prediction is a rather challenging task in extracting summary knowledge graphs, it would be an important further investigation. Leveraging document-level IE and graph learning techniques would also be an interesting direction to explore. Models that can merge entity nodes better will lead to lower entity duplication rate and improved relation accuracy. One major shortcoming of our GAT model is that we do not consider the context of each entity mention in the document; incorporating contextual information for entity mentions would also be a promising research direction.

References

Beltagy, I.; Lo, K.; and Cohan, A. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *EMNLP*.

Celikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep Communicating Agents for Abstractive Summarization. In *NAACL*.

Cohan, A.; Dernoncourt, F.; Kim, D. S.; Bui, T.; Kim, S.; Chang, W.; and Goharian, N. 2018. A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents. In *NAACL*.

Collins, E.; Augenstein, I.; and Riedel, S. 2017. A Supervised Approach to Extractive Summarisation of Scientific Papers. In *CoNLL*.

Durmus, E.; He, H.; and Diab, M. 2020. FEQA: A Question Answering Evaluation Framework for Faithfulness Assessment in Abstractive Summarization. In *ACL*.

Erkan, G.; and Radev, D. R. 2004. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. In *Journal of artificial intelligence research*.

Esteva, A.; Kale, A.; Paulus, R.; Hashimotoa, K.; Yin, W.; Radev, D.; and Socher, R. 2020. CO-Search: COVID-19 Information Retrieval with Semantic Search, Question Answering, and Abstractive Summarization. *arXiv* abs/2006.09595.

Falke, T.; and Gurevych, I. 2017. Bringing Structure into Summaries: Crowdsourcing a Benchmark Corpus of Concept Maps. In *EMNLP*.

Hou, Y.; Jochim, C.; Gleize, M.; Bonin, F.; and Ganguly, D. 2019. Identification of Tasks, Datasets, Evaluation Metrics, and Numeric Scores for Scientific Leaderboards Construction. In *ACL*.

Huang, L.; Wu, L.; and Wang, L. 2020. Knowledge Graph-Augmented Abstractive Summarization with Semantic-Driven Cloze Reward. In *ACL*.

Jain, S.; Zuylen, M. V.; Hajishirzi, H.; and Beltagy, I. 2020. SciREX: A Challenge Dataset for Document-Level Information Extraction. In *ACL*.

Jia, R.; Wong, C.; and Poon, H. 2019. Document-Level N-ary Relation Extraction with Multiscale Representation Learning. In *NAACL-HLT*.

Kale, M. 2020. Text-to-Text Pre-Training for Data-to-Text Tasks. *arXiv* abs/2005.10433.

Liu, Y.; and Lapata, M. 2019. Text summarization with pretrained encoders. In *EMNLP*.

Liu, Y.; Safavi, T.; Dighe, A.; and Koutra, D. 2018. Graph Summarization Methods and Applications: A Survey. In *ACM Computing Surveys*.

Lo, K.; Wang, L. L.; Neumann, M.; Kinney, R.; and Weld, D. S. 2020. S2ORC: The Semantic Scholar Open Research Corpus. In *ACL*.

Luan, Y.; He, L.; Ostendorf, M.; and Hajishirzi, H. 2018. Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In *EMNLP*.

Luu, K.; Koncel-Kedziorski, R.; Lo, K.; Cachola, I.; and Smith, N. A. 2020. Citation Text Generation. *arXiv* abs/2002.00317.

Nallapati, R.; Zhai, F.; and Zhou, B. 2017. SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents. In *AAAI*.

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Ranking Sentences for Extractive Summarization with Reinforcement Learning. In *NAACL-HLT*.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: Bringing order to the web. In *Technical Report 1999-66, Stanford InfoLab*.

Park, N.; Kan, A.; Dong, X. L.; Zhao, T.; and Faloutsos, C. 2019. Estimating Node Importance in Knowledge Graphs Using Graph Neural Networks. In *KDD*.

Ratcliff, J. W.; and Metzener, D. E. 1988. Pattern Matching: the Gestalt Approach. *Dr. Dobb's Journal* .

Ribeiro, L. F. R.; Schmitt, M.; Schutze, H.; and Gurevych, I. 2020. Investigating Pretrained Language Models for Graph-to-Text Generation. *arXiv* abs/2007.08426.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *EMNLP*.

Safavi, T.; Belth, C.; Faber, L.; Mottin, D.; Muller, E.; and Koutra, D. 2019. Personalized Knowledge Graph Summarization: From the Cloud to Your Pocket. In *ICDM*.

Stanovsky, G.; Michael, J.; Zettlemoyer, L.; and Dagan, I. 2018. Supervised Open Information Extraction. In *NAACL*.

Tan, J.; Wan, X.; and Xiao, J. 2017. Abstractive Document Summarization with a Graph-Based Attentional Neural Model. In *ACL*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR*.

Wadden, D.; Lin, S.; Lo, K.; Wang, L. L.; Zuylen, M. V.; Cohan, A.; and Hajishirzi, H. 2020. Fact or Fiction: Verifying Scientific Claims. *arXiv* abs/2004.14974.

Wadden, D.; Wennberg, U.; Luan, Y.; and Hajishirzi, H. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. In *EMNLP*.

Wang, A.; Cho, K.; and Lewis, M. 2020. Asking and Answering Questions to Evaluate the Factual Consistency of Summaries. In *ACL*.

Wang, D.; Liu, P.; Zheng, Y.; Qiu, X.; and Huang, X. 2020a. Heterogeneous Graph Neural Networks for Extractive Document Summarization. In *ACL*.

Wang, L. L.; Lo, K.; Chandrasekhar, Y.; Reas, R.; Yang, J.; Burdick, D.; Eide, D.; Funk, K.; Katsis, Y.; Kinney, R.; Li, Y.; Liu, Z.; Merrill, W.; Mooney, P.; Murdick, D.; Rishi, D.; Sheehan, J.; Shen, Z.; Stilson, B.; Wade, A.; Wang, K.; Wang, N. X. R.; Wilhelm, C.; Xie, B.; Raymond, D.; Weld, D. S.; Etzioni, O.; and Kohlmeier, S. 2020b. CORD-19: The COVID-19 Open Research Dataset. In *ACL NLP-COVID Workshop*.

Xing, X.; Fan, X.; and Wan, X. 2020. Automatic Generation of Citation Texts in Scholarly Papers: A Pilot Study. In *ACL*.

Xu, J.; Gan, Z.; Cheng, Y.; and Liu, J. 2020. Discourse-Aware Neural Extractive Text Summarization. In *ACL*.

Yao, Y.; Ye, D.; Li, P.; Han, X.; Lin, Y.; Liu, Z.; Liu, Z.; Huang, L.; Zhou, J.; and Sun, M. 2019. DocRED: A Large-Scale Document-Level Relation Extraction Dataset. In *ACL*.

Yasunaga, M.; Kasai, J.; Zhang, R.; Fabbri, A. R.; Li, I.; Friedman, D.; and Radev, D. R. 2019. ScisummNet: A Large Annotated Corpus and Content-Impact Models for Scientific Paper Summarization with Citation Networks. In *AAAI*.

Zhang, Y.; Merck, D.; Tsai, E. B.; Manning, C. D.; and Langlotz, C. P. 2020. Optimizing the Factual Correctness of a Summary: A Study of Summarizing Radiology Reports. In *ACL*.

Zhang, Y.; Zhong, V.; Chen, D.; Angeli, G.; and Manning, C. D. 2017. Position-aware Attention and Supervised Data Improve Slot Filling. In *EMNLP*.