School of Informatics

Informatics Research Review
Combining Semantic Graph with Deep Learning Toward Better Abstractive Text Summarization

[redacted]

January 2021

Abstract

Abstractive summarization refers to create a shorter version of the given text with the same meaning, while rephrasing the sentences. Before the emergence of deep learning models and pushing many datasets to the new state-of-the-art performance, traditional summarization methods were mainly based on the construction and application of semantic representation. However, these two methods are not independent of each other but could learn from others' strengths to offset one's weakness. This work presents a comprehensive review of the previous works from 2015 to 2020 in abstractive summarization via exploring the coalescence of semantic graphs with deep learning models. Along with this, we identified three benefits for combining semantic graphs with deep learning, and categorize the selected papers accordingly: 1) generate more meaningful and rich-content summaries, 2) improve the readability and grammaticality for produced summaries, and 3) enhance the interpretability of the end-to-end deep learning-based summarizers. Finally, we also list down the opening challenges and discuss the potential future works in this direction.

Date: Friday 22nd January, 2021

Supervisor: [redacted]

Contents

1 Introduction

Summarization is one of the most challenging tasks in the natural language processing (NLP) field, which requires understanding and then generating the natural language. Summarization is to let the machine rephrase the given text with a shorter version while remains the original meaning and salient information. It greatly reduces the time and labor cost for a human to write the summaries; thus being good for the efficient information browsing and querying in the era of information explosion [1].

There were lots of works that have been investigated in summarization. In the classical NLP research, the summarizing system was mainly built in a bottom-up fashion. Firstly, a semantic parser produced the semantic representation for the document; then such linguistic illustration would be fed into a summary generator [2]. In NLP, a semantic representation normally refers to the scheme representing the meaning of the text [3]. It could be formal language simply, e.g., the first-order logic, or structurally sophisticated such as a graph. Nowadays, the semantic representations are normally designed with structural property (tree, graph, etc.), e.g., Abstractive meaning representation (AMR) scheme utilizes directed acyclic graphs to represent the sentential meaning, containing nodes (entities, attributes, etc.) and edges (parametric relationships). The solely semantic-based summarization was widely explored in early NLP works, which mainly produces the desired summaries by extracting ontological and syntactic relations in text. Recently, the deep learning-based methods such as *sequence-to-sequence framework (Seq2Seq)* [4] has had groundbreaking performances and achieves state-of-the-art performance on summarization task [5, 6, 7], due to the tremendous amount of labeled data, and large corpus for training powerful language models [8].

In fact, semantic-based or deep learning methods have their shortcomings; however, they can complement each other. The pure deep neural networks, like Seq2Seq, are widely criticized for their black-box essence for information processing [9]: in *understanding* phase, the encoder networks learn the feature representation as a flat vector for input sequence, and the decoder produces the summary from the hidden features in *generation* phase. The flat vector is hard to interpret and understand by a human; its capability for storing information is limited. Thus, vector-based representation essentially brings the problem in explainability and efficiency in

long-text processing. Comparatively, structured representation plays a crucial role in NLP as its ability to capture particularly rich structural information [10]. The research in cognitive science [11, 12, 13, 14] also revealed that the human cognitive procedure is highly reliant on the graphical structure representation of information with compositions of entities and relations [11, 15], which naturally motivates the researchers to consider involving the structured representation to make the machine better understand language. Although the pure semantic-based methods do not achieve comparable results with the deep learning approach [1], it inspired a feasible way that incorporating semantic graphs into deep neural models [16]. Such combination has three major benefits: 1) The explicit structured representation for text encoding improves the interpretability of the deep learning (DL) model; 2) The semantic graphs usually entails both semantic and syntactic information, which could guide the DL models to generate more readable and grammatical summaries; 3) The semantic information and structural information, i.e., explicit modeling of word dependencies, increase the content richness of the generated text of DL models. On the other hand, traditional semantic graph-based generation tasks are difficult to create lengthy and coherent summaries. Nevertheless, the strong language ability obtained by the giant DL-based language models has been successfully applied in many summary tasks, which could be a potential improvement for traditional semantic-based fashion [17]. In the end, due to the rise of graph neural networks, it is possible for the neural architectures to process the structured representation like the semantic graph in an end-to-end training [18, 19]. This idea might further combine semantic graph information and deep learning models.

According to the approach for choosing the salient content and organizing the generated summary, summarization could be categorized into two approaches: 1) extractive and 2) abstractive. This paper will mainly introduce abstractive summarization because it is more difficult and requires rephrasing the original meaning with new words not featured in the source text. Comparatively, the extractive model is to turn the summarization task into marking the piece of the original text to be extracted. Although the semantic graph might also benefit extraction tasks; but compared to the abstractive one, extractive summarization is much easier thus hard to show the advantage of combining DL and semantic representation.

In this paper, some selected summarization works with the improvement from the combination of structured semantic representation with deep learning models would be reviewed. We mainly discussed the benefit(s) in the three aforementioned potential aspects for abstractive summarization. We identify the opening challenges lying in the philosophy of coupling semantic graphs with DL models and discussing future trends in this research area. To conclude, the paper answer this core question: How the integration of semantic graphs and DL-based models summarize abstractively better?

The paper is organized as follows: Section 2 describes the basic background for this field. In Section 3.1, we introduced the first study in the feasibility of using semantic representations to build the abstractive summarizer. This work did not use any powerful deep neural models; however, noticeably, it was the first work that utilized semantic graphs for abstractive summarization works. Then, we categorize each work by the aspect that they contributed in. Section 3.2 focuses on improving the content richness by the various approaches to merge semantic graphs and deep neural nets. In 3.3, we reviewed the models generating more grammatical and readable summaries by using the semantic graph to guide the neural models in the NLG phase. The 3.4 will address the improvement regarding interpretability for DL-based summarizers. Finally, we point out the challenges and future direction for new researchers and the conclusion.

2 Background

This section briefly introduced the background information for understanding this review, which includes the introduction on baselines, datasets, and evaluation methods.

Classical Approaches & Baselines. According to [2], traditional abstractive summarization methods could be classified as two categories: *structure* based method involving predefined prior knowledge (e.g., rules, ontology, etc.) and *semantic* based methods (e.g., based on predicate arguments and semantic graphs) [16, 17, 20, 21]. Recently, deep neural nets [6, 7, 22, 23, 24, 25] have been widely employed in the abstractive summarization and they have since become the state-of-the-art [1]. Especially, the encoder-decoder architectures with the attention mechanism achieve promising results [26, 6] and compared as the baselines for many following works.

Summarization Datasets. There are many annotated datasets with the alignment between original articles and their summaries, which are popular for the abstractive summarization. One commonly experimented dataset is the CNN/Daily Mail [27] which contains online news articles with multi-sentence human-written summaries as gold reference. The DUC 2004 (Document Understanding Conference) dataset is also constructed by the news-summaries pairs like CNN/DM. It contains 500 news articles with four human-written summaries per each article as ground truth. Noticeably, the annotated summaries for CNN/DM and DUC 2004 are all fully grammatical, and there might be more than one sentence in the ground-truth summary. Comparatively, the ground-truth summaries could probably not be the full sentence, such as the Gigaword [28] for the headline summarization [5]. It is also worth mentioning that some datasets for semantic graph parsing provide human-written summaries for each article. These datasets are also possible for summarization tasks, especially for the semantic-based text summarization [17]. For example, the sample in the AMR Bank [29] dataset contain one paragraph with a corresponding golden reference in both summary and AMR annotation.

Common Analysis & Evaluations. The most popular metrics for the automatic evaluation for text summarization is the *Recall-Oriented Understudy for Gisting Evaluation*, or ROUGE in short [30]. It is the fraction where the numerator counts words overlapping between the generated summary and the ground truth. The denominator is the length of the gold reference. Intuitively, higher ROUGE means the summary contains more same content in the ground truth. The length of counting overlap could be different. For example, the ROUGE-1 (R-1) counts the unigram (single word) overlap; instead, we could also consider the bigram overlap named ROUGE-2 (R-2). The ROUGE-L (R-L) identifies longest co-occurring in sequence n-grams [31]. To be noticed, ROUGE has limitations as evaluation in summarization. It only counts the overlap of content, but the rephrasing in abstractive summarization might produce the summary with same meaning but without many overlappings with ground truth. Thus, most works in this area further use the human evaluation to assess the generated summary. The criteria normally include informativeness, fluency, grammaticality, and so on [32].

3 Literature Review

3.1 The First Study: *Toward Abstractive Summarization Using Semantic Representations*

Before the giant end-to-end training DL models dominate the summarization field, the pipeline processing procedure was popular. In 2015, Liu et al. [17] firstly proposed an abstractive summarization framework employing the AMR semantic Graph representation. This model

Node Features	Concept	Identity feature for concept label
	Depth	Concept freq in the input sentence set; one binary feature defined for each frequency threshold $\tau = 0/1/2/5/10$
	Entity	Two binary features indicating whether the concept is a named entity/date entity or not
Edge Features	Label	First and second most frequent edge labels between concepts; relative freq of each label, binarized by 3 thresholds
	Freq	Edge frequency (w/o label, non-expanded edges) in the document sentences; binarized using 5 frequency thresholds
	Position	Average and foremost position of sentences containing the edge (without label); binarized using 5 position thresholds

Table 1: Selected features functions for evaluating the score of subgraph prediction in [17]

composed three separate stages: 1) *Source Graph Construction*: a preprocessing step where some rules were adopted to construct the *source graph* for the given document. 2) *graph2graph* summarization stage: a trainable structural transformation were adopted to summarize the source semantic graph into the summary graph; 3) *graph2text* generation stage: a text generator transformed the summary graph into text form. The authors especially focused on the graph2graph stage by treating the summarization as a graph-level reduction or structured prediction. Their experiments on the standard AMR bank [29] revealed that the summarizer based on the graph-level structured prediction achieved 51.9 (using golden reference in AMR parsing), 51.2 (using JAMR parser, a state-of-the-art AMR parser [33]) on ROUGE-1 score. Comparatively, Oracle performance on ROUGE-1 was in the range of 89.1 when using gold-standard AMR annotations and 87.5 for JAMR.

To construct document-level semantic graph based on sentence-level AMR graph [34], some extra processing steps would be required. Firstly, they combined all coreferent nodes in the sentential AMRs as one entity node in the source graph. Then, the *sentence conjunction* would create a global ROOT node for the given document, and it connected all roots for the sentential AMRs. In the last step, the *graph expansion* would add the extra edges to constructed a fully-connected graph where all sentential roots are inter-connected. The authors assumed that the most salient semantic information would be reserved and further support the summarization after all preprocessing steps.

The authors treated the summarization as selecting a subgraph from the source graph, which could be formulated as a structured prediction problem. The general objective was to reserve the salient content while avoiding changing the meaning, keeping the representation concise. Mathematically, they quantified a score for evaluating the predicted subgraph, with two terms representing node score and edge score, respectively. Afterward, the weight parameters θ, ϕ could be trained by maximizing this objective score using a gradient optimizer. The objective is shown as followed,

$$score(V', E'; \theta, \phi) = \sum_{v \in V'} \theta^T \mathbf{f}(v) + \sum_{e \in E'} \phi^T \mathbf{g}(e) \tag{1}$$

where the (V', E') formulated the predicted subgraph, and $\mathbf{f}(v)$ and $\mathbf{g}(e)$ denoted the hand-engineering feature functions of node v and edge e, respectively. Part of the designed features was listed in Table 3.1. Finally, an integer linear programming (ILP) was introduced to decode the summary graph under some constraints ensuring the generated graph was valid. In this decoding step, the θ and ϕ would be treated as constants, and the model searched and found the best subgraph result. In the end-stage for generating summary, the authors produced a bag of words as the output, constituted by the most frequently aligned word span for each concept node in the summary graph.

To conclude, this paper conducted the first study and has inspired a new direction for subsequent abstractive summarization research: using the semantic graph to better represent the document's meaning, thus improving the summarizers. Semantic graphs not only contain semantic information which is beneficial for the summarization, but also structured information

	Automatic Eval.	Human Eval.	Improvement	Contribution	Application	Training Fashion
AMRSum [17] (2015)	R-1, prediction Acc.	No	-	First Study	NLU & NLG	pipeline
AMR+TreeLSTM [36] (2016)	R-1, R-2, R-L	No	Yes	ROUGE AE	NLU	Pipeline
LinearAMR+Seq2Seq [37] (2018)	R-1, R-2, R-L	Yes	Yes	Fluency, Grammaticality	NLU	Pipeline
GuidedNLG [38] (2018)	R-1, R-2, R-L	Yes	Yes	Fluency, ROUGE AE	NLG	End-to-End
NEG+Seq2Seq [16] (2019)	R-1, R-2, R-L	No	Yes	ROUGE AE	Pre-processing	Pipeline
LG+Seq2Seq [16] (2019)	R-1, R-2, R-L	No	Yes	ROUGE AE	Pre-processing	Pipeline
GenParse [39] (2020)	R-1, R-2, R-L	Yes	Yes	Grammaticality, Richness, RAE	NLG	End-to-End

Table 2: Summary of the reviewed works. Improvement refers whether the method has some improvements on previous works, with specific aspects in Contribution. ROUGE AE stands for the ROUGE-based Auto Evaluation. Application indicates where to apply semantic graph in a DL framework. Training Fashion shows how the model be trained as its design in the literature.

(both locally and globally) for the given document. The structured information in the graph representation for text has also been demonstrated to help the deep neural model better capture the latent dependencies among words, compared with the input only consisting of the token sequence [32, 35]. Hence, it is reasonable to make the combination of semantic graphs and current state-of-the-art neural summarizers. Based on this article, future work that can be done in this direction includes 1) The hand-engineering features in Table Table 3.1 for calculating the scores might be further replaced by the automatically learned features via end-to-end trainable machine learning algorithms; 2) The simple linear model could be further improved by a more complex one, e.g., the deep neural networks with powerful expressive capability. 3) this article did not generate a summary that follows natural language grammar but a bag of words instead, which could be further improved by a powerful neural networks-based language model.

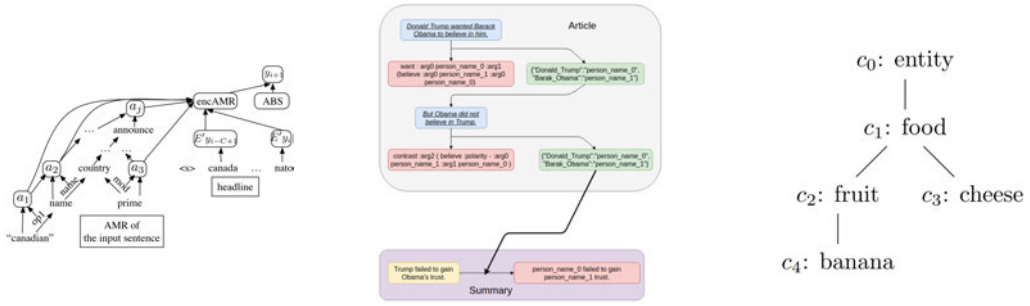
3.2 Improvement in the Quality of Summary

Unlike image or text, semantic graphs are generally expressed as graph structures or stored with a structured scheme. Hence, the semantic graph could not be directly merged and computed by neural networks to serve the summarization. The majority of past related works focused on finding a way to corporate the semantic graph information into a neural architecture, with the maximum utilization of both semantic and structural information to generate high-quality summaries. In this section, we mainly discussed the papers focusing on exploring those various information fusion approaches to cope with the multi-modality problem of semantic graph data, thus leading to improve the quality and informativeness of the generated summary. We list the performance reported in each reviewed literature in Table 3.2.

One way of thinking is to modify the input layer of the neural networks to fit with the graph structure data, just like we design 2D convolutional networks for image data and recurrent neural networks for sequence data. Inspired by previous structured neural networks (e.g., tree structure neural networks [40]), [36] utilized the tree-LSTM to directly encode the AMR graph into neural model as shown in Figure 1 (a). Specifically, based on the traditional encoder-decoder architecture, in addition to the encoder for encoding token sequence, they designed an AMR encoder. This AMR encoder was a tree neural networks which could encode nodes (concept) and edges (relationship) in AMR graph, and generated a fixed-length encoding vector to represent the output of the AMR parser. The prediction distribution with such information fusion mechanism $\mathbf{y}$ could be formulated as,

$$p(\mathbf{y}_{i+1}|\mathbf{X}, \mathbf{Y}_{C,i}) \propto exp(LM(\mathbf{Y}_{C,i}) + encDOC(\mathbf{X}, \mathbf{Y}_{C,i}) + encAMR(\mathbf{A}, \mathbf{Y}_{C,i})) \tag{2}$$

where the LM, $encDOC$ and $encAMR$ were the language model, RNN encoder with attention for the sentence and the tree encoder for the AMR, respectively. $\mathbf{X}, \mathbf{Y}$ represented the aligned



(a) (b) (c)

Figure 1: Illustrations for three discussed models: (a) ABS+AMR [36]; (b) linearization and anonymization for AMR [37]; (c): Taxonomy based concept generalization [16]

pair of the indicator vectors that indexing the appearing words in the original sentence and summarized headline, where $|\mathbf{Y}| < |\mathbf{X}|$. Specially, the subscript of C, i denoted the sub-sequence in $\mathbf{Y}$ from y_{i-C+1} to y_i. Matrix $\mathbf{A}$ represented the list of hidden states $\mathbf{a}_j$ for all node j in the AMR graph. Finally, the results on DUC-2004 and Gigaword datasets shown the benefit of incorporating AMR in such a mechanism with an improvement of around 0.5 in terms of ROUGE-1 (R-1), R-2, and R-L, which proved the contribution of AMR (AMR+ABS) [36] on pure DL-based model (ABS) [5].

Apart from modifying the neural architecture, some semantic graph (e.g., AMR) provides the analytical method for linearizing, i.e., converting the graph representation into a canonical sequence representation via a close-form solution. In 2018, [37] proposed to integrate AMR graph by linearizing the graph into sequential data (Depicted in Figure 1(b)). Similar to [17], some preprocessing was done to convert the original AMR into document level. Firstly, the author adopted the *linearization* and *anonymization* following [41], so that the concept corresponding to the same entity across the article was assigned the same anonymization token and generated a dictionary representation. After the preprocessing, the authors adopt the conventional Seq2Seq model with the attention mechanism to produce the summary following a standard supervised learning fashion. The input and output sequence for $t-th$ document would be $I_t = [a_0, a_1, ..., a_n]$ and $O'_t = [s_0, s_1, ..., s_m]$, where the a_j and s_j were tokenized linearized AMR and summarized tokens, respectively. Same with the common consideration in abstractive summarization tasks, they also employed the *coverage* method proposed by [6] in the decoder's generation. They also provided a post-processing step in which the dictionary's corresponding entity replaced the anonymous tokens. The experiment results revealed that the full model outperformed the baseline [6] in both ROUGE-based automatic evaluation and human evaluation in terms of meaningful richness, fluency, and grammaticality. It showed that the AMR semantic graph could make a positive contribution to summarization tasks. Moreover, the semantic graph's linearizing is a computationally efficient and effective approach for integrating the information from the semantic graph to neural summarizers.

The aforementioned methods directly treated the semantic graphs as the input for the neural model. However, the hierarchical information in some semantic graph can also be used to preprocess the text, e.g., *content generalization*. In 2019, [16] proposed a novel text generalization framework by utilizing the semantic and hierarchical structural information in concept taxonomy as shown in Figure 1(c). The intuition behind was based on the frequency of tokens: because the machine learning systems normally requires a large number of training samples to perform the predictions accurately, it was reasonable to consider replacing the low-frequency

	CNN/DM			DUC-2004			Gigaword		
Method	R-1 F-score	R-2 F-score	R-L F-score	R-1	R-2	R-L	R-1	R-2	R-L
AMR+TreeLSTM (2016) [36]	-	-	-	28.8	7.83	23.62	31.64	12.94	28.54
LinearAMR+Seq2Seq (2018) [37]	38.15	**17.87**	34.84	-	-	-	-	-	-
NEG+Seq2Seq (2019) [16]	-	-	-	28.73	9.87	**26.12**	46.3	23.88	**43.94**
LG+Seq2Seq (2019) [16]	-	-	-	**28.89**	**10.1**	24.46	**46.34**	**24.02**	43.65
Baselines									
PGN+Cov (2017) [6]	**39.53**	17.28	**36.38**	†27.56	†8.9	†25.2	†44.35	†22.43	†41.87
ABS (2015) [5]	-	-	-	26.55	7.06	22.05	30.88	12.22	27.77

Table 3: Dominant comparison among discussed models with various approaches to merge structured semantic representation with deep learning methods. †: the results of PGN+Cov model on DUC-2004 and Gigaword datasets are reported by [16].

words or tokens with their generalized token, i.e., hypernym (for example, in Figure 1(c), *fruit* is the hypernym of *banana*). They proposed two strategies of generalization. The first one was the generalization for the name entities (NEG) in which the named entities would be replaced by their hypernym in the entities set. For instance, given the sentence "*Anbul lives in Liverpool.*" would be generalized into "*_Person_ lives in _location_*", if the entities set were simply defined as $E = \{person, location\}$. The second strategy utilized the hierarchical structural information in the word taxonomy graph, namely level generalization (LG). A hyperparameter d determined the generalized degree, which determined each concept node's generalization strength, expressed as the depth of the taxonomy tree. For example, if $d = 2$, the sentence "*banana is delicious*" might be generalized by "*fruit is delicious*". After the generalization, the generalized text would be summarized by the classical Seq2Seq with attention model [6]. Similar to [37], as the input sequence has been changed. There was a need to do the post-processing on the predicted summary to transform the generalized tokens back to more specific entities as the final prediction. The experiments on Gigaword [28] demonstrated the effectiveness of this semantic-based generalization, with the improvement at around 2 on R-1, R-2 and R-L evaluation compared with the baseline [6]. The comparison on DUC 2004 [42] showed that the DL model with semantic-based generalization also outperformed the baseline with pure deep learning method [5], and achieved the comparable performance with the state-of-the-art method [43] on R-2 and R-L metrics.

In conclusion, the semantic graph could improve the pure neural nets-based summarizers such as [6, 5] in terms of ROUGE-based evaluation. The sampled case studies in [36, 16] also manifested such consolidation of semantic graphs with neural summarizers help to improve the informativeness and meaningness of generated summaries. Nevertheless, the comparison among all discussed models reveals that there is still an open-end question for finding a more efficient approach to inject the semantic graph into a sophisticated neural system. The previous works [37, 36] attempted to combine them via an end-to-end fashion; however, their ROUGE performances were not as comparable as simply using the hierarchical semantic information to generalize the text in preprocessing as shown in Table 3.2. Unfortunately, both [36] and [16] did not perform the human evaluation in meaningfulness, fluency, and grammar compliance like [37], the manual analysis and comparison in the summaries generated by the non-semantic model and semantic model might need to be further investigated.

3.3 More Readable and Fluent

Since the structured meaning representation usually contains the semantic meaning and entails syntactic information, previous studies attempted to use the semantic graph to guide the generation of summaries to be more readable and grammatical. In this section, we briefly reviewed the papers placing their contribution on the generation phase.

(a)

t	y^T	Stack
1	–	R (root node)
2	GEN(*a*)	R *a*
3	GEN(*man*)	R *a man*
4	REDUCE-L	R *a*←*man*
5	GEN(*escaped*)	R *a*←*man escaped*
6	REDUCE-L	R *a*←*man*←*escaped*
7	GEN(*from*)	R *a*←*man*←*escaped from*
8	GEN(*prison*)	R *a*←*man*←*escaped from prison*
9	REDUCE-L	R *a*←*man*←*escaped from*→*prison*
10	REDUCE-R	R *a*←*man*←*escaped*→*from*→*prison*
11	REDUCE-R	R *a*←*man*←*escaped*→*from*→*prison*

(b)

Figure 2: Illustration for the decoding of the GenParse model in [39].

As aforementioned, [17] utilized the heuristic approach in the summary generation step. Their summary generator simply generated bags of words that were not readable and following any natural language grammar. To improve the readability of generated summary, [38] further adopted the DL model (Seq2Seq with attention) to serve as the generator. They explored two methods to design the generator. The first one was the unguided generation from AMR, which simply linearized the summary graph G' from [17]. Then the Seq2Seq model would learn by the traditional supervised manners. The first method was similar with [37], but linearizing the summary graph instead of the original AMR representation for a given document. The second one was to adopt AMR to guide the language generation for the Seq2Seq model. The author selected top k sentences with highest similarity with the AMR graph, which was calculated by the longest common subsequence between the linearized version of two graphs. The pruned document with k selected sentences was defined as *side information*. Subsequently, an interpolated tri-gram language model would be estimated by Maximum Likelihood on the side information,

$$P_{side}(x_j|x_{j-3}^{j-1}) = \lambda_3 P_{LM}(x_j|x_{j-3}^{j-1}) + \lambda_2 P_{LM}(x_j|x_{j-2}^{j-1}) + \lambda_1 P_{LM}(x_j|x_{j-1}) \tag{3}$$

where the interpolated hyperparameters λ_i is tuned in development set. The P_{LM} denotes a basis trigram language model estimated on the pruned documents. Finally, the predictive unnormalized score to generate summary was formulated as,

$$score = logP_{s2s}(y_j|y_{<j}, z) + \Phi * log(\frac{P_{side}(y_j|y_{<j}, z)}{P_{s2s}(y_j|y_{<j}, z)} + 1) \tag{4}$$

where Φ was the hyperparameter to control the strength of AMR guidance in NLG. P_{s2s} was the predictive distribution in the unguided generation. z was the hidden representation in the encoder-decoder architecture. Their experiments and the performance comparison with [17] revealed that the guided NLG outperformed the unguided NLG with a considerable margin. The Guided NLG achieved 70.7 and 64.9 in R-1 and R-2 figures, while the unguided model only had 68.6 and 61.3, respectively. More importantly, the guided generation achieved a 2.66/6.00 score in the human evaluation regarding fluency, which is determined by whether the sentences were grammatical and natural. Comparatively, the unguided generation only earned 2.16 in this test. However, the author also pointed out the grammatical mistakes and repetition are still a severe problem.

To tackle the grammatical incorrection problem in the generated summary for the current neural model. [39] suggested to consider adding a syntactic or semantic decoder in the language generation phase. The decoder would perform either syntactic or semantic parsing together with the original decoder to generate the summary rather than only served as the summary generator in previous studies. Although the author used the syntactic decoder in this paper's

experiments, they also pointed out that since the semantic representation often entails grammatical information, it was feasible to replace the decoder with a semantic-oriented one. In details, the authors converted the language generation tasks into sequence prediction for the action set $\mathbf{y}_t^\tau = \{REDUCE-L, REDUCE-R, GEN\}$. $REDUCE-L$ and $REDUCE-R$ were reducing actions in the standard shift-reduce parsing framework. Specifically, the GEN were the action to generate token as summary, and added the new token into the stack that stored the tokens waiting for parsing. One example for inference stage to generate summary sentence "*a man escaped from prison*" was shown in Figure 2(b). The joint objective for training could be formulated by,

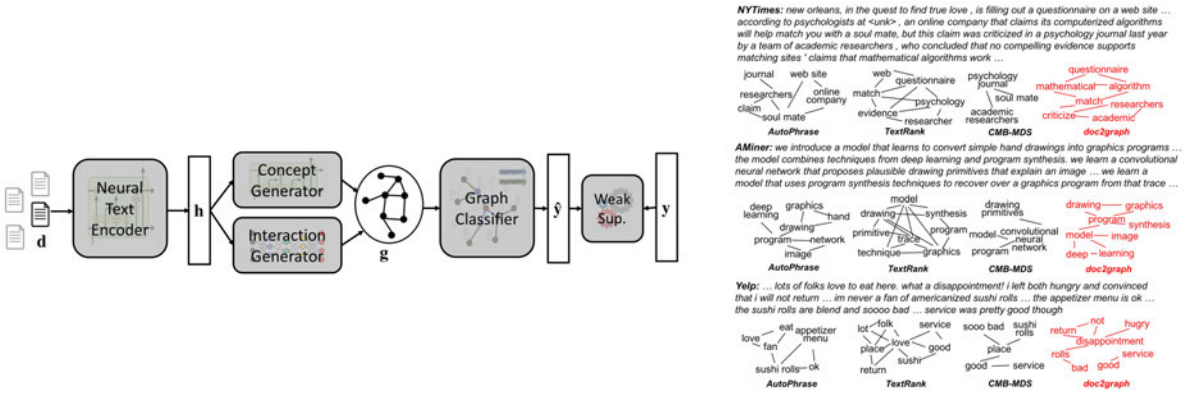
$$logP(\mathbf{y}^\tau|\mathbf{x}) = \underbrace{[\sum_t logP(\tilde{\mathbf{y}}_t^\tau = o|\tilde{\mathbf{y}}_{<t}^\tau, \mathbf{x})]}_{Parsing} + \underbrace{[\sum_{t:o_t=GEN} logP(\mathbf{y}_t^\tau = w|\mathbf{y}_{<t}^\tau, \mathbf{x})]}_{Summarization} \tag{5}$$

where first term represented the log-likelihood to be maximized for the parsing task, and the second term was for the summarization tasks. The $\mathbf{y}\tau, \mathbf{x}$ denoted the input and target sequence pair in one training sample. $\tilde{\mathbf{y}}_t$ was the predictive action of the model in $t-th$ step. o denotes the predictive action in a previously defined action set, and w was the generated word token. The convincing experiment results on four datasets, including Gigaword, NewsRoom, CNN/DM and WebMerge demonstrated their GenSUM model significantly improved the baseline's performance [6] on the auto evaluation via R-1, R-2 and R-L. In the human evaluation for grammaticality and meaningfulness, unlike previous work that only required human referees to score the summaries, this work introduced a competition mechanism for compared models, i.e., the human referee needed to rank the summaries generated by varied models (including humans). Among them, in terms of grammaticality, 12.7% referees believed that the GenParse model ranked first, which is 7.6 percent above the baseline. And, 42.4% of people ranked the summaries generated by GenParse in the second place among all models. Such convincing results suggested that the syntactic and semantic information could be useful for the NLG decoder to generate a readable and grammatical summary.

3.4 More Interpretable

Although deep learning models have achieved significantly better performance than traditional models on many complex NLP tasks, its black-box essence is still widely criticized. Especially on NLP tasks, although end-to-end data-driven systems can achieve state-of-the-art performance on tasks that require "understanding" such as text summarization and machine reading comprehension, DL-based models are still far from "understanding" of natural language [44]. One of the advantages of the semantic graph is its nice property in structured representation. Compared with flat vectors in the end-to-end DL models, it is easier for humans to understand. Therefore, understanding the learned "knowledge" by the structured representation of deep learning models becomes a research trend in the NLP community recently [1]. In this section, we discuss improving the interpretability of the end-to-end model by introducing structured representation for semantic relationships.

In 2020, [45] proposed the first end-to-end neural framework for downstream NLP tasks with the direct generation of a concept graph at the middle. The concept map is a structured semantic representation whose nodes are the concepts and the edge, indicating an interaction between two concepts. The architecture for *doc2graph* is shown in Figure 3(a). The networks computed the representation for the concept embedding and interactions in parallel. To compute the embedding for the concept, a Long Short Term Memory networks with attention was



(a) (b)

Figure 3: (a): Illustration for the doc2graph model [45], which generates concept map $\mathbf{g}$ inside the end-to-end framework. (b): Case study for the generated summary, and comparison for the visualization for the generated concept map with existing models.

introduced. The basic idea for generating concepts was the same as standard text embedding. The interactions embedding was more interesting, as it involved the challenge of converting text into graphs which were different data type. To make the graph generation fully differentiable in end-to-end framework, the concept graph would be defined as a symmetric adjacent matrix A where each element $A_{i,j}$ was the probability to connect concept i and j. Thus the matrix generation could be modeled by a two-layer feed-forward network, and the interaction vector $\mathbf{a}$ for each node (concept) would be,

$$\mathbf{a} = \sigma(\mathbf{W}_m^{(2)} ReLU(\mathbf{W}_m^{(1)}\mathbf{h} + \mathbf{b}_m^{(1)}) + \mathbf{b}_m^{(2)}) \tag{6}$$

where $\mathbf{W}_m, \mathbf{b}_m$ were trainable parameters, and superscript (j) indicated the parameters for $j-th$ layer. σ was the sigmoid function that constrained the output to be in the range $(0, 1)$ as a probability. The model was trained in an end-to-end fashion, and the loss was specified by the downstream task. The case studied in Figure 3(b) showed that compared with other text-to-graph algorithms, the graph representation generated by the text2graph model contains more accurate concept and meaningful interaction links. Although the author did not experiment on the summary task, the NYT dataset [46] used in this work was also a commonly used annotated dataset in the summarization [6]. They also pointed out that it was possible and easy to migrate the framework to other downstream tasks like summarization.

To conclude, this article proposes using intermediate structured semantic representations to enhance the interpretability of deep learning models in NLP. Although it does not use the external semantic graph to enhance the model as in the previous works, through the learning of downstream tasks, the summarizer following this framework could not only generate a summary, but also a semantic graph which can explicitly show humans the important information that the summarizer learned for the text. However, in the experiments, the authors did not introduce the analysis and human evaluation for the generated semantic graph. More research on the explainable deep neural networks for NLP is still worthy of further exploration.

4 Conclusion & Future Work

The previous studies made progress on improving the abstractive summarization by combining the semantic graph with deep learning models in many aspects. However, there a many open research questions which are yet to be solved. We list a few of them as followed,

- **Need of more efficient approach to integrating semantic graph into neural architecture.** Although the semantic graph enhances the performance of summarizers solely based on neural network systems. However, the most effective way to incorporate semantic graph information is still to use it to generalize the text in the preprocessing stage. The comparison among reviewed models in Table 3.2 shows that the generalization [16] perform better than methods that direct integrates the semantic graph into the end-to-end architecture on many datasets. Therefore, there is still a need to find a better design incorporating the semantic graph information while remaining the end-to-end property. The new-coming graph neural networks [47, 48] attracts much interest in the NLP community might empower the utilization of language graph data, e.g., semantic graph, in the deep learning-based NLP model.
- **Need of good human evaluation frameworks.** The main automatic evaluation metrics for summarization is ROUGE. Still, its shortcomings are obvious: its calculation depends on the word overlapping between generated summary with ground truth, which is especially limited for evaluating the abstractive summarizers that are required the ability to rephrase. Therefore, a lot of work in the past introduced human evaluation to judge the generated summary quality. However, there is currently no general prototype for human evaluation in abstractive summarization. Moreover, the existing evaluations could not specifically identify the semantic graph's contribution, thus better interpreting the role of semantic graphs in the performance improvement. Future work includes developing standard evaluation frameworks, e.g., more specific criteria for analyzing the improvement by adding semantic graphs into neural models.

Abstractive summarization is an interesting but difficult task involving both NLU and NLG to generate fluent, non-redundant, and informative summaries. It has attracted much interest among the community in recent years with the rise of deep neural networks and tremendous language data. This paper aims to present the recent progress in applying the structured semantic representation, especially in graph, to improve the neural networks-based abstractive summarization. The past studies revealed that both the solely-semantic-based summarizers and the deep learning-based summarizers have their shortcomings. However, these two schools are not independent but can be combined to complement each other. Specifically, we reviewed relevant literature in the past five years (2015-2020). We found that DL-based language models help the solely-semantic-based summarizer [17] to generate more readable summaries [38]. Interestingly, the semantic information of the semantic graph also helps the summarizer to generate a more meaningful and grammatical summary [39]. Besides, semantic graph enables deep learning models to capture salient semantic information better, thereby generating a more content-rich summary and achieved higher performance in both automatic and human evaluation [36, 37, 16]. At the same time, the structured property of the semantic graph also enhances the explainability of the deep learning model [45]. Future works include merging the semantic graph with current state-of-the-art neural models more efficiently and finding a better evaluation method to analyze the contribution of the semantic graph in abstractive summarization.

References

[1] Som Gupta and SK Gupta. Abstractive summarization: An overview of the state of the art. *Expert Systems with Applications*, 121:49–65, 2019.

[2] N. Moratanch and S. Chitrakala. A survey on abstractive text summarization. In *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, pages 1–7, 2016.

[3] Omri Abend and Ari Rappoport. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 77–89, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[4] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[5] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

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

[7] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, 2018.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[10] Yan Zhang, Zhijiang Guo, Zhiyang Teng, Wei Lu, Shay B. Cohen, Zuozhu Liu, and Lidong Bing. Lightweight, dynamic graph convolutional networks for AMR-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2162–2172, Online, November 2020. Association for Computational Linguistics.

[11] Christian F Doeller, Caswell Barry, and Neil Burgess. Evidence for grid cells in a human memory network. *Nature*, 463(7281):657–661, 2010.

[12] Jonathan St BT Evans. Heuristic and analytic processes in reasoning. *British Journal of Psychology*, 75(4):451–468, 1984.

[13] John T Wixted. Dual-process theory and signal-detection theory of recognition memory. *Psychological review*, 114(1):152, 2007.

[14] Jantien Van der Vegt, Hans Buffart, and Cees Van Leeuwen. The structural memory: A network model for human perception of serial objects. *Psychological Research*, 50(4):211–222, 1989.

[15] Elizabeth S. Spelke and Katherine D. Kinzler. Core knowledge. *Developmental Science*, 10(1):89–96, 2007.

[16] Panagiotis Kouris, Georgios Alexandridis, and Andreas Stafylopatis. Abstractive text summarization based on deep learning and semantic content generalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5082–5092, Florence, Italy, July 2019. Association for Computational Linguistics.

[17] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[18] Zimeng Qiu, Eunah Cho, Xiaochun Ma, and William Campbell. Graph-based semi-supervised learning for natural language understanding. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 151–158, 2019.

[19] Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. Heterogeneous graph neural networks for extractive document summarization. 2020.

[20] Ritwik Mishra and Tirthankar Gayen. Automatic lossless-summarization of news articles with abstract meaning representation. *Procedia Computer Science*, 135:178 – 185, 2018. The 3rd International Conference on Computer Science and Computational Intelligence (ICCSCI 2018) : Empowering Smart Technology in Digital Era for a Better Life.

[21] Kaiqiang Song, Logan Lebanoff, Qipeng Guo, Xipeng Qiu, Xiangyang Xue, Chen Li, Dong Yu, and Fei Liu. Joint parsing and generation for abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8894–8901, 2020.

[22] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany, August 2016. Association for Computational Linguistics.

[23] Junyang Lin, Xu Sun, Shuming Ma, and Qi Su. Global encoding for abstractive summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 163–169, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[24] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[25] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

[26] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[27] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701, 2015.

[28] Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-Scale Knowledge Extraction*, AKBC-WEKEX '12, page 95–100, USA, 2012. Association for Computational Linguistics.

[29] Kevin Knight, Lauren Baranescu, Claire Bonial, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, and Nathan Schneifer. Abstract meaning representation (amr) annotation release 1.0. *Web download*, 2014.

[30] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157, 2003.

[31] Chin-Yew Lin and Franz Josef Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612, Barcelona, Spain, July 2004.

[32] Luyang Huang, Lingfei Wu, and Lu Wang. 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, Online, July 2020. Association for Computational Linguistics.

[33] Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[34] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, pages 178–186, 2013.

[35] Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. Every document owns its structure: Inductive text classification via graph neural networks. *arXiv preprint arXiv:2004.13826*, 2020.

[36] Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. Neural headline generation on Abstract Meaning Representation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1054–1059, Austin, Texas, November 2016. Association for Computational Linguistics.

[37] Rangu Goutham. *Using Abstract Meaning Representation of Text for Summarization*. PhD thesis, Heriot-Watt University, 2018.

[38] Hardy Hardy and Andreas Vlachos. Guided neural language generation for abstractive summarization using abstract meaning representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 768–773, 2018.

[39] Kaiqiang Song, Logan Lebanoff, Qipeng Guo, Xipeng Qiu, Xiangyang Xue, Chen Li, Dong Yu, and Fei Liu. Joint parsing and generation for abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8894–8901, 2020.

[40] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. 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 1556–1566, 2015.

[41] Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, 2017.

[42] Paul Over, Hoa Dang, and Donna Harman. Duc in context. *Information Processing Management*, 43(6):1506 – 1520, 2007. Text Summarization.

[43] Yang Gao, Yang Wang, Luyang Liu, Yidi Guo, and Heyan Huang. Neural abstractive summarization fusing by global generative topics. *Neural Computing and Applications*, pages 1–10, 2019.

[44] Emily M. Bender and Alexander Koller. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online, July 2020. Association for Computational Linguistics.

[45] Carl Yang, Jieyu Zhang, Haonan Wang, Bangzheng Li, and Jiawei Han. Neural concept map generation for effective document classification with interpretable structured summarization. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1629–1632, New York, NY, USA, 2020. Association for Computing Machinery.

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

[47] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[48] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.