Expert Systems With Applications 121 (2019) 49–65

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Review

Abstractive summarization: An overview of the state of the art

Som Gupta [a,*], S. K Gupta [b]

[a] AKTU Lucknow, UP 226031, India
[b] BIET Jhansi, UP 284128, India

ARTICLE INFO

Article history:
Received 3 September 2018
Revised 28 October 2018
Accepted 6 December 2018
Available online 7 December 2018

Keywords:
Abstractive summarization
Concept finding
Semantic-Based summarization
Ontology-Based summarization
Deep learning

ABSTRACT

Summarization, is to reduce the size of the document while preserving the meaning, is one of the most researched areas among the Natural Language Processing (NLP) community. Summarization techniques, on the basis of whether the exact sentences are considered as they appear in the original text or new sentences are generated using natural language processing techniques, are categorized into extractive and abstractive techniques. Extractive summarization has been a very extensively researched topic and has reached to its maturity stage. Now the research has shifted towards the abstractive summarization. The complexities underlying with the natural language text makes abstractive summarization a difficult and a challenging task.

This paper presents a comprehensive review of the various works performed in abstractive summarization field. For this purpose, we have selected the recent papers on this topic from Elsevier, ACM, IEEE, Springer, ACL Anthology, Cornell University Library and Google Scholar. The papers are categorized according to the type of abstractive technique used. The paper lists down the various challenges and discusses the future direction for research in this field. Along with these, we have identified the advantages and disadvantages of various methods used for abstractive summarization. We have also listed down the various tools which have been used or developed by researchers for abstractive summarization. The paper also discusses the evaluation techniques being used for assessing the abstractive summaries.

1. Introduction

The exponential growth of information due to the popularity of web environment has lead to the need of automatic summarization in order to reduce the effort and time while finding the concise and relevant information according to the query. Summarization is to reduce the content of the text while preserving the meaning of the text. Various ways to write the same thing has made this topic an interesting topic among the researchers. There has been a lot of work done in the area of automatic summarization in the recent years. According to how the content is selected and organized in the summary, the summarization techniques are categorized into extractive and abstractive techniques. Extractive summarization is to find out the most salient sentences from the text by considering the statistical features and then arranging the extracted sentences to create the summary. Abstractive summarization, on the other hand is a technique in which the summary is generated by generating novel sentences by either rephrasing or using the new words, instead of simply extracting the important sentences. Internal representation of the text is created by analyzing the semantic information about the text and with the deep analysis and reasoning, new sentences are generated out of the original text (Rachabathuni, 2017). On the basis of the number of documents which are considered for summarization, they are classified into single-document and multi-document summarization. On the basis of how much information is to be summarized, they are classified into generic, where the summary of the whole text is obtained and query-focused summarization, where only the summary according to the context which is specified by user is obtained. And, on the basis of content, they are classified into indicative, where focus is on telling what the text is about and informative summarization, where the main content of the text is extracted by analyzing the original text. There has been a lot of research in extractive summarization but extractive summaries lack in terms of cohesion, readability and other quality factors due to the presence of dangling anaphora problem. Also, the extractive summaries are usually very different from the human-written summaries (Yao, Wan, & Xiao, 2017).

* Corresponding author.
E-mail addresses: somi.11ce@gmail.com (S. Gupta), guptask_biet@rediffmail.com (S.K. Gupta).
URL: http://www.aktu.ac.in (S. Gupta)

https://doi.org/10.1016/j.eswa.2018.12.011
0957-4174/

From our survey, we have found that most of the works are on extractive summarization due to the simplicity to implement it. Whereas the complexity of natural language processing, makes abstractive summarization, a challenging task. But now research on extractive summaries has stagnated as they have achieved the peak performance (Mehta, 2016) and thus the attention has shifted more towards abstractive summarization and fusion of extractive and abstractive techniques. The need of time is to produce high-quality summaries which are grammatically correct, readable, coherent, concise, and information-rich. Abstractive summarization helps resolve the dangling anaphora problem and thus helps generate readable, concise and cohesive summaries. Abstractive summarization helps reduce the sentence size as it uses fusion to merge the sentences, thus helps achieve more non-redundancy in the summary as compared to the extractive summaries, where even the non-relevant part of sentence also gets included due to the fact that it extracts the sentences and arranges them. Abstractive summaries also give higher precision than the extractive summaries. Abstractive summarization techniques not only help get the good quality summaries for the textual data but also for the non-textual data and cross-language data.

We have used the citation-based search for identifying the relevant papers. To ensure the good quality of papers for our review purpose, we have used IEEE,[1] ACM,[2] Springer,[3] ACL Anthology,[4] Cornell University Library,[5] Elsevier[6] and Google Scholar.[7]

Most of the works in the field of abstractive summarization focus on the components like parsing, coreference resolution, construction and merging of semantic graphs, natural language generation, lexical chains and distributional semantics (Yao et al., 2017). Most of the abstractive techniques involve sentence compression, fusion (Belkebir & Guessoum, 2016; Nayeem & Chali, 2017) or revision for generating the final summary out of the selected sentences.

According to whether the structure of the text is considered or the semantics, the abstractive summarization techniques are categorized into structure-based and semantic-based techniques. Apart from structure and semantic-based, now a days deep learning has emerged as a new technique to model the abstractive summarization problem which can capture both the structural and semantic information of the text.

From the literature review, we have found that the number of papers which describe the extractive summarization techniques in detail is huge but there are very few papers and are mostly outdated in the field of abstractive summarization which lists down the various recent works done in this field in categorized and detailed manner. Separate papers are available for each individual topic like sentence compression, sentence fusion, summarization using deep learning but there is no paper which lists them together. So to address this issue, we have listed down the various works done till the mid of 2018 and categorized them into the predefined classification hierarchy. Our study will systematize and enhance the knowledge in this research field. We have also discussed the popular components of abstractive summarization system.

In this paper, we discuss the research trends in the abstractive summarization field. We give an overview of various abstractive summarization techniques available. We discuss the tools that have been used for creating abstractive summaries. We also discuss the evaluation measures used for assessing these summaries. We have identified the challenges lying in the field of abstractive summarization and discussed the future trends in this area of research.

Our paper answers the following questions:

- How the abstractive summarization techniques are classified
- What are the various recent works done in this field according to the summarization technique
- What are the various tools which have been used to create the abstractive summaries
- How the results vary among the various techniques
- What are the famous data-sets which have been used for evaluating and performing the abstractive summarization task
- What are the various challenges and open research problems lying in this research field

The paper is organized as follows:- Section 2 describes the various abstractive summarization techniques. Section 3 discusses some of the common steps which are involved while creating the summaries. Section 4 discusses the various tools used for performing abstractive summarization. Section 5 points out the challenges and future direction for new researchers. And Finally the conclusion.

2. Existing approaches to abstractive summarization

Moratanch and Chitrakala (2016), Khan (2014) and Dineshnath and S.Saraswathi (2018) have classified the abstractive summarization approaches broadly into the structure-based and semantic-based. After surveying the abstractive summarization research works, we have added one more approach, deep learning with neural networks to it. Structure-based approaches are those where the important information of the text is populated into the pre-defined structure to create the abstractive summaries. Structure-based approaches are divided into tree-based, template-based, ontology-based, lead-and-body phrase, graph-based and rule-based methods according to the structure used for creating summaries. Whereas Semantic-based approaches are those which take the text document as input, create the semantic representation of text and then feed this representation to Natural Language Generation system to create the final abstractive summary. They are divided into information-item-based, predicate-argument based, semantic-graph based and multimodal. Fig. 1 is the classification of abstractive summarization approaches. In the below sub-sections, we have discussed the various abstractive summarization methods.

2.1. Structure-based approach

Structure-based approaches find the most important information from the text and then use templates, rules, trees, ontology, etc to create the abstractive summaries. They are mostly used along with extractive, semantic-based or deep learning based approaches. Few of the examples are like Ganesan, Zhai, and Han (2010) have used templates along with the graph-based approach, Bartakke, Sawarkar, and Gulati (2016) used rules and ontology along with the semantic-graphs, Li (2015) used templates along with the semantic-based approach, Wang and Ling (2016) used neural networks along with the templates to create the abstractive summaries. Sometimes, these approaches are also used as the first step for text pre-processing like Nguyen and Phan (2009) extracted the important key-phrases from the text using ontology which then can be combined with some other approaches to create the abstractive summary. Below is the list of various methods being used for creating abstractive summaries by considering the structure of the text.

[1] https://ieeexplore.ieee.org/.
[2] https://dl.acm.org/.
[3] https://link.springer.com.
[4] http://aclweb.org/anthology/.
[5] https://arxiv.org/.
[6] https://www.elsevier.com.
[7] https://scholar.google.co.in/.

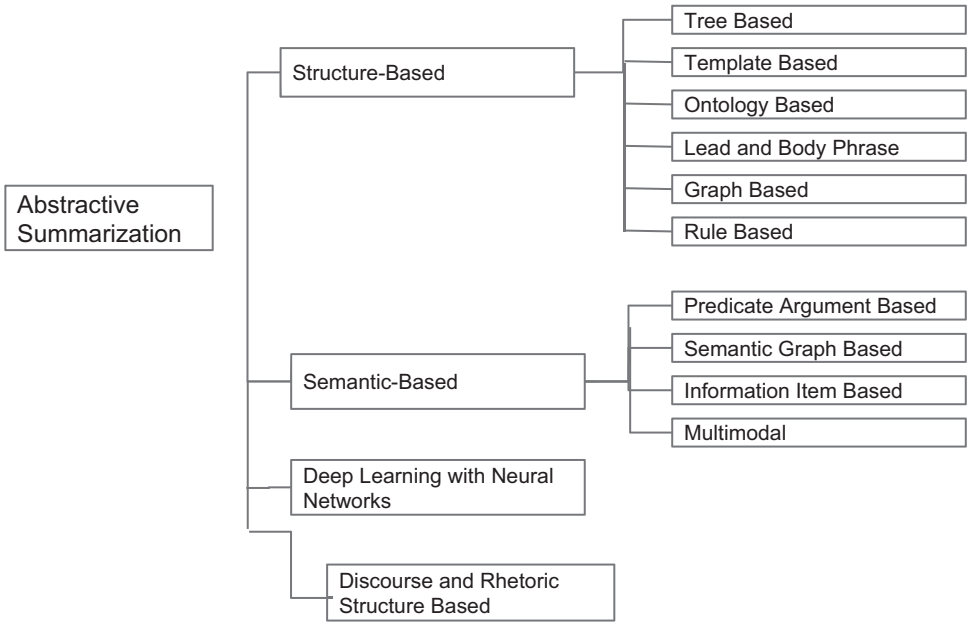

Fig. 1. Classification of abstractive summarization technique.

2.1.1. Tree based methods

In this approach, first the important text to be considered for the summary is extracted. Then, the similar sentences are identified from this text by using a shallow parser. Similar sentences of this text are then populated into the tree-like structure. Processing of trees is performed by either linearization(converting trees to strings), finding predicate-argument structure or sentence fusion to create the final abstractive summaries (Sunitha, Jaya, & Ganesh, 2016). There are various algorithms which help us perform these. For example, content theme intersection algorithm is one which is used to determine the common phrases by using predicate-argument structure, finding the basis tree (Barzilay & McKeown, 2005) by finding the centroid of the dependency tree and then augmenting these basis trees to obtain the sub-trees is also another way to evaluate the sentence. Common phrases are obtained and then either some information is added to them or are fed to the Natural Language Generation systems to create the new sentences and then are arranged together to create a summary.

Dependency tree is the most popular data structure, used to represent the text in a tree-form. Barzilay and McKeown (2005) used text-to-text generation to create the informative summaries. They represented the sentences using dependency trees and found the common information among sentences by processing the trees. They computed the fusion lattice by finding the intersection of sub-trees and then used tree traversals on them to produce the final sentence. One of the limitation of this approach is that it cannot capture the connections between the sentences without finding the intersected phrase between the sentences. Yousfi-Monod and Prince (2008) developed an approach called CoLIN, based on dependency tree pruning and linearisation while maintaining the semantic information, information content, grammatical correctness,and coherence in the summary. They performed the deep linguistic analysis of the text and focused on linguistic heuristics.

Kurisinkel, Zhang, and Varma (2017) used the partial dependency trees which are constructed from syntactic dependency trees by parsing of the text along with the recombination and transition based syntactic linearization to create multi-document abstractive summaries.

The use of Natural Language Generation systems for creating summaries help achieve the fluency and grammatical correctness. But they mostly do not consider context while finding the important phrases to be included in the summary (Kasture1, Yargal, Singh, Kulkarni, & Mathur, 2014). They mostly rely on parsing and alignment of parse trees (Khan et al., 2018). Dependency trees are constructed with the help of parsers. Thus, the performance of these methods highly relies on the parsers available, which limits its efficiency. They focus more on the syntax then the semantics.

2.1.2. Template based methods

In template-based methods, snippets are extracted using keywords or clues; and the extracted snippets are populated into templates to form the final summary. Because the structure is predefined, it helps create concise and coherent summaries. They rely on deep syntactic and semantic analysis of the text. It is one of the very popular method to create multi-document summaries. This method works well when the text is structured in some manner. But, because the rules and the patterns are manually defined, this method is very time-consuming and also requires a lot of manual effort (Alshaina, John, & Nath, 2017).

Harabagiu et al. (2001) used the template-based summarization and created a system called GISTEXTER to create the multi-document summaries where the templates are filled by following the patterns and the rules. Carenini, Ng, and Pauls (2012) used the templates to generate the natural language summaries for the evaluative corpus in their system SEA. Embar, Deshpande, K, Jain, and Kallimani (2013) used the domain-based template method to create the Kannada based abstractive summarizer. They used the hand-crafted information extraction rules to extract information to fill the templates. Oya, Mehdad, Carenini, and Ng (2014) proposed a multi-sentence fusion technique for creating abstract templates. To create templates, they have used the noun phrases along with the POS tagging and hypernyms. After finding templates, they have clustered them by extracting root verbs. They used word-graph for fusion of templates to generate the final summary. Gerani, Mehdad, Carenini, Ng, and Nejat (2014) have also used templates to generate the summaries by using the selected sentences obtained by considering the discourse structure of the text.

2.1.3. Ontology based methods

An ontology can be considered as the collection of entities and the relationship between them. An ontology along with the set of individual instances of class constitutes the knowledge base (Mohan, Sunitha, Ganesha, & Jaya Dr., 2016). Classes are the most important part of an ontology and represent the concepts. An Ontology defines the vocabulary set and also contains terms agreement, which helps remove any kind of ambiguity while finding the concepts. Ontologies are usually created by the domain experts and are mostly used to extract the concepts and relations from the text. These concepts are then used for creating the summaries (Mohan et al., 2016). One of the example of a domain-specific ontology is "WordNet". Ontologies help share a common knowledge among the community of interest. They enable the reuse of the knowledge. They help separate the domain and operational knowledge and thus help perform modifications easily when something related to domain changes. They have been used in number of fields like e-learning, user-related content analysis and image analysis (Baralis, Cagliero, Jabeena, Fiori, & Shah, 2013). Ontology-Based abstractive systems extract the information from the ontologies to create the summaries specific to user's need.

Many times the online vocabulary is limited, ontology helps in proper representation of the document (Rananavare & Reddy, 2017). Even the semantic representation of the information content can be improved a lot with the help of an ontology. Vocabulary normalization is performed to resolve the ambiguity when there are more than one synonym of a concept (Mohan et al., 2016).

Ontology is mostly represented in hierarchical structure like the directed acyclic graph, etc (Lloret, Teresa, Rom-Ferri, & Palomara, 2013). Resource Development Framework(RDF) and Web Ontology Language(OWL) are two major languages used for creating an ontology. DBPedia (Bizer et al., 2009) is one of the project which has structured the wikipedia data in the form of ontology. Ontologies help in query expansion and query search (Baralis et al., 2013). Tran, Cimiano, Rudolph, and Studer (2007) used the ontologies to build their question-answering system where they used the ontologies to convert the user question into the system query, so that system can understand the query specified by user. Ontologies help achieve topic completeness and non-redundancy in the summary (Hípola, Senso, Leiva-Mederos, & Domínguez-Velasco, 2014).

Zhang, Cheng, and Qu (2011) created an approach to create customizable ontology-based summaries where the length of summary and navigational preferences can be specified. RDF Graphs were constructed and the salience scores of RDF sentences were obtained to find the important sentences from the user's view. Hennig, Umbrath, and Wetzker (2008a) created an ontology based text summarization system where they applied the hierarchical classifier which mapped the sentences into the nodes of a predefined ontology. Lee, Chen, and Jian (2003) created an ontology-based approach for chinese news summarization by using the predefined ontology created for news articles. They used the ontology to infer the events from the news. Hípola et al. (2014) used the ontology in their system called Texminer to extract the important sentences with rhetorical structure, which summarizes the ports and coastal information. Ontology helped containing all the important concepts of the topics of coastal engineering. Ontology helped capture the semantic information of the concepts to facilitate the summarization in their system. dolfo Lozano-Tello and Gomez-Perez (2004) created a metric called OntoMetric to choose the appropriate ontology according to the requirement specifications. Nguyen and Phan (2009) used the Wikipedia ontology to extract the key-phrases which are the linguistic representation of the text, from the Vietanemese text. These keyphrases can be used for document summarization. Baralis et al. (2013) presented a multi-document abstractive summarizer which used Yago ontology for selection of sentences by the identification of entities, concepts and disambiguation.

2.1.4. Lead and body phrase methods

Here the insertion and substitution of important information-rich phrases technique are used in the lead sentence from the body text, which are called triggers (Sunitha et al., 2016). Creating the grammatically correct sentences by using this method is still an issue. Tanaka, Kinoshita, Kobayakawa, Kumano, and Kato (2009) created the lead and body phrase summaries of broadcast summarization. They revised the summaries by modifying the lead sentence by using insertion and substitution of phrases. They modified the lead sentence by adding information like "who", "where", "when", "how", etc to it.

2.1.5. Graph-based methods

Word Graph is the most popular graph data structure which is used for representing the text in these methods. Sentences are fused together to create the abstractive summaries. Sentence fusion helps remove the redundant part of the sentence and thus achieve non-redundancy. Word graphs work on the assumption that there will be lot of similar sentences in the text and this similarity will help fuse the sentences. But it is not necessary that it will be easy to find the similar sentences.

Barzilay and Lee (2003a) used the paraphrasing to generate the natural language summaries by the fusion of sentences. They used the graph-based structure to find structurally similar sentences, then identified the text which are paraphrases to each other and then generated the sentence.

Katja (2010) created the grammatically correct and informative summaries by finding the shortest path in the word-graphs. But as they ranked their sentences on the basis of informativeness, it lacked the linguistic quality. Mehdad, Carenini, Tompa, and Ng (2013) extended the work of Katja (2010) by creating the entailment graphs to eliminate the redundant information along with the word graphs to find the relevant sentences from the informal text of meeting transcripts, to create the abstractive summaries. They used the WordNet to find the relations between the words and used this information to merge the nodes of the graph. Information content and the grammatical fluency were chosen as the factors to decide the best paths of the graph and then used the generalization and aggregation to generate the abstractive summary.

Ganesan et al. (2010) created the informative, readable, concise and well-formed abstractive summaries for redundant opinions by using graph based data structure called word Graph. Their approach does not require domain knowledge and uses shallow NLP. Their opiniosis-graph captures the redundancy and helps discover the new sentences by identifying the lexical links.

Le and Le (2013) created the extractive summaries and then applied word-graph to reduce the sentence and combine the sentences. Niu, Chen, Zhao, Sun, and Atiquzzaman (2017) used the chunk graphs which are based on the word-graph to reduce the size of the graph. Oya et al. (2014) used the word-graph to merge the templates generated for meeting conversations.

Banerjee, Mitra, and Sugiyama (2017) used the word graphs along with the Integer Linear Programming(ILP) to create the multi-document abstractive summaries. They first identified the most important documents among the documents to be summarized with the help of LexRank, cosine similarity score and overall document collection similarity score. Then, they created the clusters of similar sentences among the important documents. Shortest paths are obtained by creating the word-graphs and ILP model is applied to find the sentences with maximum information and readability. ILP helps minimize the redundancy in the summary.

2.1.6. Rule-based methods

In this method, the rules and categories are fed into system, to find the meaningful candidates which are then used to create the summary. Here the text document is first categorized according to the terms and concepts present in them. Then the questions are formulated according to the domain of text document and then the answers are extracted from the document. Questions can be like: Where the event has occurred, when the event has occurred, who did the event, what was the impact of event, etc. These questions are usually answered by finding the Part of Speech of the terms and concepts in the text. By answering these questions, they are fed into some pattern which then help create the final abstractive summary (Kasture1 et al., 2014). But again as in template-based approach, here the rules are written manually and thus leads to wastage of time.

Table 2 lists down the advantages and disadvantages of structure-based approaches on page 15.

2.2. Semantic-based approach

Here first the semantic representation of the text is obtained by finding the information items, predicate-argument structure or creating semantic graphs. Then, this representation is fed to the Natural Language Generation system and by using the noun and verb phrases (Dineshnath & S.Saraswathi, 2017), the final abstractive summary is created. Content theme intersection algorithm is also used with the graphs to determine the common phrases by using predicate-argument structure.

2.2.1. Information item based(INIT) methods

Genest and Lapalme (2011) used the information item, the smallest unit of coherent information in the text to find the abstractive summaries. With information item, they mean the entities, their characteristics and the relationships or properties between them. They relied on Semantic Role Modeling, disambiguation, co-reference analysis, similarity analysis and predicate-logic analysis to find the information items. They used subject, verb and object triplets along with the time and location information to create the summary sentences. The general framework for INIT Based methods contains four modules namely information items retrieval where triplets are extracted using parser, sentence generation by using language generator, sentence selection by finding the top-rated sentences which is calculated on the basis of factors like document frequency, etc and summary generation using planning (Kasture1 et al., 2014).

2.2.2. Predicate-argument based approach

Predicate-argument structure means the verbs, subject and object, etc of the sentence. Predicate-argument structure is obtained for the sentences to represent them semantically. Then, the semantically similar structures are found from them by using similarity measures like edit distance measure, etc. Semantically similar structures are merged together by using some methods like K-means and hierarchical clustering (Alshaina et al., 2017). Features are extracted from the predicate-argument structures(PAS) and then they are scored. To maximize the salience scores of the sentences, optimization approaches like Integer Linear Programming(ILP) is used. High scoring PAS are selected and fed to the language generation system for creating the final summary. But to generate the new sentences out of them is a very difficult task (Li, 2015) from this method. Zhang, Zhou, and Zong (2016) proposed a cross-language abstractive summarization approach by using predicate-argument structure and merging them using Integer Linear Programming. They annotated the predicate-argument structures with semantic role modeling to obtain the summaries. To solve the dependency of human-written ontology , Alshaina et al. (2017) used the predicate-argument structure along with the semantic role modeling to find the semantic similarity between the arguments and created the multi-document abstractive summaries.

2.2.3. Semantic graph-based methods

It is one of the very popular way to represent the text on the basis of the semantic relations between the sentences in the text. Semantic properties include the ontological relations and the syntactical relations between the words. Ontological relations utilize the property of synonymy, hyponymy, hypernymy, etc whereas syntactical relations utilize the property of relationship among the words on the basis of subject-object-verb i.e. represented in terms of dependency tree and the syntactic tree (Joshi, Wang, & McClean, 2018).

Here the document is represented using a semantic graph. Nouns and verbs are represented as nodes of the graph, while relations between the nodes is represented by the edge. Khan et al. (2018) proposed the semantic graph-based approach for multi-document abstractive summary generation. Han, Lv, Hu, Wang, and Wang (2016) proposed a semantic graph-based model using FrameNet where each sentence is considered as a vertex and semantic relations between the sentences is represented as edges. Framenet is used to classify the sentences and find the relevance between the sentences and accuracy is achieved using Wordnet. Following are few famous semantic graphs which have been used for creating abstractive summaries.

- Rich Semantic Graphs: It is one of the way to represent the semantic information of the text in the form of graphs. Nouns and Verbs of the text represent the nodes and the semantic and topological relation between them correspond to the edges. Nouns and verbs of the text are obtained with the help of parsing or ontology. Moawad and Aref (2012) used the Rich Semantic Graphs(RSG) reduction approach for generating the single document abstractive summaries. They reduced the graph by using heuristic rules such as substitution, deletion, or fusion of nodes. Graph Nodes are obtained by using domain ontology, which consists of class hierarchy and relations. Bartakke et al. (2016) and Munot and Govilkar (2015) created the semantic graphs for each sentence and then used ontology to inter-connect the concepts to create the rich semantic subgraphs and then applied the heuristic rules to create the final reduced semantic graph and then fed it to the natural language generation tool to create the abstractive summary.
- AMR Graphs: Abstract Meaning Reprsentation(AMR) Graphs are labeled, rooted and directed acyclic graphs of the sentences. AMR Graphs provide the semantic representation of the sentence. Nodes of the graphs represent concepts and the edges represent the relationship between the concepts. AMR Graphs represent lot of information about the text like the named entities, semantic-role labeling, predicate-argument structure and co-reference relation (Foland & Martin, 2017). Concepts are mostly either the English words or PropBank[8] frames. Even though there are a number of parsers available for parsing but mostly JAMR parser[9] is used to perform the parsing; it performs parsing in two steps: first by identifying the key concepts using semi-Markov model, second by identifying the relations between the concepts by searching for maximum connected spanning graph.

  While generating the source graph for the sentence, as the similar concepts are merged together, so at the end of merging, an entity is represented in the graph only once. Thus, regardless

[8] https://propbank.github.io/.

[9] https://github.com/jflanigan/jamr.

of number of times, the concept has appeared in a sentence, it is represented as only one node. And this when applied to multiple sentences, lead to the final graph with no-redundancy (Liu, Flanigan, Thomson, Sadeh, & Smith, 2015). AMR Graphs are used in three representations namely in conjunction form to find the similarity between more than 1 AMR Graph, in PENMAN notation and as a graph data-structure (Viet, Sinh, Minh, & Satoh, 2017).

Vilca and Cabezudo (2017) used the AMR parsing for each sentence to generate the conceptual graph and incorporated Discourse-level information to identify the concepts. They used these concepts to generate the natural language summaries using SimpleNLG. Liu et al. (2015) used the graph-to-graph transformation for a domain-independent summary generation. They used Treebank to create AMR Graphs and then transformed these graphs to generate the summary. Liao, Lebanoff, and Liu (2018) created the AMR Graphs for each sentence and then merged them to create the summaries by using surface realization. They have used JAMR Parser to identify the concepts and the relationship between the concepts.

With the increasing research of Neural Networks for summarization purpose, they have been successfully applied to the task of AMR parsing also. Konstas, Iyer, Yatskar, Choi, and Zettlemoyer (2017) used the bi-directional LSTM based model at encoder side and a stacked LSTM model at the decoder side for AMR Parsing. Foland and Martin (2017) used the bi-directional Long Short Term Memory(LSTM) model to capture both the past and future sequence context. Viet et al. (2017) created a convolutional sequence-to-sequence model along with the graph linearization for AMR parsing which has outperformed than both the conventional AMR Parsing and the sentence generation approaches. They used Convolutional Models than LSTM models in order to reduce the dependency which then helped reduce the graph traversal.

- Basic Semantic Unit Based: AMR graphs focus only on graph-to-graph transformation and text generation module is not developed for it. To deal with it, Li (2015) have used semantic information to create multi-document abstractive summaries by using BSU(Basic Semantic Unit), the most basic element of coherent text. They have constructed BSU semantic link network where BSU indicates actors and receivers and consists of semantic links, semantic nodes, and reasoning rules. Semantic links help captures the context around the nodes. Text can be generated efficiently using this methods.

2.2.4. Multimodal semantic method

Most of the information in the web is not purely textual. It mostly contains images, videos, etc. along with the text. Kasture1 et al. (2014) has divided the general framework for multimodal summarization into three phases namely semantic model construction of concepts, a rating of concepts on the basis of factors like completeness, and sentence generation. Greenbacker (2011) used the information density as a metric to calculate the content obtained from multimodal summary methods. To create the abstractive multimodal summaries, they first created the semantic model using knowledge representation, where they constructed the concepts and then extracted the semantic information from the limited specific class of images. After extracting the semantic information, they identified the important concepts using information density metric. They used the phrasing methods to generate the summaries. UzZaman, Bigham, and Allen (2011) created multimodal summaries of complex sentences by finding the entities and the main idea behind the text and then added the structure to the images. Ontology is used in this approach to find the concepts but no automatically generated ontology is used in these approaches which leads to wastage of times. Also, there is no automatic method to evaluate these summaries, which limits the efficiency of these methods.

Table 4 lists down the advantages and disadvantages of semantic-based approaches on page 16.

2.3. Deep Learning and neural network based approach

Deep Learning is a part of machine learning based methods and involves training and learning data. It involves multiple layers of non-linear processing to extract the features from the text. Learning can be both supervised or unsupervised. They are based on artificial neural networks. Deep Learning has been successfully applied to various NLP tasks. Buys and Blunsom (2017) showed that parsers based on Recurrent Neural Networks(RNN) have achieved state of art performance in dependency and constitutional parsing. RNN models help predict complex relations which simple structured or semantic-based approaches cannot do alone.

2.3.1. Introduction to encoder-decoder model

Encoder and decoder are the individual neural networks, and work together as a combined neural network. Encoder's task is to understand the input sequences while the decoder's task is to determine the sequences and give the output. Encoder converts the words into the vectors representation which helps capture the context. Mostly word embedding is used for representing the words in encoders but many have used bag-of-words model (Rush, Chopra, & Weston, 2015) too. Decoders help find the next word in the summary on the basis of previous words. When both the input and output are in the form of sequence like in the case of text summarization, the learning problems are also called as Seq2Seq learning problems. Encoder-Decoder models help solve these problems. Attention mechanism is mostly used in sequence models where the information is extracted from the encoder on the basis of attention scores and this information is used by decoder. Attention helps know what part of information do we need to focus at a particular time-stamp.

Few common networks used in encoder-decoder models to solve the abstractive summarization problem are:-

- Convolutional Neural Networks(CNN): Here the input size is always fixed. Each input of the network is independent of previous and future inputs.
- Recurrent Neural Networks(RNN): In most of the practical applications, input size is usually not fixed. Also, the inputs are not independent, rather depend on each other. Moreover, the number of predictions required as output is also not fixed. These are also called as sequence learning problems. Recurrent connections are added to the neural networks which helps capture the dependency of inputs with previous or future inputs.
- Long Short-Term Memory(LSTM) Networks: RNN reads the input from left to right and updates the state after every word. But, when we reach the end, the information of first few words is lost. To enable the forgetting, selective read and selective write, LSTMs are used. It includes the gates to support the forget, read and write operations. Gated Recurrent Units(GRU) are also very popularly used in abstractive summarization process. GRU are the variants of LSTM, where instead of different forget and input gates, only one gate is used. Forget gate is not used explicitly.

2.3.2. Recent works on abstractive summarization using Deep Learning Models

Baralis et al. (2013) used neural networks for parsing the semantic graphs for generating the abstractive summaries by deep linguistic analysis. They used Minimal Recursion Semantics(MRS) for semantic representation of grammars. They performed disambiguation by using maximum entropy model. They developed this

model to solve the alignment problem of AMR Graphs. MRS can be used both for parsing and the text generation. Niu et al. (2017) proposed a multi-document abstractive summarization approach by using chunk graphs and neural networks. They used a Recurrent Neural Network Language Model which helped evaluate the linguistic quality of sentence, which further helped create good readable abstractive summaries.

Simple sequence-to-sequence model map the input sequence to the output sequence. Jobson and Gutirrez (2018) used encoder-decoder RNN along with LSTM to create the summaries. They used the word-embedding for the training purpose and attention function for creating the context vector at each time step. Nallapati, Zhou, dos Santos, Gulehre, and Lapata (2016) used the attention model along with the RNN to handle the issues of modeling keywords and capturing of the hierarchical structure between the sentence and word. They used the bidirectional encoder with GRU(Gatted Recurrent Unit, used to solve the vanishing and exploding gradient problems)-RNN and unidirectional decoder with GRU-RNN. They used the attention model on the hidden-states of source and softmax layer on the target. Rush et al. (2015) used the feed-forward neural network to work on sentence-level text summarization. They used the attention-based encoder and beam-search based decoder for sentence-level summarization. Chopra, Auli, and Rush (2016) used the encoder-decoder model along with the conditional RNN to solve the problem similar to Rush et al. (2015). Rossiello, Basile, Semeraro, Ciano, and Grasso (2016) used the neural networks along with RNN and probabilistic models to create the grammatically correct abstractive summaries. They used the prior knowledge along with the neural networks to model the problem. Liy, Lamy, Bingz, and Wangy (2017) used the sequence-to-sequence encoder-decoder model to generate the abstractive summaries. They considered the latent structured information of the text to improve the quality of summaries. They used the recurrent generative decoder to translate the source code into hidden states and then back to original word-sequences to generate the summary.

Song, Huang, and Ruan (2018) proposed a deep learning based approach called Long Short-Term Memory encoder-decoder model where instead of words, they have used phrases as input to generate the abstractive summaries. Fan, Grangler, and Auli (2018) created a personalized controllable abstractive summarization approach using sequence-to-sequence encoder-decoder based Convolutional Neural Networks model. They created the summary according to the user preferences like entity, whose information they want to know, the size of summary, part of text whose summary they want to obtain.

Most of the deep learning based models are applied to the single-document generic systems. Baumel, Eyal, and Elhadad (2018) created the multi-document query-specific abstractive summaries using Relevance Sensitive Attention-based model. Nema, Khapra, Laha, and Ravindran (2018) used the diversity-driven attention model for solving the same problem. Nema et al. (2018) have attempted to solve the problem of repeated phrase generation by sequence-to-sequence model by using diversity-driven attention model, which helped achieve a gain of 28% in ROUGE-L Score.

Even though deep learning has been applied successfully and emerged as one of the promising approaches to create the abstractive summaries. But the availability of a good large corpora for the training purpose is still a challenge. Moreover, most of the corpora are old and thus do not contain the updated morphological, semantic and syntactic features. Not just this, but also most of the corpora are in English only. Modaresi and Conrad (2016) have created an approach to create a single document corpus to address the above-mentioned problems. Lin, Sun, Ma, and Su (2018) used the Seq2Seq model along with the attention mechanism to solve the problem of repetitions with the Seq2Seq models. They used convolutional gated units along with the global encoding at the encoder side and unidirectional LSTM at the decoder side to perform the abstractive summarization.

Table 1
Summary of evaluation results on DUC dataset.

Technique	Range of ROUGE-1 Score
Tree Based	0.3–0.4
Template Based	0.21–0.35 (Oya, 2014)
Ontology Based	0.29–0.319 (Hennig, Umbrath, & Wetzker, 2008b)
Lead and Body Phrase	0.2–0.3(Highest : 0.28) (Song, Huang et al., 2018)
Graph Based	0.31
Semantic Graph Based	0.3–0.4(0.417 as best score) (Khan et al., 2018)
Predicate-Argument Based	0.3–0.4
Discourse-Based	0.2–0.35 (Cohan et al., 2018)
MultiModal	0.05–0.3
Deep Learning	0.28–0.47 (Joshi, Fidalgo, & Alegre, 2018)

2.4. Discourse and rhetorical structure based approach

Discourse Structure helps capture the structure of text and thus help find the most important sentences of the text (Khan et al., 2018). Here the main aim is to capture the internal representation of the text and convert them into the relations of discourse (Hípola et al., 2014). Rhetorical Structure Theory assumes that the documents can be represented in the form of hierarchical trees. Rhetorical structure in the form of tree is created from the discourse. Algorithm is created to assign the weights to these elements. Higher the element is in the rhetorical structure, higher the weight is given. Then on the basis of length desired for the summary, elements are extracted. Discourse connectors help increase the coherence and cohesion of the text. Discourse parser is used to create the parse trees and then discourse tree representation is created for sentences (Gerani et al., 2014). Rhetorical Structure (Chengcheng, 2010) is the way to analyze the text at the clause level and deals at document-level. Rhetorical relation is the relation between the two non-overlapping texts. A tree like structure is created to represent the coherence in the text. Discourse units participate in it with one element called as nucleus and another as satellite (Goyal & Eisenstein, 2016). The main steps are to identify the text phrase, creation of Rhetoric Structure Trees, processing of trees to find the important and unimportant sentences by finding the nucleus elements, and then creation of summary. Advantage of Rhetorical Structures are that they help create complete, grammatically correct and readable summaries.

Gerani et al. (2014) used the discourse based approach to summarize the product reviews. They have used aspects and their structured relations to generate the abstractive summaries by analyzing the discourse structure and the relations. Their approach does not require domain knowledge. Aspect-Based Discourse tree is created for each review, then they are merged to create Aggregated Rhetorical Relation Graph(ARRG). Weighted PageRank is applied to ARRG, to create the final sub-graph called Aspect Hierarchy Tree(AHT) to create the final summary by applying text planning and sentence realization. In Sentence Realization step, they have used templates along with the NLG.

Table 3 lists down the advantages and disadvantages of Deep Learning-based approaches on page 15.

Below in Table 1, we have found the range of ROUGE Scores obtained by the above mentioned methods on the DUC 2001 dataset, used for creating abstractive summaries.

Table 2
Advantages and disadvantages of structured based approach.

	Advantages	Disadvantages
Template Based Methods	Because of the fact that the snippets are filled with the information extracted by Information Extraction systems, provide highly coherent and informative summaries. They can be used for multi-document summarization also.	They lack the diversity as they are mostly pre-defined. They lack when the system needs to consider the similarity and differences between the documents (Khan, 2014).
Ontology Based Methods	Ontology Based Methods can handle the uncertainties associated with the text easily.	Ontology based methods need a good ontology or dictionary which is mostly created by an expert in the domain, thus is very time-consuming approach.
Tree Based Methods	Use of language generator improves the quality of summaries. They produce fluent and less-redundant summaries (Khan, 2014)	Tree based methods do not consider the context; thus miss many important phrases of the text.
Lead and Body Phrase Methods	They are good for semantic revisions of a lead sentence (Rananavare & Reddy, 2017).	Lead and Body Phrase Methods produce redundant, less grammatical and sometimes incomplete summaries. Parsing failure leads to the wrong phrase substitution leading to incorrect summaries.
Rule Based Methods	Summaries are of high information density.	Because of the fact that rules and patterns are mostly hand-crafted, it is very tedious to create these summaries (Kasture1 et al., 2014).

Table 3
Advantages and disadvantages of deep learning and neural networks based approach.

	Advantages	Disadvantages
1.	Neural networks help capture the proper syntactic role of each word.	Large Training is required to capture the good representation of the text.
2.	Neural Networks can help reduce the grammatical errors.	They rely on the statistical co-occurrence of words, which leads to semantic and grammatical errors (Rossiello et al., 2016).
3.	Deep neural networks help capture the intrinsic and high level distributed representation of data.	Deep Learning Models mainly work at sentence-level, which makes them effective for sentence compression but not much to the document summarization.
4.	Deep Learning Models help capture the semantic and syntactic structure of the text together which is not possible with simple extractive or abstractive techniques (Song, Huang et al., 2018).	Most of the time, there is an alignment problem between the original and training data, which leads to problems during abstractive summarization (Jobson & Gutirrez, 2018).
5.	They help achieve better ROUGE-Scores.	Even though, these techniques create short summaries but they lack in terms of information density (Baumel et al., 2018). They suffer from the problem of repetitions when creating multi-sentence summaries (Chen & Bansal, 2018) (Song, Zhao et al., 2018). Also, they suffer from the problem of slow encoding when the document to be summarized is long (Chen & Bansal, 2018).

Table 4
Advantages and disadvantages of semantic based approach.

	Advantages	Disadvantages
AMR Graphs	Helps produce coherent, less redundant, and grammatically correct sentences.	1. AMR Graphs are very much influenced by syntax and thus does not support discovery and manipulation of concepts. 2. Different graphs are generated for same sentence when they are changed to active or passive voice. 3. AMR Graphs also suffer from the data sparsity problem. 4. AMR Graphs highly rely on PropBank for relations and thus adds constraints and limitations of PropBank to it. 5. Lack of aligned annotations create challenges while parsing AMR Graphs.
INIT Based Methods	INIT Based methods produce coherent, non-redundant and informative summaries.	A Single INIT does not represent the complete sentence. Thus many INITs need to be fused together to create a complete sentence (Genest & Lapalme, 2011). INIT Based methods discard a lot of important information while attempting to create grammatically correct and meaningful summaries. Incorrect parses also lead to the generation of low linguistic quality summaries (Khan et al., 2018)
Multimodal Based Methods	Multimodal based methods produce excellent coverage summaries.	Manual evaluation is required as the document contains both images and text.
Rich Semantic Graph Based Methods	They produce less redundant and grammatically correct summaries.	Rich Semantic Graphs(RSG) are mostly limited to single-document summarization.

3. Generating the abstractive summaries

In this section, we have focused on those steps which are the important part of generating the text. From our survey, we found that sentence compression, concept fusion, calculation of path scores and summary generation are few common parts of an abstractive summarization system. In this section, we have focused on the techniques used by various researchers to perform these particular tasks.

3.1. Sentence compression

Sentence Compression is to reduce the length of sentence while preserving the original meaning of sentence (Napoles, Callison-Burch, Ganitkevitch, & Durme, 2011). Many times, irrelevant details are also present in the sentence and it is important to remove them to avoid the space problem. It is one of the very important topic of NLP community and plays a very important role in creating abstractive summaries. On the basis of the type of structure used for the sentence compression, these methods are divided into tree-based, discourse-based and sentence-based methods. Tree-based approaches (Galley & Mckeown, 2007; Knight & Marcu, 2000) create the compound sentences by editing the syntactic trees obtained by parsing the text. But they are highly dependent upon the parser, which limits them. Whereas, sentence-based approaches (McDonald, 2006) directly create the sentences but they lack in terms of lexicalization. Discourse-based approaches (Clarke & Lapata, 2008) consider the discourse information of surrounding text to compress the sentences.

On the basis of type of learning used for compression, they are also divided into supervised and unsupervised models. Supervised approaches involve the training to find the phrases for compression. To find the probabilities, they use either the generative or the discriminative model. In generative models, either the probability of target compression is found directly or indirectly by using noisy channel. Whereas, in discriminative models, main aim is to reduce the training errors. Mostly SVM, maximum entropy, decision trees and large margin learning are used for discriminative modelling (Clarke & Lapata, 2008). Unsupervised approaches use rules and language models to compress the sentence.

On the basis of whether the words are deleted or the new phrases or words are used, these approaches are divided into delete-based and generate-based (Yu, Zhang, Huang, & Zhu, 2018). In delete-based approach, unimportant words are deleted from the sentence and by stitching of rest of the text, the final sentence is created. In generate-based models, insertion, substitution and replacement is used. Generate-based models require the deeper understanding of the text.

Knight and Marcu (2000) have described the probabilistic approach to compress the sentences. They have used the generative noisy channel and decision tree based methods to compress the sentence. Galley and Mckeown (2007) also used the syntactic trees to compress the sentence by using synchronous context free grammars. McDonald (2006) used the machine learning, discriminative online learning approach(Margin Infused Relaxed Algorithm, MIRA) to learn the feature weights and find the text which need to be considered for compression. To create the feature vectors and the final sentence, they used the deep syntactic analysis of sentence by parsing their sentences with both the dependency parser and phrase-structure parser.

Nguyen and Ho (2004) used Hidden Markov models to compress the sentence. CLASSY(Clustering, Linguistics and Statistics for summarization system) (Conroy, Schlesinger, & Stewart, 2005), compress the sentences using syntactic and lexical heuristics like elimination of phrases, named entity identification, Hidden Markov Models. Integer Linear Programming(ILP) is also one of the famous approach for sentence compression. It not only helps capture the statistical information but also helps capture the discourse level information. Most of these approaches use the local information to compress the sentence. To solve this issue, Clarke and Lapata (2008) used the unsupervised learning along with the constrained optimization, called Integer Linear Programming(ILP) to compress the sentence. They used discourse information along with the simple sentence compression. Sahooa, Bhoib, and Balabantaray (2018) used Support Vector Machine(SVM) to compress the sentences by first parsing the sentences from Stanford Parser and then used these parsed sentences for training purpose.

Zajic, Lin, Dorr, and Schwartz (2006) developed an approach called TRIMMER to compress the sentences for creating multi-document summaries. They trimmed the syntactic constituents of the sentence till they reach to the threshold point and then selected the sentences with best topic coverage. But their applicability is limited to English only and lacks in terms of finding the appropriate sentences.

Mani et al. (2010) conducted an experiment to find how the machine learning algorithms work to decide the predicates to be added to the sentence fragments to combine them together to create a non-extractive sentence; and found that SVM works reasonably well in comparison to other classifiers. Napoles et al. (2011) extracted the set of paraphrases that minimizes the length of the sentence. Zhang et al. (2016) used Integer Linear Programming to generate the sentences for the summary.

Machine learning models alone have not achieved the state of art but when used with the models like Recurrent Neural Networks (RNN), have given very good results. Most of the compression techniques involve the processing of syntactic information, Filippova, Alfonseca, Colmenares, Kaiser, and Vinyals (2015) have proposed an approach which uses RNN based Long Term Short Memory Models to compress the sentence. They trained the model by using a corpus of approximately 2 million sentences. They first parsed the sentence, then created the embedding vectors of 518 dimensions size for the sentence by using dependency tree obtained by parsing, and then decoded the sequence by using beam search procedure. Their system achieved the readability score of 4.5 out of 5 and the informative score of 3.8 out of 5. Yu et al. (2018) used both the delete-based and generate-based models along with the Seq2Seq model(bidirectional RNN and Gated Recurrent Units, GRU) to compress the sentence.

3.2. Concept generalization and fusion

Here the different concepts appearing in the text are replaced with one concept which is the generalization of all the concepts. It helps in reducing the text. Belkebir and Guessoum (2016) performed the concept generalization by using WordNet corpora. They performed this by finding the generalizable sentences by generating the hyperonymy paths of concepts in a sentence and then reducing the size of generalizable versions. Sentence Fusion is also used to create the summaries. In sentence fusion, the related sentences are given to the system as an input, dependency structures of the sentences are aligned to find the common information and then on the basis of alignment, fusion tree is created. Alignments can be created at word-level, phrase-level, substring-level, etc but for the fusion purpose, alignment at dependency-tree level is considered best. Alignment helps understand how the words are related to each other. The best paths are obtained from the trees and are fed to the Natural Language Generation system to create the final sentence (Marsi & Krahmer, 2005).

Barzilay and McKeown (2005) used the dependency trees for the fusion of sentences to create the fluent, grammatically correct and non-redundant summary by utilizing the property of common information among the sentences. They created the fu-

sion lattices and linearized them by using natural language generator FUF/SURGE to create the final sentences. Filippova and Strube (2008) created the multi-document abstractive summaries for German biographies by using the concept of sentence fusion. They used informative phrase merging and pruning along with the dependency structure alignment to create the summaries. Mehdad et al. (2013) built an entailment graph for the sentences to find out the most relevant sentences, and then used the word graphs along with the generalization and aggregation to combine the sentences to form the informative summaries. Belkebir and Guessoum (2016) used concept fusion to create the text-to-text generation technique for creating the abstractive summaries. But they alone cannot create a good summary due to the fact that identifying the common fragments is a big challenge and then using the fusion lattice to combine the sentences to form the grammatically correct sentence is again a complex problem. Also the approach has the problem of referring-expression. Konstas et al. (2017) used the Integer Linear Programming(ILP) to find and merge the informative phrases to obtain the global optimal solution.

3.2.1. Paraphrasing

Replacing the phrases with shorter paraphrases help reduce the length of original text (Barzilay & Lee, 2003b). Paraphrase detection and finding entailment between the phrases and the sentences is not an easy task. Cohn and Lapata (2013) used the sentence-level abstracts to create the summary by creating an end to end paraphrasing system to not only acquire the paraphrases but also use them to create the new strings. They used Synchronous Tree Substitution Grammar to create the compressed parse tree, and then trained them discriminatively. Issa, Damonte, Cohen, Yan, and Chang (2018) used AMR parsing along with the latent semantic analysis to detect the paraphrases. Through latent semantic analysis, they tried to find semantic similarity through distributional representation. But the accuracy of AMR Parser due to the formalism and annotation problem limited the paraphrase-detection system also. Chen and Bansal (2018) used the paraphrasing by first selecting the salient sentences and then rewriting to create the abstractive summary. They operated at both the sentence-level and word-level.

3.3. Calculation of path scores

Semantic graphs which includes Rich Semantic Graph, Dense Semantic Graph, AMR, etc and are one of the very popular way to create the abstractive summaries. The semantic graphs can be obtained by two ways. One way is by representing the sentence in subject-object-verb and other way is by finding the dependency relations between the sentences. To create the graph from later way, shortest dependency path score need to calculated (Joshi, Wang et al., 2018). Bhargava, Sharma, and Sharma (2016) calculated the scores on the basis of redundancy of overlapping sentences and length of path by calculating the intersection of position of words in the sentence. Mehdad et al. (2013) used language model, information about nouns and verbs present in the sentences to find the paths which are more readable and fluent.

3.4. Summary generation

Mainly NLG tools are used for summary generation as they increase the fluency and decrease the grammatical mistakes. NLG has been used to add new vocabulary and language structures to the sentences (Lloret et al., 2013). Fig. 2 represents the modules of the Natural language generation process. It basically involves:-

- Text planning: This step is also called content determination. It is to decide which all information to be included to the summary. It is the preliminary step in most of the systems for creating summaries. It can also be divided into content selection and text structuring.
- Sentence planning: It is to organize the content in a manner to produce the sentence specification. Sentence boundaries are specified in this phase. Here the aim is to arrange the text in sub-paragraphs. Relationship between the text is considered and on the basis of relatedness, the sentences are merged to create intermediate paragraphs. This step is also known as Sentence Aggregation. It is to provide structure to the summary. Machine Learning helps learn good summary structure (Genest & Lapalme, 2011). This phase is again divided into four phases namely lexical analysis, discourse analysis, aggregation and referring expression.
  - Lexical analysis: Here the individual lexical units are chosen based on various factors like fluency, variability, language and formal language style. Here the right word and phrase is found to be substituted to the text to express the information (Dohare & Karnick, 2018). Synonyms for nouns and verbs are obtained to generate the target words. Here the domain words are classified into lexical items. It involves finding the semantically similar words, synonyms or other taxonomically related words.
  - Discourse analysis: Here the individual sentences are generated from the synonyms and phrases which are found during the lexical analysis phase.
  - Aggregation process: It is to decide how the sentences will be merged to form the intermediate paragraphs. Many times the hand crafted rules are used to find the way to merge the sentences.
  - Referring expression process: Here the subject is replaced with the pronouns. Here the appropriate words are chosen which enables distinguishing the entities.
- Realization: Here the objective is to convert the intermediate paragraphs so obtained from sentence planning phase to the final paragraphs. Here the paragraphs are corrected considering syntax coherence, grammatical and punctuation correctness. It involves morphological and syntactic transformations.
- Evaluation: Here the paragraphs are ranked and sorted according to various factors like coherence between the sentences, most frequently used word synonyms, etc.

To summarize the works done in this field of research, we have listed them in the tabular form in Table 5 in page 26 which lists down the author name, Evaluation measure used, dataset used, and type of abstractive summary created.

4. Tools used and developed

From our survey, we have found that most of the abstractive summarization systems consist of 3 steps namely pre-processing, inferencing and Natural Language Generation. Pre-processing of the text involves creating the representation for the text and it includes identifying the named-entities, coreference resolution, finding part of speech tagging, construction of dependency trees, construction of semantic trees, etc. Inferencing the text includes learning the representation of text obtained from pre-processing step. This step mainly includes fusion, deletion, applying some learning model like neural networks, etc. And the final step includes Natural Language Generation where the final grammatically correct summary is generated for the text. On the basis of the steps involved in creating abstractive summaries, we have divided the tools used or created into 3 categories namely:- pre-processing tools, summa-

Table 5
Summary of works.

Author	Method	Dataset	Evaluation Measure	Single/Multi-Document
(Belkebir & Guessoum, 2016)	Concept Fusion and Generalization			Single-Document
(Cohn & Lapata, 2013)	Sentence Compression Approach	Newspaper articles from British National Corpus	Human Based Evaluation(Spearman's coefficient for distribution of ratings)	Single-Document
(Bhargava et al., 2016)	Structure Based(Graph Based + Rule Based)	DUC 2002	ROUGE-N	Single-Document
(Kurisinkel et al., 2017)	Tree Based	DUC 2004, DUC 2007, TAC 2011	ROUGE Metric, Human Based Evaluation	Multi-Document
(Yousfi-Monod & Prince, 2008)	Tree-Based Sentence Compression	Opinions dataset	Compression Rate, Human Evaluation	Single-Document
(Baralis et al., 2013)	Yago-Ontology Based	DUC 2004	Maximal Marginal Relevance(MMR), Human Evaluation	Single-Document
(Barzilay & McKeown, 2005)	Tree Based	DUC 2002	Human Based Evaluation,(Compression Ratio, Grammatical Correctness)	Multi-Document
(Oya et al., 2014)	Template Based	AMI Corpus	ROUGE-1, ROUGE-2, ROUGE-SU4	Single-Document
(Gerani et al., 2014)	Discourse Structure + Template Based	Reviews from Amazon.com and Cnet.com	Human Based Evaluation	Multi-Document
(Banerjee et al., 2017)	Graph Based + ILP	DUC 2004 and DUC 2005	ROUGE-2, ROUGE-L, ROUGE-SU4 and Human Evaluation	Multi-Document
(Genest & Lapalme, 2011)	INIT Based		Pyramid Score, Linguistic quality and overall responsiveness score	Multi-Document
(Sahooa et al., 2018)	Rule Based	DUC 2002	ROUGE-1, ROUGE-2, Avg Precision, Avg Recall and Avg F-Score	Single-Document
(Khan et al., 2018)	Semantic Graph Based Summarization	DUC 2002	ROUGE-1, ROUGE-2 and Pyramid Scores	Multi-Document
(Bartakke et al., 2016)	Semantic Graph Based Summarization		Precision and Recall	Multi-Document
(Moawad & Aref, 2012)	Rich Semantic Graph Based Summarization	Graduate Student text	compression rate	Single-Document
(Liu et al., 2015)	AMR Graph Based Summarization	AMR Bank	ROUGE-1	Single-Document
(Le & Le, 2013)	Syntactic and Discourse Rules Based Summarization	Vietanamese NewsPaperss	ROUGE-1 and ROUGE-2	Single-Document
(Ganesan et al., 2010)	Graph Based Abstractive Summarization	Opiniosis Dataset	ROUGE-1, ROUGE-2, ROUGE-SU4	Single-Document
(Zhang et al., 2016)	Cross-Language Predicate Argument Structure Based Abstractive Summarization	DUC 2001	ROUGE-1, ROUGE-2, ROUGE-SU4	Multi-Document
(Vilca & Cabezudo, 2017)	Semantic Analysis and Discourse Based	DUC 2002	F-1 Measure, ROUGE-1	Single- Document
(Alshaina et al., 2017)	Abstractive Summarization using Predicate-Argument Structure	DUC 2002	Pyramid Score	Multi-Document
(Song, Huang et al., 2018)	Abstractive Summarization using Deep Learning	CNN and DailyMail	ROUGE-N	Single-Document
(Niu et al., 2017)	Graph + Neural Networks Based Abstractive Summarization	DUC2004	ROUGE-2, ROUGE-SU4	Multi- Document
(Nallapati et al., 2016)	Attention Encoder-Decoder Recurrent Neural Networks	DUC2003, DUC2004, CNN Mail Corpus	ROUGE-1, ROUGE-2, ROUGE-L	Single-Document
(Rush et al., 2015)	Attention Model Based Deep Learning	DUC 2003, Gigawords, DUC 2004	ROUGE-1, ROUGE-2, ROUGE-L	Single-Document
(Chopra et al., 2016)	Convolutional RNN Based Deep Learning	Gigawords, DUC 2004	ROUGE-1, ROUGE-2, ROUGE-L	Single-Document
(Liy et al., 2017)	Deep Learning Based Sequence-to-sequence Model	Gigawords, DUC 2004, LCSTS	F-Measures of ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-SU4	Single-Document
(Baumel et al., 2018)	Deep Learning Based on Relevance Sensitive Attention Model	Debatepedia, DUC 2005,2006 and 2007	ROUGE-1, ROUGE-2, ROUGE-L	Multi-Document Query-Specific
(Nema et al., 2018)	Deep Learning Based on Diversity Driven Attention Model	Debatepedia based dataset	ROUGE-1, ROUGE-2 and ROUGE-L	Multi-Document Query Specific
(Fan et al., 2018)	Encoder-Decoder Convolutional Neural Network Model	CNN-Dailymail	ROUGE-1, ROUGE-2 and ROUGE-L	Personalized Controllable Single-Document

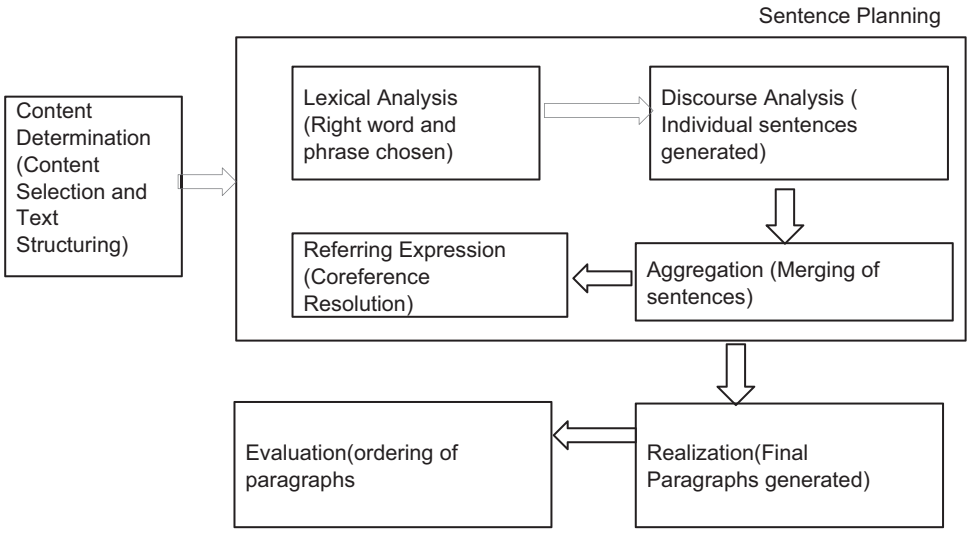

Fig. 2. Steps involved in creating summaries using NLG.

rization techniques and natural language generation tools. Below is the list of tools along with their functionalities.

1. Pre-processing tools: These are the tools which are used for performing pre-processing during the summarization process and their aim is to mainly find the named-entities, perform sentence segmentation, semantic role labeling, create the dependency trees, help in co-reference resolution process,find synonyms, etc.
   - WordNet[10]: It is a very popular english lexical database where the nouns, verbs, adjectives and adverbs are grouped and organized into synsets. Synsets are related to each other by synonyms, hypernyms, hyponoms, meronyms, holonyms, etc. and helps find the semantic relations between the words. Thus, wordnet help in text classification and find the concepts in the text which then helps in text summarization process.
   - FrameNet[11]: It is a human and machine readable english lexical database with annotating examples. It helps assign the semantic roles, and helps find the relationships between the predicates.
   - SParser[12]: It is a phrase structure-based chart parser. It has an extensive and extendible semantic grammar. Greenbacker (2011) used the SParser to generate the multimodal abstractive summaries. It has been used in Greenbacker, McCoy, Carberry, and McDonald (2011) to perform the linguistic analysis by identifying each part of the text in terms of subject and obtained the concepts in terms of partially-saturated referents.
   - MINIPAR Parser[13]: Genest and Lapalme (2011) used this parser to find the subject-verb-object triplets. It takes a sentence as input, parses the sentence and finds the dependency relations among the words in the sentence.
   - GATE information Extraction Engine[14]: GATE is a component-based and open source NLP tool. It helps in POS tagging and semantic tagging. They help identify the lexicons, entities and ontologies. They have a finite state transduction capabilities which make them suitable for summarization as help in pattern matching and identifying annotations. It is also possible to use rules with them for effective summarization. Genest and Lapalme (2011) used it for syntactic analysis.
   - PropBank[15]: It is a corpus of text annotated with information about semantic propositions. Predicate-argument relations are also available to syntactic Penn TreeBank. Vilca and Cabezudo (2017) used it for semantic role modelling which helps identify questions like “Why”, “Whom”, “Where”, “How”, etc.
   - Stanford NLP[16]: It helps perform many NLP based applications like sentence splitting, part of speech tagging, named entity recognition, constituency parsing, dependency parsing, open information extraction, etc. Thus, helps perform text summarization. In most of the papers, to create the dependency trees, stanford NLP parser has been used. Stanford CoreNLP helps perform co-reference resolution.
   - Lingsoft[17]: It is a morphological and syntactic analyzer tool. They provide base and grammatical forms of words, which can be used for labeling and organizing the output. Moawad and Aref (2012) used Lingsoft to syntactically analyze the sentence.
   - SENNA Role Labeller: Khan et al. (2018) used this parser to determine the predicate argument structure for the sentence from the text. Along with the role modeling, it also helps perform part of speech tagging, named entity recognition, chunking and syntactic parsing. It creates frames for each verb in the sentence.
   - LibSVM[18] is a software for support vector classification. It helps perform multi-class classification, weighted Support Vector Machines for unbalanced data. Support Vector Machines(SVM) helps find important sentences by performing text categorization, chunking and dependency analysis.
   - Illinios Chunker[19] is a software tool which partitions the text into semantically related words. It has been used (Oya et al., 2014) to identify all the noun phrases along with

[10] https://wordnet.princeton.edu/.
[11] https://framenet.icsi.berkeley.edu/fndrupal/about.
[12] https://github.com/charlieg/Sparser.
[13] https://gate.ac.uk/releases/gate-7.0-build4195-ALL/doc/tao/splitch17.html.
[14] https://gate.ac.uk/sale/tao/splitch1.html.
[15] https://propbank.github.io/.
[16] https://stanfordnlp.github.io/CoreNLP/simple.html.
[17] http://www.lingsoft.fi/en/499.
[18] https://www.csie.ntu.edu.tw/~cjlin/libsvm/.
[19] https://cogcomp.org/page/software_view/Chunker.

the part of speech tagging in the sentence to create template based summaries.

- VerbNet[20]: It is a hierarchial, domain-independent, online english verb lexicon. It maps verbs to other lexicons like WordNet. Roles are also associated with the verbs. It provides syntactic description of frames and semantic predicates for the verbs which helps in restricting the thematic roles. It has been used widely for the creation of semantic graphs, calculation of sentence similarity and token-level disambiguation for text summarization.
- PractNLP[21]: They are python libraries which are build over SENNA and stanford dependency extractor. It has been used to extract the predicate-argument structure from the text (Alshaina et al., 2017).
- NLTK ToolKit[22]: NLTK helps remove the stop words, fragment the sentence, find the frequency of each word. Katja (2010) used the NLTK Toolkit to label the chunks in the sentences.
- Stanford Core-NLP Toolkit[23]: It has been widely used to perform co-reference resolution (Katja, 2010).
- JAMR Parser: They are used for AMR Parsing. Along with the prediction of graph for each sentence, it also provides the alignment between the span of words and fragments of predicted graph, which then helps during text generation phase (Liu et al., 2015).

2. Summarization tools
   - SIGHT (Demir et al., 2010): It is a tool to generate textual summaries of information graphics. It conveys the underlying message along with the highlights by using visual features. It's visual extraction module has been used (Greenbacker et al., 2011) to analyze the image and intention recognition module has been used to identify the communicative signals present in the graphics. It is mostly used for multimodal document summarization.
   - SUMMONS (Dohare & Karnick, 2018): Tool which creates template based summaries. It is one of the first multi-document summarization system. It has two major components mainly content planner and linguistic generator. Content planner determines the important information to be included to the summary by combining the input templates whereas linguistic generator selects the right words to be used to create a coherent and grammatically correct summary. But the hard-coded templates and domain specificity limits its usage (Bhartiya & Singh, 2018).
   - SUMMARIST[24]: It is a hybrid of Topic Identification, Interpretation and Natural Language Generation. The system uses dictionary and a thesaurus SENSUS to identify the topics and generalize them. Topics fusion is performed for interpretation and then used phrase template generator to create the final abstract summaries.
   - COMPENDIUM (Lloret et al., 2013) uses both the extractive and abstractive techniques to create the final summary. The system first identifies the relevant sentences on the basis of surface linguistic analysis, redundancy detection by using Textual entailment tool, topic identification by using TF-IDF and relevance detection by code quantity principle; then the information compression and fusion is performed to generate the abstractive summaries. They obtained the word graphs for the sentences, filtered the incorrect paths in the word graphs and then created the summaries.
   - MultiGEN:[25] is a tool for performing multi-document summarization based on the similarities and dissimilarities between the documents. They work on the central idea of theme extraction to identify the similar sentences and reformulation on the basis of common themes to generate the summary.
   - System SEA (Carenini, Ng, & Pauls, 2006): Summarizer of Evaluative Arguments, is used for generating the evaluative arguments. It creates the abstractive summaries by calculating the aggregation of extracted information and then applying natural language generation to it.
   - NAMAS[26]: It is a neural abstractive based text summarization system developed by Facebook, which uses Gigaword Corpus. It trains the system and then evaluates by using ROUGE score.
   - GISTEXTER (Harabagiu et al., 2001): Tool which creates both single document and multi-document abstractive summaries. It requires domain knowledge. It creates Template based summaries by extracting information using topic model, CICERO. Because of manual extraction and linguistic rules, require more time and effort to generate summaries. It also uses cue-phrases to achieve the cohesion in the summary.
   - SemanticSumm: It is a semantic[27] approach based summarization system which uses AMR graphs for creating summaries.
3. Natural Language Generation Tools
   - SimpleNLG[28] to generate the sentences (Genest & Lapalme, 2011).
   - FUF or SURGE Language Generator[29]: It is used to generate the sentences by fusing and merging the phrases. FUF is a natural language generation tool which is based on unification of grammars. SURGE is a comprehensive grammar set for FUF.

5. Challenges and future direction

Complexity of natural language processing poses a lot of challenges to the abstractive text summarization. There are lot of open research problems in this field which are yet to be solved. Few of the challenges in this field are listed down as below:

- **Need of quantitative measures**:
  On the basis of whether the evaluation is performed by comparing the results obtained by using some automatic method executed by the system or by comparing the results obtained by people assessment, the evaluation techniques are divided into: quantitative and qualitative. And, On the basis of whether the quality of summary is assessed by itself or the impact of summary on other tasks like readability is calculated to determine its effectiveness, are divided into intrinsic and extrinsic evaluation.
  In quantitative evaluation, the informativeness of sentence is analyzed on the basis of content available. ROUGE is the famous metric employed for quantitative evaluation. While in qualitative analysis, user satisfaction is evaluated against the generated summaries. Some researchers (Banerjee et al., 2017) have used human evaluation along with the ROUGE Score to find the overall quality of summary.

[20] https://verbs.colorado.edu/verbnet/.
[21] https://pntl.readthedocs.io/en/latest/_modules/pntl/tools.html.
[22] https://www.nltk.org/.
[23] https://stanfordnlp.github.io/CoreNLP/.
[24] https://www.isi.edu/natural-language/projects/SUMMARIST.html.
[25] http://www.cs.columbia.edu/diglib/sumDemo/multiGen/main.html.
[26] https://github.com/facebookarchive/NAMAS.
[27] https://github.com/summarization/semantic_summ.
[28] https://github.com/simplenlg/simplenlg.
[29] https://www.cs.cmu.edu/Groups/AI/areas/nlp/nlg/fuf/0.html.

ROUGE is a metric used for performing quantitative analysis and is recall-based. It calculates the number of overlapping n-grams between the system generated summary and the human written summary. There are many variants of ROUGE metric like ROUGE-2, where word sequences of size 2 are considered for comparison; ROUGE-L, where longest common subsequence is considered for comparison; ROUGE-SU4, where skip bigrams are compared with unigrams. Pyramid score has also been used for evaluation purpose as it can evaluate the quality beyond word-level matching (Li, 2015).

Banerjee et al. (2017) used the informativeness and linguistic quality as 2 measures for human based evaluation. Oya et al. (2014) used human based, 5-scale method based on fluency, too many, grammatic mistakes, informativeness and coverage to find the overall quality of summary. Cohn and Lapata (2013) used the human evaluation approach along with the spearman's coefficient to compare the results of their abstractive sentence compression approach to ensure the appropriateness of distribution of ratings they used . Baralis et al. (2013) used human evaluation on the basis of readability to find the effectiveness of their system.

In most of the works, the assessment of summaries has been done by using ROUGE Scores. ROUGE scores help measure coverage but they do not help find coherence and other factors like non-redundancy as they only calculate the repeatability of N-grams (Song, Huang et al., 2018). It has also been observed that same ROUGE-Score has been obtained for different summaries of the same text when their contents are completely different and when the overlapping is also almost different (Mehta, 2016). Even the ROUGE scores fail to give insights to help specify the strengths and weaknesses of the summary (Carenini & Cheung, 2008). More work on finding the variants of ROUGE Score is the need of day (Yao et al., 2017). Baumel et al. (2018) and Zhang et al. (2016) have addressed the issue of designing quantitative measures which can find the quality of abstractive summaries. ROUGE Score works well for extractive summaries but it is not a good metric for evaluating abstractive summaries as they need a metric which can find the semantic overlap than the word overlap.

Pyramid Score is one of the metric used to evaluate the summary from the semantic perspective. It is an annotation based scoring method where the summarization content units (SCU) are obtained from the model summaries and then the weight is assigned to each SCU on the basis of frequency of its occurrence in the human-reference summaries. But, lot of human-intervention is required to perform this evaluation to assess the semantic content of text. AutoPyramid is an extension of Pyramid approach where Passonneau, Chen, Guo, and Perin (2013) tried to reduce this manual effort by automating the process of finding whether the SCU is present in the system-generated summary or not. They used dynamic programming and latent vector method to solve this issue. But they could not quantify the quality of summary on the basis of agreement and contradiction with the human-reference summary. Yang, Passonneau, and de Melo (2016) created one more approach called PEAK(Pyramid Evaluation via Automated Knowledge Extraction) to automatically assess the SCU by using information extraction and graph algorithms. But this approach also lacked many things like it doesn't take into consideration, the co-reference resolution, paraphrase identification, and failed to model the contradiction.

Thus, to resolve the above-mentioned issues of Pyramid, ROUGE and Auto-pyramid methods to evaluate the abstractive summaries; Vadapalli, Kurisinkel, Gupta, and Varma (2017) created one metric, Semantic Similarity for Abstractive Summarization(SSAS) for evaluating the abstractive summaries at semantic inference level. They used deep semantic analysis to find the lexical and semantic measures to compute the similarity between the system-generated and human-generated summaries. But at present, this approach takes too much time compared to ROUGE, Pyramid, Auto-pyramid and PEAK methods. Thus more work on paralleling and finding better ways to obtain the feature vectors to make SSAS more effective is required.

- **Issue of rare words learning in neural networks**: Song, Zhao, and Liu (2018) mentioned how the neural networks or sequence-to-sequence learning fails to preserve the meaning of summary. Rare words sometimes cause the problems to seq-2-seq models because of the fact that rare words usually occur very less in the training system and thus the system does not learn their patterns. Also, because in deep learning approaches, if used individually, the syntactic structure is not fed explicitly, thus they lack the syntactic structure and are bias. The authors have also shown some examples, where the Seq2Seq model fails to capture the main verb of the sentence. To address this issue, more efforts need to be explored which combines the syntactic structure of the text with the neural networks.
- **Availability of datasets**: The availability of good and large training corpus is one of the major roadblock to this area; most of the datasets belong to news articles. From the survey, we have observed that most of the abstractive summarization systems have used DUC[30] and TAC[31] dataset. Gigawords dataset and CNN daily mail dataset are also the famous datasets among the deep learning community. More work on creating good datasets for reasearch community is required. Deep Learning Models have proved to be one of the effective models to model the abstractive summarization problem. But the huge amount of data required for training purpose is one of the major issue with using these models. Most of the Deep Learning Models are applied for the single document extractive summarization and there are very few works available on the query-specific abstractive summarization. The reason is the availability of large scale dataset required for performing query-specific summarization. Nema et al. (2018) have created one dataset[32] using Debatepedia[33] for solving this problem. Alignment of data is also one of the issue for extracting source and target sentences (Mehta, 2016). Mostly the data is aligned at document-level and not at sentence level. Fan et al. (2018) have created the controllable abstractive summaries and allowed the users to mention about the portion of text they want to summarize. But to create the summary for the specific portion of text, they faced the challenge of dataset. They had to create their own dataset as there was no readily available dataset which can be used to perform this kind of summary. They aligned the summaries to full documents.
- **Need of good algorithms for concept fusion and generalization**: Sentence fusion is one of the very challenging area of abstractive summarization field. There is not much work done in the field of concept generalization and fusion. Belkebir and Guessoum (2016) in their paper have used noun rule and adjective rule for concept fusion but more work on finding more rules to perform concept fusion is required. Their approach generated the set of generalizable sentences but their algorithm has exponential space complexity and thus more work on finding the algorithms which can reduce the space of generalizable sentences is required. Mehdad et al. (2013) addressed the need of finding good approaches for community detection and sentence fusion for informal text like meeting transcript infor-

[30] https://duc.nist.gov/.
[31] https://tac.nist.gov/.
[32] https://github.com/PrekshaNema25/DiverstiyBasedAttentionMechanism.
[33] http://www.debatepedia.org/en/index.php/.

mation. Community sentences are those sentences of the text which can be merged together to create an abstract sentence.

- **Scalability issues:** Even though there has been a lot of research in this field but the algorithms require lot of data and power to give good results with long documents (Singhal, Vats, & Karnick, 2018). Also, Most of the work in this field are performed on simple and compound sentences. But more work, on finding scalable approaches which consider the complex-compounded sentences too are required (Sahooa et al., 2018). Even when the neural networks are applied to the summarization of long document, it suffers from the problem of slow and inaccurate encoding due to the fact that attention mechanism is mostly applied and it looks at all the encoded words for decoding purpose (Chen & Bansal, 2018).
- **Semantic similarity calculation:** Semantic Similarity Calculation between the sentences is one of the problems of Natural Language Processing field. It plays a very important role in abstractive summarization process as it helps find the concepts and concepts are the heart of the abstractive summarization systems. Mostly Word co-occurrence , lexical database, and search engine results are used to calculate the semantic similarity between the words or the sentences. Word co-occurrence based models calculate this similarity by comparing the query vector and document vector but as they do not consider the order and context of words, they cannot capture the semantic similarity very efficiently. Lexical databases like WordNet are also used to calculate the similarity between phrases or sentence but direct matching of words or phrases with lexical database information and the fact that the meaning of word differs from corpus to corpus, limits them. Even, the search engine based methods do not give very good results due to the fact that words with opposite meanings also occur together with the search engine results. To solve these issues (Pawar & Mago, 2018) created an approach by creating the semantic vectors for the sentences by using WordNet as the lexical source. But more efforts to extend these methods to various domains and analyze how the results differ by using different ontology is also required. Most of the works have used WordNet to find the lexical information in the text to create the graphs, Mehdad et al. (2013) mentioned the need of finding how the graphs change with using different knowledge sources like Yago Ontology or DBPedia.
- **Need of increasing the efficiency of AMR Graphs**: AMR is a rooted, directed and acyclic graph used to represent the semantic information of a sentence of the text. AMR Graphs are built upon the PropBank frameset, thus the frameset limitation, limits the AMR Graphs. When AMR Graphs are used for summarization purpose, individual graphs are merged together by identification of similar concepts (Liu et al., 2015). But, as the size of text increases, the number of AMR Graphs increases, the merging leads to the complexities. Thus more decoding algorithms like Lagrangian relaxation or approximate algorithms should be discovered to make AMR Based Approach more effective. Liu et al. (2015) have also raised the need of performing both the entity and event co-reference resolution to make the Graphs merging more efficient. Also, to identify the edges to be selected as the candidate for the summary, subgraphs are identified, but mostly the subgraph prediction is at the sentence-level. Identifying how the subgraphs can be found at the document-level, can help achieve more-coverage. AMR Graphs construction depends upon the available parsers which limits its efficiency, as the whole concept of summarization using AMR Graphs depend upon the concept identification. And the efficiency of concept identification depends upon the efficiency of parsers. So, with improvement upon the parsing models, AMR Graphs based summarization models can be improved more. At present, AMR-Graphs are used along with the NLG-tools to create the final summary. Future work can include identifying graph-to-graph based summarization system.
- **Need of single platform for specifying the ontologies**: There are a number of external ontologies available for use. There is a need for single platform which can accommodate all the explicit ontologies and thus it will help get the extensive abstractive summarization system for different domains. Xiang, Jiang, Chang, and Sui (2015) have tried to address this issue by creating an ontology matching approach called ERSOM, which finds the semantically related entities between different ontologies.
- **Need of cross-language based abstractive summarization systems**: Cross-language summarization is to produce the summary of a text written in some source language like Sanskrit in some other target language like in English. In this information era, not all the documents are of same language. Different documents are of different languages. Creating the cross-language summary will help the unfamiliar readers know the essence of the document. From the survey, we have found that mostly extractive summarization techniques have been used for this task and not much work has been done in creating abstractive summaries for cross-language systems. Few of the works by Yao, Wan, and Xiao (2015) and Zhang et al. (2016) have used phrase-based compression and predicate-arguments using machine translation process along with Integer Linear Programming respectively for sentence generation of cross-language documents. More work is required in this field as the cross-language imposes extra complexity to the abstractive summarization because of the fact that not only conciseness, informativeness, and coherency is required but the quality of translation is also one important aspect which decides the quality of summary (Zhang et al., 2016).
- **Need of deep learning based query specific abstractive summarization systems:** Query-specific problems are where the summaries highlight points relevant in the context of query and mostly extractive techniques have been applied to generate query-specific summaries but they suffer from the coherence problem and this probability of getting incoherent summary is very high in case of query-specific summarization problem as without knowing context and just connecting the sentences, it is hard to resolve the co-reference problem. Baumel et al. (2018) used the deep learning as an approach to create query-specific abstractive summaries. Nema et al. (2018) raised the issue of repeated phrases by using encoder-decoder based models while attempting to generate query-specific abstractive summaries. Most of the deep learning based abstractive models are applied for single-document generic summarization. There are very few works which are available on query-specific multi-document summarization. More work on identifying the approaches to deal with this summarization is required.

6. Conclusion

Abstractive summarization is an interesting topic of research among the NLP community and helps produce coherent, concise, non-redundant and information rich summaries. The idea of the paper is to present the recent studies and progresses done in this field to help researchers get familiar about the techniques present, challenges existing and pointers for future work in this area. Along with these we have also mentioned the tools which have been used in various researches related to abstractive summarization. Evaluation of summaries is a big challenge in this field. Semantic analysis, and discourse analysis along with the new emerging technologies like neural networks help overcome the difficulties associated with the abstractive summarization.

CRediT authorship contribution statement

Som Gupta: Conceptualization, Methodology, Validation, Visualization, Written both the original and revised version. **S. K Gupta:** Helped Finalize the topic, Decided the methodology, Supervised during the writing process, Validated the contents, Reviewed the contents.

References

Alshaina, S., John, A., & Nath, A. G. (2017). Multi-document abstractive summarization based on predicate argument structure. In *International conference on signal processing, informatics, communication and energy systems (SPICES)* (pp. 32–37). doi:10.1109/SPICES.2017.8091339.

Banerjee, S., Mitra, P., & Sugiyama, K. (2017). Multi-document abstractive summarization using ilp based multi-sentence compression. In *IJCAI5 proceedings of the 24th international conference on artificial intelligence* (pp. 1208–1214).

Baralis, E., Cagliero, L., Jabeena, S., Fiori, A., & Shah, S. (2013). Multi-document summarization based on the yago ontology. *Expert Systems with Applications, 40*(17), 6976–6984. doi:10.1016/j.eswa.2013.06.047.

Bartakke, D., Sawarkar, D. S., & Gulati, A. (2016). A semantic based approach for abstractive multi-document text summarization. *International Journal of Innovative Research in Ccmputer and Communication Engineering, 4*(7), 13620–13628.

Barzilay, R., & Lee, L. (2003a). Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of HLT-NAACL 2003* (pp. 16–23).

Barzilay, R., & Lee, L. (2003b). Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of HLT-NAAC* (pp. 16–23).

Barzilay, R., & McKeown, K. R. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics, 31*(3), 297–327.

Baumel, T., Eyal, M., & Elhadad, M. (2018). *Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models* (pp. 297–327). Cornell University Libraray.

Belkebir, R., & Guessoum, A. (2016). Concept generalization and fusion for abstractive sentence generation. *Expert Systems and Applications, 53*, 43–56. doi:10.1016/j.eswa.2016.01.007.

Bhargava, R., Sharma, Y., & Sharma, G. (2016). Atssi: Abstractive text summarization using sentiment infusion. In *Twelfth international multi-conference on information processing* (pp. 404–411). doi:10.1016/j.procs.2016.06.088.

Bhartiya, D., & Singh, A. (2014). A Semantic Approach to Summarization. *CoRR*. eprint arXiv: 1406.1203

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S. (2009). Dbpedia - a crystallization point for the web of data. *Journal of Web Semantics First Look*, 154–165. doi:10.1016/j.websem.2009.07.002.

Buys, J., & Blunsom, P. (2017). *Robust incremental neural semantic graph parsing*: 40. Cornell University Libraray.

Carenini, G., & Cheung, J. C. K. (2008). Extractive vs. nlg-based abstractive summarization of evaluative text: The effect of corpus controversiality. In *INLG '08 proceedings of the fifth international natural language generation conference: 1* (pp. 33–41).

Carenini, G., Ng, R., & Pauls, A. (2006). Multi-document summarization of evaluative text. In *11th EACL 2006: 61* (pp. 305–312).

Carenini, G., Ng, R. T., & Pauls, A. (2012). Multi-document summarization of evaluative text,. 0 (0). https://doi.org/10.1111/j.1467-8640.2012.00417.x

Chen, Y.-C., & Bansal, M. (2018). Fast abstractive summarization with reinforce-selected sentence rewriting. *ACL*.

Chengcheng, L. (2010). Automatic text summarization based on rhetorical structure theory. In *International conference on computer application and system modeling* (pp. 595–598).

Chopra, S., Auli, M., & Rush, A. M. (2016). Abstractive sentence summarization with attentive recurrent neural networks. In *Conference: Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies*.

Clarke, J., & Lapata, M. (2008). Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research, 31*, 399–421.

Cohan, A., Dernoncourt, F., Kim, D. S., Bui, T., Kim, S., Chang, W., & Goharian, N. (2018). A discourse-aware attention model for abstractive summarization of long documents, arXiv:1804.05685v2[cs.CL]

Cohn, T., & Lapata, M. (2013). Paraphrastic sentence compression with a character-based metric: tightening without deletion. In *Proceedings of the workshop on monolingual text-to-text generation*.

Conroy, J. M., Schlesinger, J. D., & Stewart, J. G. (2005). Classy query-based multi–document summarization. *2005 document understanding conference*.

Sunitha, C., Jaya, A., & Ganesh, D. A. (2016). A study on abstractive summarization techniques in indian languages. In *Fourth international conference on recent trends in computer science and engineering* (pp. 25–31).

Demir, S., Oliver, D., Schwartz, E., Elzer, S., Carberry, S., & McCoy, K. F. (2010). Interactive sight into information graphics. *W4A2010*.

Dineshnath, G., & S. Saraswathi, D. (2017). Text summarization using abstractive methods. *Journal of Network Communications and Emerging Technologies (JNCET), 7*(12), 26–29.

Dineshnath, G., & S. Saraswathi, D. (2018). Comprehensive survey for abstractive text summarization. *International Journal of Innovations and Advancement in Computer Science, 7*(1), 215–219.

Dohare, S., & Karnick, H. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research, 61*, 65–170. arXiv:1703.09902v4.

Embar, V. R., Deshpande, S. R., Vaishnavi, A. K., Jain, V., & Kallimani, J. (2013). sArAmsha - a Kannada Abstractive Summarizer. In *Advances in computing, communications and informatics (ICACCI), 2013 International Conference on* (pp. 540–544). doi:10.1109/ICACCI.2013.6637229.

Fan, A., Grangler, D., & Auli, M. (2018). Controllable abstractive summarization. In *Proceedings of 2nd workshop on neural machine translation and generation* (pp. 45–54).

Filippova, K., Alfonseca, E., Colmenares, C. A., Kaiser, L., & Vinyals, O. (2015). Sentence compression by deletion with lstms. In *Proceedings of the 2015 conference on empirical methods in natural language processing*.

Filippova, K., & Strube, M. (2008). Sentence fusion via dependency graph compression. In *Proceedings of the 2008 conference on empirical methods in natural language processing* (pp. 177–185).

Foland, W., & Martin, J. H. (2017). Abstract meaning representation parsing using lstm recurrent neural networks. In *Proceedings of the 55th annual meeting of the association for computational linguistics* (pp. 463–472). doi:10.18653/v1/P17-1043.

Galley, M., & Mckeown, K. R. (2007). Lexicalized markov grammars for sentence compression. In *Conference of the North American chapter of the association of computational linguistics*.

Ganesan, K., Zhai, C., & Han, J. (2010). Opinosis: A graph-based approach to abstractive summarization of highly redundant opinions. In *23rd international conference on computational linguistics* (pp. 340–348).

Genest, P. E., & Lapalme, G. (2011). Framework for abstractive summarization using text-to-text generation. In *Workshop on monolingual text-to-text generation* (pp. 64–73).

Gerani, S., Mehdad, Y., Carenini, G., Ng, R. T., & Nejat, B. (2014). Abstractive summarization of product reviews using discourse structure. In *2014 conference on empirical methods in natural language processing* (pp. 1602–1613).

Goyal, N., & Eisenstein, J. (2016). A joint model of rhetorical discourse structure and summarization. In *Proceedings of the workshop on structured prediction for NLP* (pp. 595–598). doi:10.18653/v1/W16-5903.

Greenbacker, C. F. (2011). Towards a framework for abstractive summarization of multimodal documents. In *HLT-SS '11 proceedings of the ACL 2011 student session* (pp. 75–80).

Greenbacker, C. F., McCoy, K. F., Carberry, S., & McDonald, D. D. (2011). Semantic modeling of multimodal documents for abstractive summarization. In *HLT-SS '11 proceedings of the ACL 2011 student session* (pp. 75–80).

Han, X., Lv, T., Hu, Z., Wang, X., & Wang, C. (2016). Text summarization using framenet-based semantic graph model. *Scientific programming*.

Harabagiu, S., Moldovan, D., Morarescu, P., Lacatusu, F., Mihalcea, R., Rus, V., & Girju, R. (2001). Gistexter: A system for summarizing text documents,.

Hennig, L., Umbrath, W., & Wetzker, R. (2008a). An ontology-based approach to text summarization. In *2008 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology* (pp. 540–544). doi:10.1109/WIIAT.2008.175.

Hennig, L., Umbrath, W., & Wetzker, R. (2008b). An ontology-based approach to text summarization. In *IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology* (pp. 291–294). doi:10.1109/WIIAT.2008.175.

Hípola, P., Senso, J. A., Leiva-Mederos, A., & Domínguez-Velasco, S. (2014). Ontology-based text summarization. the case of texminer. *The University of British Columbia Library: Emerald Insight, 32*(2), 229–248. doi:10.1108/LHT-01-2014-0005.

Issa, F., Damonte, M., Cohen, S. B., Yan, X., & Chang, Y. (2018). Abstract meaning representation for paraphrase detection. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: Human language technologies: 1* (pp. 442–452).

Jobson, E., & Gutirrez, A. (2018). Abstractive text summarization using attentive sequence-to-sequence rnns. Stanford Reports,.

Joshi, A., Fidalgo, E., & Alegre, E. (2018). Deep learning based text summarization: Approaches, databases and evaluation measures,.

Joshi, M., Wang, H., & McClean, S. (2018). Dense semantic graph and its application in single document summarization,.

Kasture1, N. R., Yargal, N., Singh, N. N., Kulkarni, N., & Mathur, V. (2014). A study of abstractive summarization using semantic representations and discourse level information. *International Journal for Research in Emerging Science and Technology, 1*(6), 53–57.

Katja, C. (2010). Multi-sentence compression: Finding shortest paths in word graphs. In *COLING '10 proceedings of the 23rd international conference on computational linguistics* (pp. 322–330).

Khan, A. (2014). A review on abstractive summarization methods. *Journal of Theoretical and Applied Information Technology, 59*(1), 64–72.

Khan, A., Salim, N., Farman1, H., Khan, M., Jan, B., Ahmad, A., et al. (2018). Abstractive text summarization based on improved semantic graph approach. *International Journal of Parallel Programming*, 1–25. doi:10.1016/j.eswa.2016.01.007.

Knight, K., & Marcu, D. (2000). Statistics-based summarization step one: Sentence compression. *AAAI-00*.

Konstas, I., Iyer, S., Yatskar, M., Choi, Y., & Zettlemoyer, L. (2017). Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the*

55th annual meeting of the association for computational linguistics (pp. 146–157). doi:10.18653/v1/P17-1014.

Kurisinkel, L. J., Zhang, Y., & Varma, V. (2017). Abstractive multi-document summarization by partial tree extraction,recombination and linearization. In *8th international joint conference on natural language processing* (pp. 812–821).

Le, H. T., & Le, T. M. (2013). An approach to abstractive text summarization. In *International conference on soft computing and pattern recognition (SoCPaR)* (pp. 371–376). doi:10.1109/SOCPAR.2013.7054161.

Lee, C.-S., Chen, Y.-J., & Jian, Z.-W. (2003). Ontology-based fuzzy event extraction agent for chinese e-news summarization. *Expert Systems with Applications*, 431–447. doi:10.1016/S0957-4174(03)00062-9.

Li, W. (2015). Abstractive multi-document summarization with semantic information extraction. In *2015 conference on empirical methods in natural language processing* (pp. 17–21).

Liao, K., Lebanoff, L., & Liu, F. (2018). Abstract meaning representation for multi-document summarization,. arXiv:1806.05655v1

Lin, J., Sun, X., Ma, S., & Su, Q. (2018). Global encoding for abstractive summarization. In *Proceedings of the 56th annual meeting of the association for computational linguistics* (pp. 163–169).

Liu, F., Flanigan, J., Thomson, S., Sadeh, N., & Smith, N. A. (2015). Toward abstractive summarization using semantic representations. *HLT-NAACL*.

Liy, P., Lamy, W., Bingz, L., & Wangy, Z. (2017). Deep recurrent generative decoder for abstractive text summarization.

Lloret, E., Teresa, M., Rom-Ferri, & Palomara, M. (2013). Compendium: A text summarization system for generating abstracts of research papers. *Data and Knowledge Engineering, 88*, 164–175. doi:10.1016/j.datak.2013.08.005.

dolfo Lozano-Tello, & Gomez-Perez, A. (2004). Ontometric: A method to choose the appropriate ontology. *Journal of Database Management (JDM), 15*(2), 18.

Mohan, J. M., Sunitha, C., Ganesha, A., & Jaya, A. (2016). A study on ontology based abstractive summarization. In *Fourth international conference on recent trends in computer science and engineering* (pp. 32–37). doi:10.1016/j.procs.2016.05.122.

Mani, I., House, D., Klein, G., Hirschman, L., Firmin, T., & Sundheim, B. (2010). Inserting rhetorical predicates for quasi-abstractive summarization. In *RIAO 2010* (pp. 138–139).

Marsi, E., & Krahmer, E. (2005). Explorations in sentence fusion. In *Proceedings of the association for computational linguistics (ACL)*.

McDonald, R. T. (2006). Discriminative sentence compression with soft syntactic evidence. In *11st conference of the European chapter of the association for computational linguistics* (pp. 297–304).

Mehdad, Y., Carenini, G., Tompa, F. W., & Ng, R. T. (2013). Abstractive meeting summarization with entailment and fusion. In *Proceedings of the 14th European workshop on natural language generation*.

Mehta, P. (2016). From extractive to abstractive summarization: A journey. In *Proceedings of the 54th annual meeting of the association for computational linguistics* (pp. 100–106).

Moawad, I. F., & Aref, M. (2012). Semantic graph reduction approach for abstractive text summarization. In *Seventh international conference on computer engineering and systems* (pp. 132–138). doi:10.1109/ICCES.2012.6408498.

Modaresi, P., & Conrad, S. (2016). Simurg: An extendable multilingual corpus for abstractive single document summarization. In *FIRE'16* (pp. 24–27). doi:10.1145/3015157.3015161.

Moratanch, N., & Chitrakala, S. (2016). A survey on abstractive text summarization. In *2016 international conference on circuit, power and computing technologies* (pp. 762–765). doi:10.1109/ICCPCT.2016.7530193.

Munot, N., & Govilkar, S. S. (2015). Conceptual framework for abstractivve text summarization. *International Journal on Natural Language Computing, 4*(1), 39–50. doi:10.5121/ijnlc.2015.4104.

Nallapati, R., Zhou, B., dos Santos, C., Gulehre, C., & Lapata, M. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. *The SIGNLL conference on computational natural language learning.*.

Napoles, C., Callison-Burch, C., Ganitkevitch, J., & Durme, B. V. (2011). Paraphrastic sentence compression with a character-based metric: tightening without deletion. In *Proceedings of the workshop on monolingual text-to-text generation* (pp. 84–90).

Nayeem, M. T., & Chali, Y. (2017). Paraphrastic fusion for abstractive multi-sentence compression generation. In *CIKM '17 proceedings of the 2017 ACM on conference on information and knowledge management* (pp. 2223–2226). doi:10.1145/3132847.3133106.

Nema, P., Khapra, M. M., Laha, A., & Ravindran, B. (2018). *Diversity driven attention model for query-based abstractive summarization*. Cornell University Libraray.

Nguyen, C. Q., & Phan, T. T. (2009). An ontology-based approach for key phrase extraction. In *Proceedings of ACL-IJCNLP 2009 conference short papers* (pp. 181–184).

Nguyen, L. M., & Ho, T. B. (2004). Example-based sentence reduction using the hidden markov model. *ACM Transactions on Asian Language Information Processing, 1*, 1–12.

Niu, J., Chen, H., Zhao, Q., Sun, L., & Atiquzzaman, M. (2017). Multi-document abstractive summarization using chunk-graph and recurrent neural network. *IEEE icc 2017 SAC symposium big data networking track*.

Oya, T. (2014). Automatic abstractive summarization of meeting conversations,.

Oya, T., Mehdad, Y., Carenini, G., & Ng, R. (2014). A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th international natural language generation conference* (pp. 45–53).

Passonneau, R. J., Chen, E., Guo, W., & Perin, D. (2013). Automated pyramid scoring of summaries using distributional semantics. In *Proceedings of the 51st annual meeting of the association for computational linguistics* (pp. 143–147).

Pawar, A., & Mago, V. (2018). Calculating the similarity between words and sentences using a lexical database and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering*, 1–14. arXiv:1802.05667v2.

Rachabathuni, P. K. (2017). A survey on abstractive summarization techniques. In *Inventive computing and informatics (ICICI), international conference on* (pp. 762–765). doi:10.1109/ICICI.2017.8365239.

Rananavare, L. B., & Reddy, P. V. S. (2017). An overview of text summarization. *International Journal of Computer Applications, 171*(10), 1–17.

Rossiello, G., Basile, P., Semeraro, G., Ciano, M. D., & Grasso, G. (2016). Improving neural abstractive text summarization with prior knowledge. *URANIA-16*.

Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization.

Sahooa, D., Bhoib, A., & Balabantaray, R. C. (2018). Hybrid approach to abstractive summarization. In *International conference on computational intelligence and data science (ICCIDS 2018)* (pp. 1228–1237). doi:10.1016/j.procs.2018.05.038.

Singhal, S., Vats, A., & Karnick, P. H. (2018). Neural abstractive summarization. *Report*. IIT Kanpur.

Song, K., Zhao, L., & Liu, F. (2018). Structure-infused copy mechnaisms for abstractive summarization. In *Proceedings of the 27th international conference on computational linguistics* (pp. 1717–1729).

Song, S., Huang, H., & Ruan, T. (2018). Abstractive text summarization using lstm-cnn based deep learning. *Multimedia Tools and Applications, 1*(6), 53–57. doi:10.1007/s11042-018-5749-3.

Tanaka, H., Kinoshita, A., Kobayakawa, T., Kumano, T., & Kato, N. (2009). Syntax–driven sentence revision for broadcast news summarization. In *Workshop on language generation and summarisation* (pp. 39–47).

Tran, T., Cimiano, P., Rudolph, S., & Studer, R. (2007). Ontology-based interpretation of keywords for semantic search. In *International semantic web conference* (pp. 523–563).

UzZaman, N., Bigham, J. P., & Allen, J. F. (2011). Multimodal summarization of complex sentences. In *IUI'11* (pp. 595–598).

Vadapalli, R., Kurisinkel, L. J., Gupta, M., & Varma, V. (2017). Ssas: Semantic similarity for abstractive summarization. In *Proceedings of the 8th international joint conference on natural language processing* (pp. 198–203).

Viet, L. D., Sinh, V. T., Minh, N. L., & Satoh, K. (2017). Convamr: Abstract meaning representation parsing for legal document. In *SCIDOCA* (pp. 1–7).

Vilca, G. C. V., & Cabezudo, M. A. S. (2017). A study of abstractive summarization using semantic representations and discourse level information. In *Text, speech, and dialogue* (pp. 482–490). Springer.

Wang, L., & Ling, W. (2016). Neural network-based abstract generation for opinions and arguments. In *Proceedings of NAACL-HLT 2016* (pp. 47–57).

Xiang, C., Jiang, T., Chang, B., & Sui, Z. (2015). Ersom: A structural ontology matching approach using automatically learned entity representation. In *Proceedings of 2015 conference on empirical methods in natural language processing* (pp. 2419–2429).

Yang, Q., Passonneau, R. J., & de Melo, G. (2016). Peak: Pyramid evaluation via automated knowledge extraction. In *Proceedings of the thirtieth AAAI conference on artificial intelligence* (pp. 2673–2679).

Yao, J.-G, Wan, X., & Xiao, J. (2015). Phrase-based compressive cross-language summarization. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 118–127).

Yao, J.-G., Wan, X., & Xiao, J. (2017). Recent advances in document summarization. *Knowledge Information Systems*. doi:10.1007/s10115-017-1042-4.

Yousfi-Monod, M., & Prince, V. (2008). Sentence compression as a step in summarization or an alternative path in text shortening. In *Coling 2008* (pp. 139–142).

Yu, N., Zhang, J., Huang, M., & Zhu, X. (2018). An operation network for abstractive sentence compression. In *Proceedings of the 27th international conference on computational linguistics* (pp. 1065–1076).

Zajic, D. M., Lin, J., Dorr, B. J., & Schwartz, R. (2006). Sentence compression as a component of a multi-document summarization system. *2006 document understanding conference*.

Zhang, J., Zhou, Y., & Zong, C. (2016). Abstractive cross-language summarization via translation model enhanced predicate argument structure fusing. *EEE/ACM Transactions on Audio, Speech, and Language Processing*, 1–25. doi:10.1109/TASLP.2016.2586608.

Zhang, X., Cheng, G., & Qu, Y. (2011). Ontology summarization based on rdf sentence graph. In *WWW 2007 / track: Semantic web* (pp. 707–715).