
CONGBO MA, The University of Adelaide
WEI EMMA ZHANG, The University of Adelaide
MINGYU GUO, The University of Adelaide
HU WANG, The University of Adelaide
QUAN Z. SHENG, Macquarie University

Multi-document summarization (MDS) is an effective tool for information aggregation that generates an informative and concise summary from a cluster of topic-related documents. Our survey, the first of its kind, systematically overviews the recent deep learning based MDS models. We propose a novel taxonomy to summarize the design strategies of neural networks and conduct a comprehensive summary of the state-of-the-art. We highlight the differences between various objective functions that are rarely discussed in the existing literature. Finally, we propose several future directions pertaining to this new and exciting field.


Additional Key Words and Phrases: Multi-document summarization, Deep neural networks, Machine learning

1 INTRODUCTION

In this era of rapidly advancing technology, the exponential increase of data availability makes analyzing and understanding text files a tedious, labor-intensive, and time-consuming task [65, 120]. The need to process this abundance of text data rapidly and efficiently calls for new, effective text summarization techniques. Text summarization is a key natural language processing (NLP) task that automatically converts a text, or a collection of texts within the same topic, into a concise summary that contains key semantic information which can be beneficial for many downstream applications such as creating news digests, search engine, and report generation [127].

Text can be summarized from one or several documents, resulting in single document summarization (SDS) and multi-document summarization (MDS). While simpler to perform, SDS may not produce comprehensive summaries because it does not make good use of related, or more recent, documents. Conversely, MDS generates more comprehensive and accurate summaries from documents written at different times, covering different perspectives, but is accordingly more complicated as it tries to resolve potentially diverse and redundant information [151]. In addition, incredibly lengthy input documents often lead to model degradation [74]. It is challenging for models to retain the most critical contents of complex input sequences while generating a coherent,
non-redundant, factually consistent, and grammatically readable summary. Therefore, MDS requires models to have stronger capabilities for analyzing the input documents, identifying and merging consistent information.

MDS enjoys a wide range of real-world applications, including summarization of news [44], scientific publications [172], emails [23, 176], product reviews [50], medical documents [1], lecture feedback [102], software project activities [3], and Wikipedia articles [97]. Recently, MDS technology has also received a great amount of industry attention; an intelligent multilingual news reporter bot named Xiaomingbot [166] was developed for news generation, which can summarize multiple news sources into one article and translate it into multiple languages. Massive application requirements and rapidly growing online data have promoted the development of MDS. Existing methods using traditional algorithms are based on: term frequency-inverse document frequency (TF-IDF) [11, 131], clustering [52, 159], graphs [104, 158] and latent semantic analysis [8, 60]. Most of these works still generate summaries with manually crafted features [108, 158], such as sentence position features [12, 41], sentence length features [41], proper noun features [157], cue-phrase features [59], biased word features, sentence-to-sentence cohesion and sentence-to-centroid cohesion.

Deep learning has gained enormous attention in recent years due to its success in various domains, for instance, computer vision [81], natural language processing [36] and multi-modal learning [67]. Both industry and academia have embraced deep learning to solve complex tasks due to its capability of capturing highly nonlinear relations of data. Moreover, deep learning based models reduce dependence on manual feature extraction and pre-knowledge in the field of linguistics, drastically improving the ease of engineering [152]. Therefore, deep learning based methods demonstrate outstanding performance in MDS tasks in most cases [21, 85, 94, 98, 101]. With recent dramatic improvements in computational power and the release of increasing numbers of public datasets, neural networks with deeper layers and more complex structures have been applied in MDS [93, 98], accelerating the development of text summarization with more powerful and robust models. These tasks are attracting attention in the natural language processing community; the number of research publications on deep learning based MDS has increased rapidly over the last five years. The prosperity of deep learning for summarization in both academia and industry requires a comprehensive review of current publications for researchers to better understand the process and research progress. However, most of the existing summarization survey papers are based on traditional algorithms instead of deep learning based methods or target general text summarization [39, 46, 61, 116, 142].

We have therefore surveyed recent publications on deep learning methods for MDS that, to the best of our knowledge, is the first comprehensive survey of this field. This survey has been designed to classify neural based MDS techniques into diverse categories thoroughly and systematically. We also conduct a detailed discussion on the categorization and progress of these approaches to establish a clearer concept standing in the shoes of readers. We hope this survey provides a panorama for researchers, practitioners and educators to quickly understand and step into the field of deep learning based MDS. The key contributions of this survey are three-fold:

- We propose a categorization scheme to organize current research and provide a comprehensive review for deep learning based MDS techniques, including deep learning based models, objective functions, benchmark datasets, and evaluation metrics.
- We review development movements and provide a systematic overview and summary of the state-of-the-art. We also summarize nine network design strategies based on our extensive studies of the current models.
- We discuss the open issues of deep learning based multi-document summarization and identify the future research directions of this field. We also propose potential solutions for some discussed research directions.

**Paper Selection.** We used Google Scholar as the main search engine to select representative works from 2015 to 2021. High-quality papers were selected from top NLP and AI journals and conferences, include ACL\(^1\), EMNLP\(^2\),

\(^1\)Annual Meeting of the Association for Computational Linguistics.
\(^2\)Empirical Methods in Natural Language Processing.

Fig. 1. Hierarchical Structure of This Survey.

COLING³, NAACL⁴, AAAI⁵, ICML⁶, ICLR⁷ and IJCAI⁸. The major keywords we used include multi-documentation summarization, summarization, extractive summarization, abstractive summarization, deep learning and neural networks.

Organization of the Survey. This survey will cover various aspects of recent advanced deep learning based works in MDS. Our proposed taxonomy categorizes the works from six aspects (Figure 1). To be more self-contained, in Section 2, we give the problem definition, the processing framework of text summarization, discuss similarities and differences between SDS and MDS. Nine deep learning architecture design strategies, six deep learning based methods, and the variant tasks of MDS are presented in Section 3. Section 4 summarizes objective functions that guide the model optimization process in the reviewed literature while evaluation metrics in Section 5 help readers choose suitable indices to evaluate the effectiveness of a model. Section 6 summarizes standard

³International Conference on Computational Linguistics
⁴Annual Conference of the North American Chapter of the Association for Computational Linguistics.
⁵AAAI Conference on Artificial Intelligence.
⁶International Conference on Machine Learning.
⁷International Conference on Learning Representations
⁸International Joint Conference on Artificial Intelligence.

ACM Comput. Surv.
Deep Learning based Model
Sentences Selection
OR Words Generation
Representation Fusion
Form the Summary
Documents
Concatenation
Preprocessing
Fig. 2. The Processing Framework of Text Summarization. Each of the highlighted steps (the one with the triangle mark) indicates the differences between SDS and MDS.

and the variant MDS datasets. Finally, Section 7 discusses future research directions for deep learning based MDS followed by conclusions in Section 8.

2 FROM SINGLE TO MULTI-DOCUMENT SUMMARIZATION

Before we dive into the details of existing deep learning based techniques, we start by defining SDS and MDS, and introducing the concepts used by both methods. The aim of MDS is to generate a concise and informative summary $\text{Sum}$ from a collection of documents $D$. $D$ denotes a cluster of topic-related documents $\{d_i \mid i \in [1, N]\}$, where $N$ is the number of documents. Each document $d_i$ consists of $M_{d_i}$ sentences $\{s_{i,j} \mid j \in [1, M_{d_i}]\}$. $s_{i,j}$ refers to the $j$-th sentence in the $i$-th document. The standard summary $\text{Ref}$ is called the golden summary or reference summary. Currently, most golden summaries are written by experts. We keep this notation consistent throughout the article.

To give readers a clear understanding of the processing of deep learning based summarization tasks, we summarize and illustrate the processing framework as shown in Figure 2. The first step is preprocessing input document(s), such as segmenting sentences, tokenizing non-alphabetic characters, and removing punctuation [144]. MDS models in particular need to select suitable concatenation methods to capture cross-document relations. Then, an appropriate deep learning based model is chosen to generate semantic-rich representation for downstream tasks. The next step is to fuse these various types of representation for later sentence selection or summary generation. Finally, document(s) are transformed into a concise and informative summary. Each of the highlighted steps in Figure 2 (indicated by triangles) indicates a difference between SDS and MDS. Based on this process, the research questions of MDS can be summarized as follows:

- How to capture the cross-document relations and in-document relations from the input documents?
- Compared to SDS, how to extract or generate salient information in a larger search space containing conflict, duplication, and complementary information?
- How to best fuse various representation from deep learning based models and external knowledge?
- How to comprehensively evaluate the performance of MDS models?

The following sections provide a comprehensive analysis of the similarities and differences between SDS and MDS.

2.1 Similarities between SDS and MDS

Existing SDS and MDS methods share the summarization construction types, learning strategies, evaluation indexes and objective functions. SDS and MDS both seek to compress the document(s) into a short and informative summary. Existing summarization methods can be grouped into abstractive summarization, extractive summarization and hybrid summarization (Figure 3). Extractive summarization methods select salient snippets from the source documents to create informative summaries, and generally contain two major components: sentence ranking and sentence selection [20, 112]. Abstractive summarization methods aim to present the main information of input documents by automatically generating summaries that are both succinct and coherent; this cluster of methods allows models to generate new words and sentences from a corpus pool [127]. Hybrid models are proposed to combine the advantages of both extractive and abstractive methods to process the input texts.
Research on summarization focuses on two learning strategies. One strategy seeks to enhance the generalization performance by improving the architecture design of the end-to-end models [31, 44, 74, 98]. The other leverages external knowledge or other auxiliary tasks to complement summary selection or generation [19, 94]. Furthermore, both SDS and MDS aim to minimize the distance between machine-generated summary and golden summary. Therefore, SDS and MDS could share some indices to evaluate the performance of summarization models such as Recall-Oriented Understudy for Gisting Evaluation (ROUGE, see Section 5), and objective functions (see Section 4) to guide model optimization.

2.2 Differences between SDS and MDS

In the early stages of MDS, researchers directly applied SDS models to MDS [105]. However, a number of aspects in MDS that are different from SDS and these differences are also the breakthrough point for exploring the MDS models. We summarize the differences in the following five aspects:

- **More diverse input document types;**
- **Insufficient methods to capture cross-document relations;**
- **High redundancy and contradiction across input documents;**
- **Larger searching space but lack of sufficient training data;**
- **Lack of evaluation metrics specifically designed for MDS.**

A defining different character between SDS and MDS is the number of input documents. MDS tasks deal with multiple sources, of types that can be roughly divided into three groups:

- **Many short sources**, where each document is relatively short but the quantity of the input data is large. A typical example is product reviews summarization that aims to generate a short, informative summary from numerous individual reviews [5].
- **Few long sources.** For example, generating a summary from a group of news articles [44], or constructing a Wikipedia-style article from several web articles [97].
- **Hybrid sources** containing one or few long documents with several to many shorter documents. For example, news article(s) with several readers’ comments to this news [92], or a scientific summary from a long paper with several short corresponding citations [172].

As SDS only uses one input document, no additional processing is required to assess relationships between SDS inputs. By their very nature, the multiple input documents used in MDS are likely to contain more contradictory, redundant, and complementary information [130]. MDS models therefore require sophisticated algorithms to identify and cope with redundancy and contradictions across documents to ensure that the final summary is comprehensive. Detecting these relations across documents can bring benefits for MDS models. In MDS tasks, there are two common methods to concatenate multiple input documents:
Flat concatenation is a simple yet powerful concatenation method, where all input documents are spanned and processed as a flat sequence; to a certain extent, this method converts MDS to SDS tasks. Inputting flat-concatenated documents requires models to have a strong ability to process long sequences.

Hierarchical concatenation is able to preserve cross-document relations. However, many existing deep learning methods do not make full use of this hierarchical relationship [44, 97, 160]. Taking advantage of hierarchical relations among documents instead of simply flat concatenating articles facilitates the MDS model to obtain representation with built-in hierarchical information, which in turn improves the effectiveness of the models. The input documents within a cluster describe a similar topic logically and semantically. Figure 4 illustrates two representative methods of hierarchical concatenation. Existing hierarchical concatenation methods either perform document-level condensing in a cluster separately [4] or process documents in word/sentence-level inside document cluster [6, 114, 160]. In Figure 4(a), the extractive or abstractive summaries, or representation from the input documents are fused in the subsequent processes for final summaries generation. The models using document-level concatenation methods are usually two-stage models. In Figure 4(b), sentences in the documents can be replaced by words. For word or sentence-level concatenation methods, clustering algorithms and graph-based techniques are the most commonly used methods. Clustering methods could help MDS models decrease redundancy and increase the information coverage for the generated summaries [114]. Sentence relation graph is able to model hierarchical relations among multi-documents as well [6, 172, 173]. Most of the graph construction methods utilize sentences as vertexes and the edge between two sentences indicates their sentence-level relations [6]. Cosine similarity graph [41], discourse graph [30, 98, 173], semantic graph [126] and heterogeneous graph [160] can be used for building sentence graph structures. These graph structures could all serve as an external knowledge to improve the performance of MDS models.
In addition to capture cross-document relation, hybrid summarization models can also be used to capture complex documents semantically, as well as to fuse disparate features that are more commonly adopted by MDS tasks. These models usually process data in two stages: extractive-abstractive and abstractive-abstractive (the right part of Figure 3). The two-stage models try to gather important information from source documents with extractive or abstractive methods at the first stage, to significantly reduce the length of documents. In the second stage, the processed texts are fed into an abstractive model to form final summaries [4, 85, 94, 97, 98].

Furthermore, conflict, duplication, and complementarity among multiple source documents require MDS models to have stronger abilities to handle complex information. However, applying the SDS model directly on MDS tasks is difficult to handle much higher redundancy [105]. Therefore, the MDS models are required not only to generate coherent and complete summary but also more sophisticated algorithms to identify and cope with redundancy and contradictions across documents ensuring that the final summary should be complete in itself. MDS also involves larger searching spaces but has smaller-scale training data than SDS, which sets obstacles for deep learning based models to learn adequate representation [105]. In addition, there are no specific evaluation metrics designed for MDS; however, existing SDS evaluation metrics cannot evaluate the relationship between the generated abstract and different input documents well.

3 DEEP LEARNING BASED MULTI-DOCUMENT SUMMARIZATION METHODS

Deep neural network (DNN) models learn multiple levels of representation and abstraction from input data and can fit data in a variety of research fields, such as computer vision [81] and natural language process [36]. Deep learning algorithms replace manual feature engineering by learning distinctive features through back-propagation to minimize a given objective function. It is well known that linear solvable problems possess many advantages, such as being easily solved and having numerous theoretically proven supports; however, many NLP tasks are highly non-linear. As theoretically proven by Hornik et al. [64], neural networks can fit any given continuous function as a universal approximator. For MDS tasks, DNNs also perform considerably better than traditional methods to effectively process large-scale documents and distill informative summaries due to their strong fitting abilities. In this section, we first introduce our novel taxonomy that generalizes nine neural network design strategies (Section 3.1). We then present the state-of-the-art DNN based MDS models according to the main neural network architecture they adopt (Section 3.2 – 3.7), before finishing with a brief introduction to MDS variant tasks (Section 3.8).

3.1 Architecture Design Strategies

Architecture design strategies play a critical role in deep learning based models, and many architectures have been applied to variants MDS tasks. Here, we have generalized the network architectures and summarized them into nine types based on how they generate or fuse semantic-rich and syntactic-rich representation to improve MDS model performance (Figure 5); these different architectures can also be used as basic structures or stacked on each other to obtain more diverse design strategies. In Figure 5, deep neural models are in green boxes and can be flexibly substituted with other backbone networks. The blue boxes indicate the neural embeddings processed by neural networks or heuristic-designed approaches, e.g., "sentence/document" or "other" representation. The explanation for each sub-figure is listed as follows:

- **Naive Networks (Figure 5(a)).** Multiple concatenated documents are input through DNN based models to extract features. Word-level, sentence-level or document-level representation is used to generate the downstream summary or select sentences. Naive networks represent the most naive model that lays the foundation for other strategies.

- **Ensemble Networks (Figure 5(b)).** Ensemble based methods leverage multiple learning algorithms to obtain better performance than individual algorithms. To capture semantic-rich and syntactic-rich representation, ensemble networks feed input documents to multiple paths with different network structures or operations. Later on, the representation from different networks is fused to enhance model expression capability. The majority vote or the average score can be used to determine the final output.
Auxiliary Task Networks (Figure 5(c)) employ different tasks in the summarization models, where text classification, text reconstruction, or other auxiliary tasks serve as complementary representation learners to obtain advanced features. Meanwhile, auxiliary task networks also provide researchers with a solution to use appropriate data from other tasks. In this strategy, parameter sharing schemes are used for jointly optimizing different tasks.

Reconstruction Networks (Figure 5(d)) optimize models from an unsupervised learning paradigm, which allows summarization models to overcome the limitation of insufficient annotated golden summaries. The use of such a paradigm enables generated summaries to be constrained in the natural language domain in a good manner.
• **Fusion Networks (Figure 5(e))** fuse representation generated from neural networks and hand-crafted features. These hand-crafted features contain adequate prior knowledge that facilitates the optimization of summarization models.

• **Graph Neural Networks (Figure 5(f)).** This strategy captures cross-document relations, crucial and beneficial for multi-document model training, by constructing graph structures based on the source documents, including word, sentence, or document-level information.

• **Encoder-Decoder Structure (Figure 5(g)).** The encoder embeds source documents into the hidden representation, i.e., word, sentence and document representation. This representation, containing compressed semantic and syntactic information, is passed to the decoder which processes the latent embeddings to synthesize local and global semantic/syntactic information to produce the final summaries.

• **Pre-trained Language Models (Figure 5(h))** obtain contextualized text representation by predicting words or phrases based on their context using large amounts of the corpus, which can be further fine-tuned for downstream task adaption [37]. The models can fine-tune with randomly initialized decoders in an end-to-end fashion since transfer learning can assist the model training process [94].

• **Hierarchical Networks (Figure 5(i)).** Multiple documents are concatenated as inputs to feed into the first DNN based model to capture low-level representation. Another DNN based model is cascaded to generate high-level representation based on the previous ones. The hierarchical networks empower the model with the ability to capture abstract-level and semantic-level features more efficiently.

### 3.2 Recurrent Neural Networks based Models

Recurrent Neural Networks (RNNs) [137] excel in modeling sequential data by capturing sequential relations and syntactic/semantic information from word sequences. In RNN models, neurons are connected through hidden layers and unlike other neural network structures, the inputs of each RNN neuron come not only from the word or sentence embedding but also from the output of the previous hidden state. Despite being powerful, vanilla RNN models often encounter gradient explosion or vanishing issues, so a large number of RNN-variants have been proposed. The most prevalent ones are Long Short-Term Memory (LSTM) [63], Gated Recurrent Unit (GRU) [32] and Bi-directional Long Short-Term Memory (Bi-LSTM) [66]. The DNN based Model in Figure 5 can be replaced with RNN based models to design models.

RNN based models have been used in MDS tasks since 2015. Cao et al. [20] proposed an RNN-based model termed *Ranking framework upon Recursive Neural Networks (R2N2)*, which leverages manually extracted words and sentence-level features as inputs. This model transfers the sentence ranking task into a hierarchical regression process, which measures the importance of sentences and constituents in the parsing tree. Zheng et al. [187] used a hierarchical RNN structure to utilize the subtopic information by extracting not only sentence and document embeddings, but also topic embeddings. In this SubTopic-Driven Summarization (STDS) model, the readers’ comments are seen as auxiliary documents and the model employs soft clustering to incorporate comment and sentence representation for further obtaining subtopic representation. Arthur et al. [16] introduced a GRU-based encoder-decoder architecture to minimize the diversity of opinions reflecting the dominant views while generating multi-review summaries. Mao et al. [105] proposed a maximal margin relevance guided reinforcement learning framework (RMMR) to incorporate the advantages of neural sequence learning and statistical measures. The proposed soft attention for learning adequate representation allows more exploration of search space.

To leverage the advantage of the hybrid summarization model, Reinald et al. [4] proposed a two-stage framework, viewing opinion summarization as an instance of multi-source transduction to distill salient information from source documents. The first stage of the model leverages a Bi-LSTM auto-encoder to learn word and document-level representation; the second stage fuses multi-source representation and generates an opinion summary with a simple LSTM decoder combined with a vanilla attention mechanism [9] and a copy mechanism [156].

Since paired MDS datasets are rare and hard to obtain, Li et al. [93] developed a RNN-based framework to extract salient information vectors from sentences in input documents in an unsupervised manner. Cascaded attention retains the most relevant embeddings to reconstruct the original input sentence vectors. During the
reconstruction process, the proposed model leverages a sparsity constraint to penalize trivial information in the output vectors. Also, Chu et al. [31] proposed an unsupervised end-to-end abstractive summarization architecture called MeanSum. This LSTM-based model formalizes product or business reviews summarization problem into two individual closed-loops. Inspired by MeanSum, Coavoux et al. [33] used a two-layer standard LSTM to construct sentence representation for aspect-based multi-document abstractive summarization, and discovered that the clustering strategy empowers the model to reward review diversity and handle contradictory ones.

3.3 Convolutional Neural Networks Based Models

Convolutional neural networks (CNNs) [87] achieve excellent results in computer vision tasks. The convolution operation scans through the word/sentence embeddings and uses convolution kernels to extract important information from input data objects. Using a pooling operation at intervals can return simple to complex feature levels. CNNs have been proven to be effective for various NLP tasks in recent years [38, 77] as they can process natural language after sentence/word vectorization. Most of the CNN based MDS models use CNNs for semantic and syntactic feature representation. As with RNN, CNN-based models can also replace DNN-based models in network design strategies (Please refer to Figure 5).

A simple way to use CNNs in MDS is by sliding multiple filters with different window sizes over the input documents for semantic representation. Cao et al. [21] proposed a hybrid CNN-based model PriorSum to capture latent document representation. The proposed representation learner slides over the input documents with filters of different window widths and two-layer max-over-time pooling operations [34] to fetch document-independent features that are more informative than using standard CNNs. Similarly, HNet [145] uses distinct CNN filters and max-over-time-pooling to generate salient feature representation for downstream processes. Cho et al. [29] also used different filter sizes in DPP-combined model to extract low-level features. Yin et al. [174] presented an unsupervised CNN-based model termed Novel Neural Language Model (NNLM) to extract sentence representation and diminish the redundancy of sentence selection. The NNLM framework contains only one convolution layer and one max-pooling layer, and both element-wise averaging sentence representation and context words representation are used to predict the next word. For aspect-based opinion summarization, Stefanos et al. [5] leveraged a CNN based model to encode the product reviews which contain a set of segments for opinion polarity.

People with different background knowledge and understanding can produce different summaries of the same documents. To account for this variability, Zhang et al. [184] suggested a MV-CNN model that ensembles three individual models to incorporate multi-view learning and CNNs to improve the performance of MDS. In this work, three CNNs with dual-convolutional layers used multiple filters with different window sizes to extract distinct saliency scores of sentences.

To overcome the MDS bottlenecks of insufficient training data, Cao et al. [19] developed a TCSum model incorporating an auxiliary text classification sub-task into MDS to introduce more supervision signals. The text classification model uses a CNN descriptor to project documents onto the distributed representation, and to classify input documents into different categories. The summarization model shares the projected sentence embedding from the classification model, and the TCSum model then chooses the corresponding category based transformation matrices according to classification results to transform the sentence embedding into the summary embedding.

Unlike RNNs that support the processing of long time-serial signals, a naive CNN layer struggles to capture long-distance relations while processing sequential data due to the limitation of the fixed-sized convolutional kernels, each of which has a specific receptive field size. Nevertheless, CNN based models can increase their receptive fields through formation of hierarchical structures to calculate sequential data in a parallel manner. Because of this highly parallelizable characteristic, training of CNN-based summarization models is more efficient than for RNN-based models. However, summarizing lengthy input articles is still a challenging task for CNN based models because they are not skilled in modeling non-local relationships.
Table 1. Multi-document Summarization Models based on Graph Neural Networks.

<table>
<thead>
<tr>
<th>Models</th>
<th>Nodes</th>
<th>Edges</th>
<th>Edge Weights</th>
<th>GNN Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>HeterDoc-SumGraph [160]</td>
<td>word, sentence, document</td>
<td>word-sentence, word-document</td>
<td>TF-IDF</td>
<td>Graph Attention Networks</td>
</tr>
<tr>
<td>Graph-based Neural MDS [173]</td>
<td>sentence</td>
<td>sentence-sentence</td>
<td>Personalized Discourse Graph</td>
<td>Graph Convolutional Networks</td>
</tr>
<tr>
<td>SemSentSum [6]</td>
<td>sentence</td>
<td>sentence-sentence</td>
<td>Cosine Similarity Graph</td>
<td>Graph Convolutional Networks</td>
</tr>
<tr>
<td>ScisummNet [172]</td>
<td>sentence</td>
<td>sentence-sentence</td>
<td>Cosine Similarity Graph</td>
<td>Graph Convolutional Networks</td>
</tr>
</tbody>
</table>

3.4 Graph Neural Networks Based Models

CNNs have been successfully applied to many computer vision tasks to extract distinguished image features from the Euclidean space, but struggle when processing non-Euclidean data. Natural language data consist of vocabularies and phrases with strong relations which can be better represented with graphs than with sequential orders. Graph neural networks (GNNs, Figure 5 (f)) are composed of an ideal architecture for NLP since they can model strong relations between entities semantically and syntactically. Graph convolution networks (GCNs) and graph attention networks (GAns) are the most commonly adopted GNNs because of their efficiency and simplicity for integration with other neural networks. These models first build a relation graph based on input documents, where nodes can be words, sentences or documents, and edges capture the similarity among them. At the same time, input documents are fed into a DNN based model to generate embeddings at different levels. The GNNs are then built over the top to capture salient contextual information. Table 1 describes the current GNN based models used for MDS with details of nodes, edges, edge weights, and applied GNN methods.

Yasunage et al. [173] developed a GCN based extractive model to capture the relations between sentences. This model first builds a sentence-based graph and then feeds the pre-processed data into a GCN [78] to capture sentence-wise related features. Defined by the model, each sentence is regarded as a node and the relation between each pair of sentences is defined as an edge. Inside each document cluster, the sentence relation graph can be generated through a cosine similarity graph [41], approximate discourse graph [30], and the proposed personalized discourse graph. Both the sentence relation graph and sentence embeddings extracted by a sentence-level RNN are fed into GCN to produce the final sentence representation. With the help of a document-level GRU, the model generates cluster embeddings to fully aggregate features between sentences.

Similarly, Antognini et al. [6] proposed a GCN based model named SemSentSum that constructs a graph based on sentence relations. In contrast to Yasunage et al. [173], this work leverages external universal embeddings, pre-trained on the unrelated corpus, to construct a sentence semantic relation graph. Additionally, an edge removal method has been applied to deal with the sparse graph problems emphasizing high sentence similarities; if the weight of the edge is lower than a given threshold, the edge is removed. The sentence relation graph and sentence embeddings are fed into a GCN [78] to generate saliency estimation for extractive summaries.

Yasunage et al. [172] also designed a GCN based model for summarizing scientific papers. The proposed ScisummNet model uses not only the abstract of source scientific papers but also the relevant text from papers that cite the original source. The total number of citations is also incorporated into the model as an authority feature. A cosine similarity graph is applied to form the sentence relation graph, and GCNs are adopted to predict the sentence salience estimation from the sentence relation graph, authority scores and sentence embeddings.

Existing GNN based models focused mainly on the relationships between sentences, and do not fully consider the relationships between words, sentences, and documents. To fill this gap, Wang et al. [160] proposed a heterogeneous GAN based model, called HeterDoc-SUM Graph, that is specific for extractive MDS. This heterogeneous graph structure includes word, sentence, and document nodes, where sentence nodes and document nodes are connected according to the contained word nodes. Word nodes thus act as an intermediate bridge to connect the sentence and document nodes, and are used to better establish document-document, sentence-sentence and sentence-document relations. TF-IDF values are used to weight word-sentence and word-document edges, and...
the node representation of these three levels are passed into the graph attention networks for model update. In each iteration, bi-directional updating of both word-sentence and word-document relations are performed to better aggregate cross-level semantic knowledge.

3.5 Pointer-generator Networks Based Models

Pointer-generator (PG) networks [139] are proposed to overcome the problems of factual errors and high redundancy in the summarization tasks. This network has been inspired by Pointer Network [156], CopyNet [57], forced-attention sentence compression [107], and coverage mechanism from machine translation [153]. PG networks combine sequence-to-sequence (Seq2Seq) model and pointer networks to obtain a united probability distribution allowing vocabularies to be selected from source texts or generated by machines. Additionally, the coverage mechanism prevents PG networks from consistently choosing the same phrases.

The Maximal Marginal Relevance (MMR) method is designed to select a set of salient sentences from source documents by considering both importance and redundancy indices [22]. The redundancy score controls sentence selection to minimize overlap with the existing summary. The MMR model adds a new sentence to the objective summary based on importance and redundancy scores until the summary length reaches a certain threshold.

Inspired by MMR, Alexander et al. [44] proposed an end-to-end Hierarchical MMR-Attention Pointer-generator (Hi-MAP) model to incorporate PG networks and MMR [22] for abstractive MDS. The Hi-MAP model improves PG networks by modifying attention weights (multiplying MMR scores by the original attention weights) to include better important sentences in, and filter redundant information from, the summary. Similarly, the MMR approach is implemented by PG-MMR model [86] to identify salient source sentences from multi-document inputs, albeit with a different method for calculating MMR scores from Hi-MAP; instead, ROUGE-L Recall and ROUGE-L Precision [95] serve as evaluation metrics to calculate the importance and redundancy scores. To overcome the scarcity of MDS datasets, the PG-MMR model leverages a support vector regression model that is pre-trained on a SDS dataset to recognize the important contents. This support vector regression model also calculates the score of each input sentence by considering four factors: sentence length, sentence relative/absolute position, sentence-document similarities, and sentence quality obtained by a PG network. Sentences with the top-$K$ scores are fed into another PG network to generate a concise summary.

3.6 Transformer Based Models

As discussed, CNN based models are not as good at processing sequential data as RNN based models. However, RNN based models are not amenable to parallel computing, as the current states in RNN models highly depend on results from the previous steps. Additionally, RNNs struggle to process long sequences since former knowledge will fade away during the learning process. Adopting Transformer based architectures [155] is one solution to solve these problems. The Transformer is based on the self-attention mechanism, has natural advantages for parallelization, and retains relative long-range dependencies. The Transformer model has achieved promising results in MDS tasks [74, 94, 97, 98] and can replace the DNN based Model in Figure 5. Most of the Transformer based models follow an encoder-decoder structure. Transformer based models can be divided into flat Transformer, hierarchical Transformer, and pre-train language models.

Flat Transformer. Liu et al. [97] introduced Transformer to MDS tasks, aiming to generate a Wikipedia article from a given topic and set of references. The authors argue that the encoder-decoder based sequence transduction model cannot cope well with long input documents, so their model selects a series of top-$K$ tokens and feeds them into a Transformer based decoder-only sequence transduction model to generate Wikipedia articles. More specifically, the Transformer decoder-only architecture combines the results from the extractive stage and golden summary into a sentence for training. To obtain rich semantic representation from different granularity, Jin et al. [74] proposed a Transformer based multi-granularity interaction network MSGSum and unified extractive and abstractive MDS. Words, sentences, and documents are considered as three granular levels of semantic unit connected by a granularity hierarchical relation graph. In the same granularity, a self-attention mechanism is used to capture the semantic relationships. Sentence granularity representation is employed in the extractive
summarization, and word granularity representation is adapted to generate an abstractive summary. MGSum employs a fusion gate to integrate and update the semantic representation. Additionally, a spare attention mechanism is used to ensure the summary generator focus on important information. Brazinskas et al. [17] created a precedent for few-shot learning for MDS that leverages a Transformer conditional language model and a plug-in network for both extractive and abstractive MDS to overcome rapid overfitting and poor generation problems resulting from naive fine-tuning of large parameter models.

**Hierarchical Transformer.** To handle huge amounts of input documents (currently many large scale MDS datasets contain more than ten thousand input document sets), Yang et al. [98] proposed a two-stage Hierarchical Transformer (HT) model with an inter-paragraph and graph-informed attention mechanism that allows the model to encode multiple input documents hierarchically instead of by simple flat-concatenation. A logistic regression model is employed to select the top-K paragraphs, which are fed into a local Transformer layer to obtain contextual features. A global Transformer layer mixes the contextual information to model the dependencies of the selected paragraphs. To leverage graph structure to capture cross-document relations, Li et al. [94] proposed an end-to-end Transformer based model GraphSum, based on the HT model. In the graph encoding layers, GraphSum extends the self-attention mechanism to the graph-informed self-attention mechanism, which incorporates the graph representation into the Transformer encoding process. Furthermore, the Gaussian function is applied to the graph representation matrix to control the intensity of the graph structure’s impact on the summarization model. The HT and GraphSum models are both based on the self-attention mechanism leading quadratic memory growth increases with the number of input sequences; to address this issue, Pasunuru et al. [126] modified the full self-attention with local and global attention mechanism [14] to scale the memory linearly. Dual encoders are proposed for encoding truncated concatenated documents and linearized graph information from full documents.

**Pre-trained language models (LMs).** Pre-trained Transformers on large text corpora have shown great successes in downstream NLP tasks including text summarization. The pre-trained LMs can be trained on non-summarization or SDS datasets to overcome lack of MDS data [94, 126, 178]. Most pre-trained LMs such as BERT [35] and RoBERTa [99] can work well on short sequences. In hierarchical Transformer architecture, replacing the low-level Transformer (token-level) encoding layer with pre-trained LMs helps the model breakthrough length limitations to perceive further information [94]. Inside a hierarchical Transformer architecture, the output vector of the "[CLS]" token can be used as input for high-level Transformer models. To avoid the self-attention quadratic-memory increment when dealing with document-scale sequences, a Longformer based approach [14], including local and global attention mechanisms, can be incorporated with pre-trained LMs to scale the memory linearly for MDS [126]. Another solution for computational issues can be borrowed from SDS is to use a multi-layer Transformer architecture to scale the length of documents allowing pre-trained LMs to encode a small block of text and the information can be shared among the blocks between two successive layers [55]. BART [88], GPT-2 [133], and T5 [134] are pre-trained language models that can be used for language generation and they have been applied for MDS tasks [2, 122, 147]. Instead of regular language models, PEGASUS [178] is a pre-trained Transformer-based encoder-decoder model with gap-sentences generation (GSG) that focused on abstractive summarization. GSG shows that masking whole sentences based on importance, instead of through random or lead selection, works well for downstream summarization tasks. BART, T5, and PEGASUS are based on data-rich SDS settings. Goodwin et al. [54] evaluated these three pre-trained models on four MDS datasets and suggested that while large improvements have been made on the standard SDS task, highly abstractive MDS remains a challenge. PRIMER [164] is a pre-trained model specifically designed for MDS which can serve as a zero-shot summarizer.

### 3.7 Deep Hybrid Models

Many neural models can be integrated to formalize a more powerful and expressive model. In this section, we summarize the existing deep hybrid models that have proven to be effective for MDS.
CNN + LSTM + Capsule networks. Cho et al. [29] proposed a hybrid model based on the determinantal point processes for semantically measuring sentence similarities. A convolutional layer slides over the pairwise sentences with filters of different sizes to extract low-level features. Capsule networks [138, 168] are employed to identify redundant information by transforming the spatial and orientational relationships for high-level representation. The authors also used LSTM to reconstruct pairwise sentences and add reconstruction loss to the final objective function.

CNN + Bi-LSTM + Multi-layer Perceptron (MLP). Abhishek et al. [145] proposed an extractive MDS framework that considers document-dependent and document-independent information. In this model, a CNN with different filters captures phrase-level representation. Full binary trees formed with these salient representation are fed to the recommended Bi-LSTM tree indexer to enable better generalization abilities. A MLP with ReLU function is employed for leaf node transformation. More specifically, the Bi-LSTM tree indexer leverages the time serial power of LSTMs and the compositionality of recursive models to capture both semantic and compositional features.

PG networks + Transformer. In generating a summary, it is necessary to consider the information fusion of multiple sentences, especially sentence pairs. Logan et al. [85] found the majority of summary sentences are generated by fusing one or two source sentences; so they proposed a two-stage summarization method that considers the semantic compatibility of sentence pairs. This method joint-scores single sentence and sentence pairs to filter representative from the original documents. Sentences or sentence pairs with high scores are then compressed and rewritten to generate a summary that leverages PG network. This paper uses a Transformer based model to encode both single sentence and sentence pairs indiscriminately to obtain the deep contextual representation of words and sequences.

3.8 The Variants of Multi-document Summarization

In this section, we briefly introduce several MDS task variants to give researchers a comprehensive understanding of MDS. These tasks can be modeled as MDS problems and adopt the aforementioned deep learning techniques and neural network architectures.

Query-oriented MDS calls for a summary from a set of documents that answers a query. It tries to solve realistic query-oriented scenario problems and only summarizes important information that best answers the query in a logical order [125]. Specifically, query-oriented MDS combines the information retrieval and MDS techniques. The content that needs to be summarized is based on the given queries. Liu et al. [98] incorporated the query by simply prepending the query to the top-ranked document during encoding. Pasunuru [125] involved a query encoder and integrated query embedding into an MDS model, ranking the importance of documents for a given query.

Dialogue summarization aims to provide a succinct synopsis from multiple textual utterances of two or more participants, which could help quickly capture relevant information without having to listen to long and convoluted dialogues [96]. Dialogue summary covers several areas, including meetings [45, 80, 192], email threads [180], medical dialogues [40, 75, 146], customer service [96] and media interviews [191]. Challenges in dialogue summarization can be summarized into the following seven categories: informal language use, multiple participants, multiple turns, referral and coreference, repetition and interruption, negations and rhetorical questions, role and language change [25]. The flow of the dialogue would be neglected if MDS models are directly applied for dialogue summarization. Liu et al. [96] relied on human annotations to capture the logic of the dialogue. Wu et al. [163] used summary sketch to identify the interaction between speakers and their corresponding textual utterances in each turn. Chen et al. [25] proposed a multi-view sequence to sequence based encoder to extract dialogue structure and a multi-view decoder to incorporate different views to generate final summaries.
Table 2. Deep Learning based Methods. "Ext", "Abs" and "Hyd" mean extractive, abstractive and hybrid respectively; "FC" and "HC" represent Flat Concatenate, Hierarchical Concatenate respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Works</th>
<th>Construction Types</th>
<th>Document-level Relationship</th>
<th>Comparison of DL based techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN</td>
<td>MeanSum [31]</td>
<td>✓</td>
<td>✓</td>
<td>Pros: Can capture sequential relations and syntactic/semantic information from word sequences; Cons: Not easy to parallel computing; Highly depending on results from the previous steps</td>
</tr>
<tr>
<td></td>
<td>Zhang et al. [177]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STDS [181]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ParaFuse_doc [114]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R2N2 [20]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CondaSum [4]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C-Attention [93]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wang et al. [162]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RL-MMR [105]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Coavoux et al. [33]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>CNN</td>
<td>MV-CNN [184]</td>
<td>✓</td>
<td>✓</td>
<td>Pros: Good parallel computing; Cons: Not good at processing sequential data</td>
</tr>
<tr>
<td></td>
<td>TCSum [19]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CNNLM [174]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PriorSum [21]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Angelidis et al. [5]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>GNN</td>
<td>Yasunaga et al. [173]</td>
<td>✓</td>
<td>✓</td>
<td>Pros: Can capture cross-document and in-document relations; Cons: Inefficient when dealing with large graphs</td>
</tr>
<tr>
<td></td>
<td>SemiSentSum [6]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Scisummnet [172]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>PG</td>
<td>PG-MMR [86]</td>
<td>✓</td>
<td>✓</td>
<td>Pros: Low redundancy; Cons: Hard to train</td>
</tr>
<tr>
<td></td>
<td>Hi-MAP [44]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Transformer</td>
<td>HT [98]</td>
<td>✓</td>
<td>✓</td>
<td>Pros: Good performance; Good parallel computing; Can capture in-document relations; Cons: Time-consuming; Problems with position encoding</td>
</tr>
<tr>
<td></td>
<td>MSGSum [14]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FewSum [17]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GraphSum [94]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bart-Long [126]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WikiSum [97]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Deep Hybrid Model</td>
<td>Cho et al. [29]</td>
<td>✓</td>
<td>✓</td>
<td>Pros: Combines the advantages of different DL models</td>
</tr>
<tr>
<td></td>
<td>GT-SingPairMix [85]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HNet [145]</td>
<td>✓</td>
<td>✓</td>
<td>Cons: Computationally intensive</td>
</tr>
</tbody>
</table>

Stream summarization aims to summarize new documents in a continuously growing document stream, such as information from social media. Temporal summarization and real-time summarization (RTS) can be seen as a form of stream document summarization. Stream summarization considers both historical dependencies and future uncertainty of the document stream. Yang et al. [167] used deep reinforcement learning to solve the relevance, redundancy, and timeliness issues in steam summarization. Tan et al. [150] transformed the real time summarization task as a sequential decision-making problem and used a LSTM layer and three fully connected neural network layers to maximize the long-term rewards.

3.9 Discussion

In this section, we have reviewed the state-of-the-art works of deep learning based MDS models according to the neural networks applied. Table 2 summarizes the reviewed works by considering the type of neural networks, construction types, and concatenation methods; and provides a high-level summary of their relative advantages and disadvantages. Transformer based models have been most commonly used in the last three years because they overcome the limitations of CNN’s fixed-size receptive field and RNN’s inability to parallel process. However, deep learning based MDS models face some challenges. Firstly, the complexity of deep learning based models and the data-driven deep learning systems do require more training data, with concomitant increased efforts.

http://trecrts.github.io/
in data labelling, and computing resources than non-deep learning based methods. Inevitably, deep learning based MDS models require more computation during the training phase. During the inference process, they generally consume more computation power than non-deep learning based methods as well. Secondly, deep learning based methods lack linguistic knowledge that can serve as important roles in assisting deep learning based learners to have informative representation and better guide the summary generation. We believe that this is one possible reason that some non-deep learning based MDS methods sometimes show better performance than deep learning based methods [21, 101] as non-deep learning based methods pay more attention to linguistic information. We discuss this point in Section 7.1. Further researches could also be based on techniques adopted in non-deep learning based MDS as reviewed in [39, 46, 142]. Thirdly, deep learning based models can be regarded as black boxes with high non-linearity. It is challenging to understand the detailed transformation inside them. Exploring the interpretability of MDS models allows researchers to understand the effect of each module in MDS neural models, therefore guiding the model design with a more accurate target. However, there is little existing work on the interpretability of MDS models, which is of great help in improving the quality of summaries. More discussions about explainable deep learning MDS models can be found in Section 7.6.

4 OBJECTIVE FUNCTIONS

In this section, we will take a closer look at different objective functions adopted by various MDS models. In summarization models, objective functions play an important role by guiding the model to achieve specific purposes. To the best of our knowledge, we are the first to provide a comprehensive survey on different objectives of summarization tasks.

4.1 Cross-Entropy Objective

Cross-entropy usually acts as an objective function to measure the distance between two distributions. Many existing MDS models adopt it to measure the difference between the distributions of generated summaries and the golden summaries [20, 29, 160, 172, 177, 184]. Formally, the cross-entropy loss is defined as:

\[ L_{CE} = - \sum_{i=1}^{N} y_i \log(\hat{y}_i), \]  

where \( y_i \) is the target score from golden summaries and machine-generated summaries, and \( \hat{y}_i \) is the predicted estimation from the deep learning based models. Different from calculations in other tasks, such as text classification, in summarization tasks, \( y_i \) and \( \hat{y}_i \) have several methods to calculate. \( \hat{y}_i \) usually is calculated by Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Please refer to Section 5). For example, ROUGE-1 [6], ROUGE-2 [98] or the normalized average of ROUGE-1 and ROUGE-2 scores [173] could be adopted to compute the ground truth score between the selected sentences and golden summary.

4.2 Reconstructive Objective

Reconstructive objectives are used to train a distinctive representation learner by reconstructing the input vectors in an unsupervised learning manner. The objective function is defined as:

\[ L_{Rec} = \| x_i - \phi'(\phi(x_i; \theta); \theta') \|_*, \]  

where \( x_i \) represents the input vector; \( \phi \) and \( \phi' \) represent the encoder and decoder with \( \theta \) and \( \theta' \) as their parameters respectively, \( \| \cdot \|_* \) represents norm (* stands for 0, 1, 2, ..., infinity). \( L_{Rec} \) is a measuring function to calculate the distance between source documents and their reconstructive outputs. Chu et al. [31] used a reconstructive loss to constrain the generated text into the natural language domain, reconstructing reviews in a token-by-token manner. Moreover, this paper also proposes a variant termed reconstruction cycle loss. By using the variant, the reviews are encoded into a latent space to further generate the summary, and the summary is then decoded to the reconstructed reviews to form another reconstructive closed-loop. An unsupervised learning loss was designed by Li et al. [93] to reconstruct the condensed output vectors to the original input sentence vectors with \( L_2 \) distance. This paper further constrains the condensed output vector with a \( L_1 \) regularizer to ensure
sparsity. Similarly, Zheng et al. [187] adopted a bi-directional GRU encoder-decoder framework to reconstruct both news sentences and comment sentences in a word sequence manner. Liu et al. [97] concatenated both input and output sequences to predict the next token to train the abstractive model. There are also some variants, such as leveraging the latent vectors of variational auto-encoder for reconstruction to capture better representation. Li et al. [92] introduced three individual reconstructive losses to consider both news reconstruction and comments reconstruction separately, along with a variational auto-encoder lower bound. Bravzinskas et al. [16] utilized a variational auto-encoder to generate the latent vectors of given reviews, where each review is reconstructed by the latent vectors combined with other reviews.

4.3 Redundancy Objective

Redundancy is an important objective to minimize the overlap between semantic units in a machine-generated summary. By using this objective, models are encouraged to maximize information coverage. Formally,

$$L_{Red} = Sim(x_i, x_j),$$  \hspace{1cm} (3)

where $Sim(\cdot)$ is the similarity function to measure the overlap between different $x_i$ and $x_j$, which can be phrases, sentences, topics or documents. The redundancy objective is often treated as an auxiliary objective combined with other loss functions. Li et al. [93] penalized phrase pairs with similar meanings to eliminate the redundancy. Nayeem et al. [114] used the redundancy objective to avoid generating repetitive phrases, constraining a sentence to appear only once while maximizing the scores of important phrases. Zheng et al. [187] adopted a redundancy loss function to measure overlaps between subtopics; intuitively, smaller overlaps between subtopics resulted in less redundancy in the output domain. Yin et al. [174] proposed a redundancy objective to estimate the diversity between different sentences.

4.4 Max Margin Objective

Max Margin Objectives (MMO) are also used to empower the MDS models to learn better representation. The objective function is formalized as:

$$L_{Margin} = \max(0, f(x_i; \theta) - f(x_j; \theta) + \gamma),$$  \hspace{1cm} (4)

where $x_i$ and $x_j$ represent the input vectors, $\theta$ are parameters of the model function $f(\cdot)$, and $\gamma$ is the margin threshold. The MMO aims to force function $f(x_i; \theta)$ and function $f(x_j; \theta)$ to be separated by a predefined margin $\gamma$. In Cao et al. [19], a MMO is designed to constrain a pair of randomly sampled sentences with different salience scores – the one with a higher score should be larger than the other one more than a marginal threshold. Two max margin losses are proposed in Zhong et al. [188]: a margin-based triplet loss that encouraged the model to pull the golden summaries semantically closer to the original documents than to the machine-generated summaries; and a pair-wise margin loss based on a greater margin between paired candidates with more disparate ROUGE score rankings.

4.5 Multi-Task Objective

Supervision signals from MDS objectives may not be strong enough for representation learners, so some works seek other supervision signals from multiple tasks. A general form is as follows:

$$L_{Mul} = L_{Summ} + L_{Other},$$  \hspace{1cm} (5)

where $L_{Summ}$ is the loss function of MDS tasks, and $L_{Other}$ is the loss function of an auxiliary task. Angelidis et al. [5] assumed that the aspect-relevant words not only provide a reasonable basis for model aspect reconstruction, but also a good indicator for product domain. Similarly, multi-task classification was introduced by Cao et al. [19]. Two models are maintained: text classification and text summarization models. In the first model, CNN is used to classify text categories and cross-entropy loss is used as the objective function. The summarization model and the text classification model share parameters and pooling operations, so are equivalent to the shared document

ACM Comput. Surv.
vector representation. Coavoux et al. [33] jointly optimized the model from a language modeling objective and two other multi-task supervised classification losses, which are polarity loss and aspect loss.

4.6 Other Types of Objectives
There are many other types of objectives in addition to those mentioned above. Cao et al. [21] proposed using ROUGE-2 to calculate the sentence saliency scores and the model tries to estimate this saliency with linear regression. Yin et al. [174] suggested summing the squares of the prestige vectors calculated by the PageRank algorithm to identify sentence importance. Zhang et al. [184] proposed an objective function by ensembling individual scores from multiple CNN models; besides the cross-entropy loss, a consensus objective is adopted to minimize disagreement between each pair of classifiers. Amplay et al. [4] used two objectives in the abstract module: the first to optimize the generation probability distribution by maximizing the likelihood; and the second to constrain the model output to be close to its golden summary in the encoding space, as well as being distant from the random sampled negative summaries. Chu et al. [31] designed a similarity objective that shares the encoder and decoder weights within the auto-encoder module, while in the summarization module, the cosine distance indicates the similarity between the generated summary and the reviews. A variant similarity objective termed early cosine objective is further proposed to compute the similarity in a latent space which is the average of the cell states and hidden states to constrain the generated summaries semantically close to reviews.

4.7 Discussion
In MDS, cross-entropy is the most commonly adopted objective function that bridges the predicted candidate summaries and the golden summaries by treating the golden summaries as strong supervision signals. However, adopting cross-entropy loss alone may not lead the model to achieve good performance since the supervisory signal for cross-entropy objective is not strong enough by itself to effectively learn good representation. Several other objectives can thus serve as complements, e.g., reconstruction objectives offer a view from the unsupervised learning perspective; the redundancy objective constrains models from generating redundant content; while max-margin objectives require a step-change improvements from previous versions. By using multiple objectives, model optimization could be conducted with the input documents themselves if the manual annotation is scarce. The models that adopt multi-task objectives explicitly define multiple auxiliary tasks to assist the main summarization task for better generalization, and provide various constraints from different angles that lead to better model optimization.

5 EVALUATION METRICS
Evaluation metrics are used to measure the effectiveness of a given method objectively, so well-defined evaluation metrics are crucial to MDS research. We classify the existing evaluation metrics into two categories and will discuss each category in detail: (1) ROUGE: the most commonly used evaluation metrics in the summarization community; and (2) other evaluation metrics that have not been widely used in MDS research to date.

5.1 ROUGE
Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [95] is a collection of evaluation indicators that is one of the most essential metrics for many natural language processing tasks, including machine translation and text summarization. ROUGE obtains prediction/ground-truth similarity scores through comparing automatically generated summaries with a set of corresponding human-written references. ROUGE has many variants to measure candidate abstracts in a variety of ways [95]. The most commonly used ones are ROUGE-N and ROUGE-L.

**ROUGE-N** (ROUGE with n-gram co-occurrence statistics) measures a n-gram recall between reference and their corresponding candidate summaries [95]. ROUGE-N can be calculated as:

\[
\text{ROUGE-N} = \frac{\sum_{\text{Sum} \in \{\text{Ref}\}} \sum_{\text{gram}_n \in \text{Sum}} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{\text{Sum} \in \{\text{Ref}\}} \sum_{\text{gram}_n \in \text{Sum}} \text{Count}(\text{gram}_n)},
\]

where \(\text{Ref}\) and \(\text{Sum}\) are reference and machine-generated summary, \(n\) represents the length of n-gram, and \(\text{Count}_{\text{match}}(\text{gram}_n)\) represents the maximum number of n-grams in the reference summary and corresponding candidates. The numerator of ROUGE-N is the number of n-grams owned by both the reference and generated summary, while the denominator is the total number of n-grams occurring in the golden summary. The denominator could also be set to the number of candidate summary n-grams to measure precision; however, ROUGE-N mainly focuses on quantifying recall, so precision is not usually calculated. ROUGE-1 and ROUGE-2 are special cases of ROUGE-N that are usually chosen as best practices and represent the unigram and bigram.

\[
\text{ROUGE-L} (\text{ROUGE with Longest Common Subsequence})\] adopts the longest common subsequence algorithm to count the longest matching vocabularies [95]. Formally, ROUGE-L is calculated using:

\[
F_{\text{lcs}} = \frac{(1 + \beta^2) R_{\text{lcs}} P_{\text{lcs}}}{R_{\text{lcs}} + \beta^2 P_{\text{lcs}}},
\]

where

\[
R_{\text{lcs}} = \frac{\text{LCS}(\text{Ref}, \text{Sum})}{n},
\]

and

\[
P_{\text{lcs}} = \frac{\text{LCS}(\text{Ref}, \text{Sum})}{m},
\]

where LCS(\_) represents the longest common subsequence function. ROUGE-L is termed as LCS-based F-measure as it is obtained from LCS-Precision \(P_{\text{lcs}}\) and LCS-Recall \(R_{\text{lcs}}\). \(\beta\) is the balance factor between \(R_{\text{lcs}}\) and \(P_{\text{lcs}}\). It can be set by the fraction of \(P_{\text{lcs}}\) and \(R_{\text{lcs}}\); by setting \(\beta\) to a big number, only \(R_{\text{lcs}}\) is considered. The use of ROUGE-L enables the measurement of the similarity of two text sequences at sentence-level. ROUGE-L also has the advantage of deciding the n-gram without extra manual input, since the calculation of LCS can count grams adaptively.

Other ROUGE Based Metrics. ROUGE-W [95] is proposed to weight consecutive matches to better measure semantic similarities between two texts. ROUGE-S [95] stands for ROUGE with Skip-bigram co-occurrence statistics that allows the bigram to skip arbitrary words. An extension of ROUGE-S, ROUGE-SU [95] refers to ROUGE with Skip-bigram plus Unigram-based co-occurrence statistics and is able to be obtained from ROUGE-S by adding a begin-of-sentence token at the start of both references and candidates. ROUGE-WE [119] is proposed to further extend ROUGE by measuring the pair-wise summary distances in word embedding space. In recent years, more ROUGE-based evaluation models have been proposed to compare golden and machine-generated summaries, not just according to their literal similarity, but also considering semantic similarity [141, 181, 186]. In terms of the ROUGE metric for multiple golden summaries, the Jackknifing procedure (similar to K-fold validation) has been introduced [95]. The \(M\) best scores are computed from sets composed of \(M-1\) reference summaries and the final ROUGE-N is the average of \(M\) scores. This procedure can also be applied to ROUGE-L, ROUGE-W and ROUGE-S.

5.2 Other Evaluation Metrics

Besides ROUGE-based [95] metrics, other evaluation metrics for MDS exist, but have received less attention than ROUGE. We hope this section will give researchers and practitioners a holistic view of alternative evaluation metrics in this field. Based on the mode of summaries matching, we divide the evaluation metrics into two groups: lexical matching metrics and semantic matching metrics.

Lexical Matching Metrics. BLEU [123] is a commonly used vocabulary-based evaluation metric that provides a precision-based evaluation indicator, as opposed to ROUGE that mainly focuses on recall. Perplexity [72] is used to evaluate the quality of the language model by calculating the negative log probability of a word’s appearance.

ACM Comput. Surv.
Table 3. Advantages and disadvantages of different evaluation metrics.

<table>
<thead>
<tr>
<th>Evaluation Metrics</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lexical Matching Metrics</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROUGE</td>
<td>• Widely used</td>
<td>• Cannot measure texts semantically</td>
</tr>
<tr>
<td></td>
<td>• Intuitive</td>
<td>• Exact matching</td>
</tr>
<tr>
<td></td>
<td>• Easily computed</td>
<td></td>
</tr>
<tr>
<td>BLEU</td>
<td>• Intuitive</td>
<td>• Cannot measure texts semantically</td>
</tr>
<tr>
<td></td>
<td>• Easily computed</td>
<td>• Cannot deal with languages lacking word boundaries</td>
</tr>
<tr>
<td></td>
<td>• High correlations with human judgments</td>
<td></td>
</tr>
<tr>
<td>Perplexity</td>
<td>• Easily computed</td>
<td>• Sensitive to certain symbols and words</td>
</tr>
<tr>
<td></td>
<td>• Intuitive</td>
<td></td>
</tr>
<tr>
<td>Pyramid</td>
<td>• High correlations with human judgments</td>
<td>• Requires manual extraction of units</td>
</tr>
<tr>
<td></td>
<td>• Consider both content and linguistic quality</td>
<td>• Bias results easily</td>
</tr>
<tr>
<td>Responsiveness</td>
<td>• Can be calculated without reference</td>
<td>• Not widely adopted</td>
</tr>
<tr>
<td>Data Statistics</td>
<td>• Can measure the density and coverage of summary</td>
<td>• Cannot measure texts semantically</td>
</tr>
<tr>
<td><strong>Semantic Matching Metrics</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>METEOR</td>
<td>• Consider non-exact matching</td>
<td>• Sensitive to length</td>
</tr>
<tr>
<td>SUPERT</td>
<td>• Can measure texts semantic similarity</td>
<td>• Not widely adopted</td>
</tr>
<tr>
<td>Preferences based Metric</td>
<td>• Does not depend on the golden summaries</td>
<td>• Require human annotations</td>
</tr>
<tr>
<td>BERTScore</td>
<td>• Semantically measure texts to some extent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Mimic human evaluation</td>
<td>• High computational demands</td>
</tr>
<tr>
<td>MoverScore</td>
<td>• Semantically measure texts to some extent</td>
<td>• High computational demands</td>
</tr>
<tr>
<td></td>
<td>• More similar to human evaluation by adopting earth mover’s distance</td>
<td></td>
</tr>
<tr>
<td>Importance</td>
<td>• Combining redundancy, relevance and informativeness</td>
<td>• Non-trivial for implementation</td>
</tr>
<tr>
<td></td>
<td>• Theoretically supported</td>
<td></td>
</tr>
<tr>
<td>Human Evaluation</td>
<td>• Can accurately and semantically measure texts</td>
<td>• Require human annotations</td>
</tr>
</tbody>
</table>

A low perplexity on a test dataset is a strong indicator of a summary’s high grammatical quality because it measures the probability of words appearing in sequences. Based on Pyramid [117] calculation, the abstract sentences are manually divided into several Summarization Content Units (SCUs), each representing a core concept formed from a single word or phrase/sentence. After sorting SCUs in order of importance to form the Pyramid, the quality of automatic summarization is evaluated by calculating the number and importance of SCUs included in the document [118]. Intuitively, more important SCUs exist at higher levels of the pyramid. Although Pyramid shows a strong correlation with human judgment, it requires professional annotations to match and evaluate SCUs in generated and golden summaries. Some recent works focus on the construction of Pyramid [48, 62, 124, 143, 169]. Responsiveness [100] measures content selection and linguistic quality of summaries by directly rating scores. Additionally, the assessments are calculated without reference to model summaries. Data Statistics [56] contain three evaluation metrics: extractive fragment coverage measures the novelty of generated summaries by calculating the percentage of words in the summary that are also present in source documents; extractive fragment density measures the average length of the extractive block to which each word in the
summary belongs; and compression ratio compares the word numbers in the source documents and generated summary.

**Semantic Matching Metrics.** METEOR (Metric for Evaluation of Translation with Explicit Ordering) [10] is an improvement to BLEU. The main idea behind METEOR is that while candidate summaries can be correct with similar meanings, they are not exactly matched with references. In such a case, WordNet[10] is introduced to expand the synonym set, and the word form is also taken into account. SUPERT [49] is an unsupervised MDS evaluation metric that measures the semantic similarity between the pseudo-reference summary and the machine-generated summary. SUPERT obviates the need for human annotations by not referring to golden summaries. Contextualized embeddings and soft token alignment techniques are leveraged to select salient information from the input documents to evaluate summary quality. Preferences based Metric [194] is a pairwise sentence preference-based evaluation model and it does not depend on the golden summaries. The underlying premise is to ask annotators about their pair-wise preferences rather than writing complex golden summaries, and are much easier and faster to obtain than traditional reference summary-based evaluation models. BERTScore [181] computes a similarity score for each token within the candidate sentence and the reference sentence. It measures the soft overlap of two texts’ BERT embeddings. MoverScore [186] adopts a distance to evaluate the agreement between two texts in the context of BERT and ELMo word embeddings. This proposed metric has a high correlation with human judgment of text quality by adopting earth mover’s distance. Importance [129] is a simple but rigorous evaluation metric from the aspect of information theory. It is a final indicator calculated from the three aspects: Redundancy, Relevance, and Informativeness. A good summary should have low Redundancy and high Relevance and high Informativeness. The cluster of Human Evaluation is used to supplement automatic evaluation on relatively small instances. Annotators evaluate the quality of machine-generated summaries by rating Informativeness, Fluency, Conciseness, Readability, Relevance. Model ratings are usually computed by averaging the rating on all selected summary pairs.

5.3 Discussion

We summarize the advantages and disadvantages of above-mentioned evaluation metrics in Table 3. Although there are many evaluation metrics for MDS, the indicators of the ROUGE series are generally accepted by the summarization community. Almost all research works utilize ROUGE for evaluation, while other evaluation indicators are just for assistance currently. Among the ROUGE family, ROUGE-1, ROUGE-2 and ROUGE-L are the most commonly used evaluation metrics. In addition, there are plenty of existing evaluation metrics in other natural language processing tasks that could be potentially adjusted for MDS tasks, such as efficiency, effectiveness and coverage from information retrieval.

6 DATASETS

Compared to SDS tasks, large-scale MDS datasets, which contain more general scenarios with many downstream tasks, are relatively scarce. In this section, we present our investigation on the 10 most representative datasets commonly used for MDS and its variant tasks.

**DUC & TAC.** DUC[11] (Document Understanding Conference) provides official text summarization competitions each year from 2001-2007 to promote summarization research. DUC changed its name to Text Analysis Conference (TAC)[12] in 2008. Here, the DUC datasets refer to the data collected from 2001-2007; the TAC datasets refer to the datasets after 2008. Both DUC and TAC are from the news domains, including various topics such as politics, natural disasters, and biography. Nevertheless, as shown in Table 4, the DUC and TAC datasets provide small datasets for model evaluation that only include hundreds of news documents and human-annotated summaries. Of note, the first sentence in a news item is usually information-rich that renders bias in the news datasets, so it

---

10https://wordnet.princeton.edu/
11http://duc.nist.gov/
12http://www.nist.gov/tac/
Table 4. Comparison of Different Datasets. In the table, “Ave”, “Summ”, “Len”, “bus”, “rev” and “#” represent average, summary, length, business, reviews and numbers respectively; “Docs” and “sents” mean documents and sentences respectively.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Cluster #</th>
<th>Document #</th>
<th>Summ #</th>
<th>Ave Summ Len</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUC01</td>
<td>30</td>
<td>309 docs</td>
<td>60 summ</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>DUC02</td>
<td>59</td>
<td>567 docs</td>
<td>116 summ</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>DUC03</td>
<td>30</td>
<td>298 docs</td>
<td>120 summ</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>DUC04</td>
<td>50</td>
<td>10 docs / cluster</td>
<td>200 summ</td>
<td>665 bytes</td>
<td>News</td>
</tr>
<tr>
<td>DUC05</td>
<td>50</td>
<td>25-50 docs / cluster</td>
<td>140 summ</td>
<td>250 words</td>
<td>News</td>
</tr>
<tr>
<td>TAC 2008</td>
<td>48</td>
<td>10 docs / cluster</td>
<td>4 summ / cluster</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>TAC 2009</td>
<td>44</td>
<td>10 docs / cluster</td>
<td>4 summ / cluster</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>TAC 2010</td>
<td>46</td>
<td>10 docs / cluster</td>
<td>4 summ / cluster</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>TAC 2011</td>
<td>44</td>
<td>10 docs / cluster</td>
<td>4 summ / cluster</td>
<td>100 words</td>
<td>News</td>
</tr>
<tr>
<td>OPOSUM</td>
<td>60</td>
<td>600 rev</td>
<td>1 summ / cluster</td>
<td>100 words</td>
<td>Amazon reviews</td>
</tr>
<tr>
<td>WikiSum</td>
<td>-</td>
<td>train / val / test 1579360 / 38144 / 38205</td>
<td>1 summ / cluster</td>
<td>139.4 tokens / summ</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>Multi-News</td>
<td>-</td>
<td>train / val / test 44972 / 5622 / 5622</td>
<td>1 summ / cluster</td>
<td>263.66 words / summ</td>
<td>News</td>
</tr>
<tr>
<td>Opinosis</td>
<td>51</td>
<td>6457 rev</td>
<td>5 summ / cluster</td>
<td>-</td>
<td>Site reviews</td>
</tr>
<tr>
<td>Rotten Tomatoes</td>
<td>3731</td>
<td>99.8 rev / cluster</td>
<td>1 summ / cluster</td>
<td>19.6 tokens / summ</td>
<td>Movie reviews</td>
</tr>
<tr>
<td>Yelp</td>
<td>-</td>
<td>train / val / test bus: 10695 / 1337 / 1337 rev: 1038184 / 129856 / 129840</td>
<td>-</td>
<td>-</td>
<td>Customer reviews</td>
</tr>
<tr>
<td>Scisumm</td>
<td>1000</td>
<td>21 - 928 cites / paper 15 sents / refer</td>
<td>1 summ / cluster</td>
<td>151 words</td>
<td>Science Paper</td>
</tr>
<tr>
<td>WCEP</td>
<td>10200</td>
<td>235 docs / cluster</td>
<td>1 summ / cluster</td>
<td>32 words</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>Multi-XScience</td>
<td>-</td>
<td>train / val / test 30369/ 5066 / 5093</td>
<td>1 summ / cluster</td>
<td>116.44 words / summ</td>
<td>Science Paper</td>
</tr>
</tbody>
</table>

fails to reflect the structure of natural documents in daily lives. These two datasets are on a relatively small scale and not ideal for large-scale deep neural based MDS model training and evaluation.

**OPOSUM.** OPOSUM [5] collects multiple reviews of six product domains from Amazon. This dataset not only contains multiple reviews and corresponding summaries but also products' domain and polarity information. The latter information could be used as auxiliary supervision signals.

**WikiSum.** WikiSum [97] targets abstractive MDS. For a specific Wikipedia theme, the documents cited in Wikipedia articles or the top-10 Google search results (using the Wikipedia theme as a query) are seen as the source documents. Golden summaries are the real Wikipedia articles. However, some of the URLs are not available and can be identical to each other in parts. To remedy these problems, Liu et al. [98] cleaned the dataset and deleted duplicated examples, so here we report statistical results from [98].

**Multi-News.** Multi-News [44] is a relatively large-scale dataset in the news domain; the articles and human-written summaries are all from the Web\(^\text{13}\). This dataset includes 56,216 article-summary pairs and contains trace-back links to the original documents. Moreover, the authors compared the Multi-News dataset with prior datasets in terms of coverage, density, and compression, revealing that this dataset has various arrangement styles of sequences.

\(^{13}\)http://newser.com

ACM Comput. Surv.
Opinosis. The Opinosis dataset [47] contains reviews of 51 topic clusters collected from TripAdvisor\textsuperscript{14}, Amazon\textsuperscript{15}, and Edmunds\textsuperscript{16}. For each topic, approximately 100 sentences on average are provided and the reviews are fetched from different sources. For each cluster, five professionally written golden summaries are provided for model training and evaluation.

Rotten Tomatoes. The Rotten Tomatoes dataset [162] consists of the collected reviews of 3,731 movies from the Rotten Tomato website\textsuperscript{17}. The reviews contain both professional critics and user comments. For each movie, a one-sentence summary is created by professional editors.

Yelp. Chu et al. [31] proposed a dataset named Yelp based on the Yelp Dataset Challenge. This dataset includes multiple customer reviews with five-star ratings. The authors provided 100 manual-written summaries for model evaluation using Amazon Mechanical Turk (AMT), within which every eight input reviews are summarized into one golden summary.

Scisumm. Scisumm dataset [172] is a large, manually annotated corpus for scientific document summarization. The input documents are a scientific publication, called the reference paper, and multiple sentences from the literature that cite this reference paper. In the SciSumm dataset, the 1,000 most cited papers from the ACL Anthology Network [132] are treated as reference papers, and an average of 15 citation sentences are provided after cleaning. For each cluster, one golden summary is created by five NLP-based Ph.D. students or equivalent professionals.

WCEP. The Wikipedia Current Events Portal dataset (WCEP) [51] contains human-written summaries of recent news events. Similar articles are provided by searching similar articles from Common Crawl News dataset\textsuperscript{18} to extend the inputs to obtain large-scale news articles. Overall, the WCEP dataset has good alignment with real-world industrial use cases.

Multi-XScience. The source data of Multi-XScience [101] are from Arxiv and Microsoft academic graphs and this dataset is suitable for abstractive MDS. Multi-XScience contains fewer positional and extractive biases than the WikiSum and Multi-News datasets, so the drawback of obtaining higher scores from a copy sentence at a certain position can be partially avoided.

Datasets for MDS Variants. The representative query-oriented MDS datasets are Debatepedia [115], AQUAMUSE [82], and QBSUM [185]. The representative dialogue summarization datasets are DIALOGSUM [26], AMI [24], MEDIASUM [191], and QMSum [189]. RTS is a track at the Text Retrieval Conference (TREC) which provides several RTS datasets\textsuperscript{19}. Tweet Contextualization track [13] (2012-2014) is derived from the INEX 2011 Question Answering Track, that focuses on more NLP-oriented tasks and moves to MDS.

Discussion. Table 4 compares 20 MDS datasets based on the numbers of clusters and documents; the number and the average length of summaries; and the field to which the dataset belongs. Currently, the main areas covered by the MDS datasets are news (60%), scientific papers (10%) and Wikipedia (10%). In the early development of the MDS, most studies were performed on the DUC and TAC datasets. However, the size of these datasets is relatively small, and thus not highly suitable for training deep neural network models. Datasets on news articles are also common, but the structure of news articles (highly compressed information in the first paragraph or first sentence of each paragraph) can cause positional and extractive biases during training. In recent years, large-scale datasets such as WikiSum and Multi-News datasets have been developed and used by researchers to meet training requirements, reflecting the rising trend of data-driven approaches.

\textsuperscript{14}https://www.tripadvisor.com/
\textsuperscript{15}https://www.amazon.com.au/
\textsuperscript{16}https://www.edmunds.com/
\textsuperscript{17}http://rottentomatoes.com
\textsuperscript{18}https://commoncrawlr.org/2016/10/news-dataset-available/
\textsuperscript{19}http://trecrts.github.io/
7 FUTURE RESEARCH DIRECTIONS AND OPEN ISSUES

Although existing works have established a solid foundation for MDS it is a relatively understudied field compared with SDS and other NLP topics. Summarizing on multi-modal data, medical records, codes, project activities and MDS combining with Internet of Things [183] have still received less attention. Actually, MDS techniques are beneficial for a variety of practical applications, including generating Wikipedia articles, summarizing news, scientific papers, and product reviews, and individuals, industries have a huge demand for compressing multiple related documents into high-quality summaries. This section outlines several prospective research directions and open issues that we believe are critical to resolving in order to advance the field.

7.1 Capturing Cross-document Relations for MDS

Currently, many MDS models still center on a simple concatenation of input documents into a flat sequence, ignoring cross-document relations. Unlike SDS, MDS input documents may contain redundant, complementary, or contradictory information [130]. Discovering cross-document relations, which can assist models to extract salient information, improve the coherence and reduce redundancy of summaries [94]. Research on capturing cross-document relations has begun to gain momentum in the past two years; one of the most widely studied topics is graphical models, which can easily be combined with deep learning based models such as graph neural networks and Transformer models. Several existing works indicate the efficacy of graph-based deep learning models in capturing semantic-rich and syntactic-rich representation and generating high-quality summaries [94, 160, 172, 173]. To this end, a promising and important direction would be to design a better mechanism to introduce different graph structures [30] or linguistic knowledge [15, 103], possibly into the attention mechanism in deep learning based models, to capture cross-document relations and to facilitate summarization.

7.2 Creating More High-quality Datasets for MDS

Benchmark datasets allow researchers to train, evaluate and compare the capabilities of different models at the same stage. High-quality datasets are critical to developing MDS tasks. DUC and TAC, the most common datasets used for MDS tasks, have a relatively small number of samples so are not very suitable for training DNN models. In recent years, some large datasets have been proposed, including WikiSum [97], Multi-News [44], and WCEP [51], but more efforts are still needed. Datasets with documents of rich diversity, with minimal positional and extractive biases are desperately required to promote and accelerate MDS research, as are datasets for other applications such as summarization of medical records or dialogue [111], email [154, 176], code [106, 135], software project activities [3], legal documents [76], and multi-modal data [89]. The development of large-scale cross-task datasets will facilitate multi-task learning [165]. However, the datasets of MDS combining with text classification, question answering, or other language tasks have seldom been proposed in the MDS research community, but these datasets are essential and widely employed in industrial applications.

7.3 Improving Evaluation Metrics for MDS

To our best knowledge, there are no evaluation metrics specifically designed for MDS models – SDS and MDS models share the same evaluation metrics. New MDS evaluation metrics should be able to: (1) evaluating the relations between the different input documents in the generated summary; (2) measuring to what extent the redundancy in input documents is reduced; and (3) judging whether the contradictory information across documents is reasonably handled. A good evaluation indicator is able to reflect the true performance of an MDS model and guide design of improved models. However, current evaluation metrics [43] still have several obvious defects. For example, despite the effectiveness of commonly used ROUGE metrics, they struggle to accurately measure the semantic similarity between a golden and generated summary because ROUGE-based evaluation metrics only consider vocabulary-level distances; as such, even if a ROUGE score improves, it does not necessarily mean that the summary is of a higher quality and so is not ideal for model training. Recently, some works extend ROUGE along with WordNet [141] or pre-trained LMs [181] to alleviate these drawbacks. It is challenging to propose evaluation indicators that can reflect the true quality of generated summaries comprehensively and
as semantically as human raters. Another frontline challenge for evaluation metrics research is unsupervised evaluation, being explored by a number of recent studies [49, 148].

7.4 Reinforcement Learning for MDS
Reinforcement learning [110] is a cluster of algorithms based on dynamic programming according to the Bellman Equation to deal with sequential decision problems, where state transition dynamics of the environment are provided in advance. Several existing works [113, 127, 171] model the document summarization task as a sequential decision problem and adopt reinforcement learning to tackle the task. Although deep reinforcement learning for SDS has made great progress, we still face challenges to adapt existing SDS models to MDS, as the latter suffers from a large state, action space, and problems with high redundancy and contradiction [105]. Additionally, current summarization methods are based on model-free reinforcement learning algorithms, in which the model is not aware of environment dynamics but continuously explores the environment through simple trial-and-error strategies, so they inevitably suffer from low sampling efficiencies. Nevertheless, the model-based approaches can leverage data more efficiently since they update models upon the prior to the environment. In this case, data-efficient reinforcement learning for MDS could potentially be explored in the future.

7.5 Pre-trained Language Models for MDS
In many NLP tasks, the limited labeled corpora are not adequate to train semantic-rich word vectors. Using large-scale, unlabeled, task-agnostic corpora for pre-training can enhance the generalization ability of models and accelerate convergence of networks [109, 128]. At present, pre-trained LMs have led to successes in many deep learning based NLP tasks. Among the reviewed papers [2, 85, 94, 147, 188], multiple works adopt pre-trained LMs for MDS and achieve promising improvements. Applying pre-trained LMs such as BERT [35], GPT-2 [133], GPT-3 [18], XLNet [170], ALBERT [84], or T5 [134], and fine-tuning them on a variety of downstream tasks allows the model to achieve faster convergence speed and can improve model performance. MDS requires the model to have a strong ability to process long sequences. It is promising to explore powerful LMs specifically targeting long sequence input characteristics and avoiding quadratic memory growth for self-attention mechanism, such as Longformer [14], REFORMER [79], or Big Bird [175] with pre-trained models. Also, tailor-designed pre-trained LMs for summarization have not been well-explored, e.g., using gap sentences generation is more suitable than using masked language model [178]. Most MDS methods focus on combining pre-trained LMs in encoder and, as for capturing cross-document relations, applying them in decoder is also a worthwhile direction for research [126]. Other promising directions in this area involve exploring pre-trained LMs in languages other than English and specialized LMs for dealing with specific summarization tasks, e.g. LMs pre-trained on scientific articles.

7.6 Creating Explainable Deep Learning Model for MDS
Researchers are more focused on designing deep architectures towards a certain MDS task by improving the models performance while ignoring their interpretabilities. However, an explainable model can reveal how it generates candidate summaries – to distinguish whether the model has learned the distribution of generating condensed and coherent summaries from multiple documents without bias – and is thus crucial for model building. Recently, a large number of researches into explainable models [136, 179] have proposed easing the non-interpretable concern of deep neural networks, within which model attention plays an especially important role in model interpretation [140, 190]. While explainable methods have been intensively researched in NLP [68, 83], studies into explainable MDS models are relatively scarce and would benefit from future development.

7.7 Adversarial Attack and Defense for MDS
Adversarial examples are strategically modified samples that aim to fool deep neural networks based models. An adversarial example is created via the worst-case perturbation of the input to which a robust DNN model would still assign correct labels, while a vulnerable DNN model would have high confidence in the wrong prediction. The idea of using adversarial examples to examine the robustness of a DNN model originated from research in...

ACM Comput. Surv.
Computer Vision [149] and was introduced in NLP by Jia et al. [73]. An essential purpose for generating adversarial examples for neural networks is to utilize these adversarial examples to enhance the model’s robustness [53]. Therefore, research on adversarial examples not only helps identify and apply a robust model but also helps to build robust models for different tasks. Following the pioneering work proposed by Jia et al. [73], many attack methods have been proposed to address this problem in NLP applications [182] with limited research for MDS [28]. It is worth filling this gap by exploring existing and developing new, adversarial attacks on the state-of-the-art DNN-based MDS models.

7.8 Multi-modality for MDS

Existing multi-modal summarization is based on non-deep learning techniques [69–71, 90], leaving a huge opportunity to exploit deep learning techniques for this task. Multi-modal learning has led to successes in many deep learning tasks, such as Visual Language Navigation [161] and Visual Question Answering [7]. Combining MDS with multi-modality has a range of applications:

- text + image: generating summaries with pictures and texts for documents with pictures. This kind of multi-modal summary can improve the satisfaction of users [193];
- text + video: based on the video and its subtitles, generating a concise text summary that describes the main context of video [121]. Movie synopsis is one application;
- text + audio: generating short summaries of audio files that people could quickly preview without actually listening to the entire audio recording [42].

Deep learning is well-suited for multi-modal tasks [58], as it is able to effectively capture highly nonlinear relationships between images, text or video data. Existing MDS models target at dealing with textual data only. Involving richer modalities based on textual data requires models to embrace larger capacity to handle these multi-modal data. The big models such as UNITER [27], VisualBERT [91] deserve more attention in multi-modality MDS tasks. However, at present, there is little multi-modal research work based on MDS; this is a promising, but largely under-explored, area where more studies are expected.

8 CONCLUSION

In this article, we have presented the first comprehensive review of the most notable works to date on deep learning based multi-document summarization (MDS). We propose a taxonomy for organizing and clustering existing publications and devise the network design strategies based on the state-of-the-art methods. We also provide an overview of the existing multi-document objective functions, evaluation metrics and datasets, and discuss some of the most pressing open problems and promising future extensions in MDS research. We hope this survey provides readers with a comprehensive understanding of the key aspects of MDS tasks, clarifies the most notable advances, and sheds light on future studies.

REFERENCES


ACM Comput. Surv.


Haili Tan, Ziyu Lu, and Wenjie Li. 2017. Neural Network based Reinforcement Learning for Real-time Pushing on Text Stream. In


ACM Comput. Surv.


