Hierarchical Human-Like Deep Neural Networks for Abstractive Text Summarization

Min Yang, Chengming Li, *Member, IEEE*, Ying Shen, *Member, IEEE*, Qingyao Wu, *Member, IEEE*, Zhou Zhao, *Member, IEEE*, and Xiaojun Chen, *Member, IEEE*

***Abstract*— Developing an abstractive text summarization (ATS) system that is capable of generating concise, appropriate, and plausible summaries for the source documents is a long-term goal of artificial intelligence (AI). Recent advances in ATS are overwhelmingly contributed by deep learning techniques, which have taken the state-of-the-art of ATS to a new level. Despite the significant success of previous methods, generating high-quality and human-like abstractive summaries remains a challenge in practice. The human reading cognition, which is essential for reading comprehension and logical thinking, is still relatively new territory and underexplored in deep neural networks. In this article, we propose a novel Hierarchical Human-like deep neural network for ATS (HH-ATS), inspired by the process of how humans comprehend an article and write the corresponding summary. Specifically, HH-ATS is composed of three primary components (i.e., a knowledge-aware hierarchical attention module, a multitask learning module, and a dual discriminator generative adversarial network), which mimic the three stages of human reading cognition (i.e., rough reading, active reading, and postediting). Experimental results on two benchmark data sets (CNN/Daily Mail and Gigaword) demonstrate that HH-ATS consistently and substantially outperforms the compared methods.**

***Index Terms*— Abstractive text summarization (ATS), external knowledge base (KB), generative adversarial network, human reading cognition, multitask learning.**

I. Introduction

ABSTRACTIVE text summarization (ATS) aims at generating condensed and appropriate summaries that keep the salient information and overall meaning of the source documents. Different from extractive text summarization, which extracts the most salient phrases and sentences from the source documents, abstractive summaries potentially consist of new words and phrases that do not appear in the source documents. In recent years, ATS has attracted considerable attention due to its wide applications in information retrieval (IR) and natural language processing (NLP).

Recent advances in deep learning-based approaches (i.e., the sequence-to-sequence (seq2seq) models [1], [2]) have taken the state-of-the-art of ATS to a new level. The general idea behind these methods is to encode the source documents as distributed representations using the long short-term memory (LSTM) network [3] and then use another LSTM decoder to produce the corresponding summaries. The seq2seq framework has become the mainstream approach for ATS mainly because it enables one to train an ATS system in an end-to-end way with the potential to learn the semantic alignments between the source documents and the corresponding summaries.

Although great efforts have been made to build effective ATS models, generating human-like and high-quality abstractive summaries is still a challenge since the computers struggle to understand the meanings of the input document and do not have the language capability of writing a summary highlighting its main points. Despite the usefulness of the human reading ability, so far, the principles of the human reading cognition are underutilized in ATS (i.e., how humans summarize an article).

When humans read, comprehend, and summarize a source article, a three-stage human reading cognition process would be explored, including the rough reading, active reading (i.e., reading comprehension), and postediting [4]–[6]. The human reading cognition process can be depicted as an ongoing process as humans proceed through the document, resulting in a gradually emerging understanding and summarization of the input document.

In the rough reading phase, humans set the reading purpose and skim a document with commonsense knowledge to get a general understanding of its meaning, forming the prereading cognition. As revealed by previous work [7], prior knowledge[1] facilitates and enhances human reading, which would have a significant influence on text summarization since it helps the readers to construct a coherent mental representation of

Manuscript received February 11, 2019; revised January 6, 2020 and April 30, 2020; accepted June 28, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61906185, in part by the Natural Science Foundation of Guangdong Province of China under Grant 2019A1515011705 and Grant 2018A030313943, and in part by the Shenzhen Basic Research Foundation under Grant JCYJ20180302145607677, Grant JCYJ20180302145645821, and Grant JCYJ20180302145633177. *(Corresponding authors: Ying Shen; Qingyao Wu.)*

Min Yang and Chengming Li are with the Shenzhen Key Laboratory for High Performance Data Mining, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: min.yang@siat.ac.cn; cm.li@siat.ac.cn).

Ying Shen is with the School of Intelligent Engineering, Sun Yat-sen University, Guangzhou 510275, China (e-mail: shenying76@mail.sysu.edu.cn).

Qingyao Wu is with the School of Software Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: qyw@scut.edu.cn).

Zhou Zhao is with the College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: zhaozhou@zju.edu.cn).

Xiaojun Chen is with the College of Computer Science and Software, Shenzhen University, Shenzhen 518060, China (e-mail: xjchen@szu.edu.cn).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3008037

[1]Here, prior knowledge is also known as the commonsense knowledge of human beings, which is available before conducting a specific task.

the document and obtain the sketched information of the document.

Active reading is an essential part of task-specific reading comprehension. It ensures an in-depth understanding of the input document and chooses salient information to form the summary. To refine the main idea of the input document, humans need to go beyond its literal meaning and capture the essential features of the document. The reading comprehension process is what makes humans move from the coarse-grained literal information to the fine-grained salient information that combines the meanings of multiple sentences to form the abstractive summary [7]. During the active reading process, humans are required to make a series of inferences that can facilitate generating high-quality summaries, such as pointing out the document category and locating the salient information of a document.

As more essential fine-grained information becomes available, humans can check whether there is sufficient salient information to write a high-quality summary and further refine the generated summary when necessary. This postediting (polishing) phase creates opportunities for the readers to understand the source document profoundly and ensure that the summarization goal has been met via error correction. For example, in practice, a generated summary usually needs to be edited for accuracy and fluency by adding new words and deleting and rephrasing the generated words when necessary. In brief, if we desire to develop an ATS model possessing the reading comprehension and language generation ability for ATS, studying these hierarchical stages of the human reading cognition process is quite necessary.

In this study, we design a novel Hierarchical Human-like deep neural network for ATS (HH-ATS), which extends the existing seq2seq neural network architecture by mimicking the process of how humans write a summary for a piece of text. Similar to the previous state-of-the-art methods [8], [9], the seq2seq framework is used as the backbone of our summarization system. In addition, HH-ATS contains three hierarchical components that are consistent with the three-stage human reading cognition process. First, we propose a knowledge-aware hierarchical attention module to simulate the rough reading, which leverages the external knowledge from knowledge base (KB) as prior knowledge to distinguish the essential features from the source document and decide the focus of the summary. Intuitively, KB provides comprehensive information about entities and relations between them and can highlight the features that are essential to write a summary. Second, to simulate the process of active reading and deeply understand a text, multitask learning is proposed to optimize the ATS task and two auxiliary tasks (i.e., text categorization and syntax annotation) simultaneously. Specifically, text categorization helps to learn a category-specific document encoder for effectively detecting the most salient information from the source document. Syntax annotation helps to mitigate the problem of generating a grammatically incorrect summary by exploiting word-level syntax knowledge. Third, we employ a dual discriminator generative adversarial network (GAN) to further improve the performance of our model by using a binary discriminator and a ranking-based discriminator to guide the generator in an adversarial process. This adversarial training can eventually make G generate more plausible and precise summaries.

The main contributions of this article are shown as follows.

1) We propose a novel hierarchical human-like deep neural network to improve the performance of ATS, which extends the seq2seq framework by mimicking the process of how humans write a summary for a piece of text.
2) We propose a knowledge-aware hierarchical attention module, which leverages the commonsense knowledge from KB as prior knowledge to capture important information from source documents at different levels of granularity.
3) We propose a multitask learning method to optimize the ATS task and two auxiliary tasks (text categorization and syntax annotation) simultaneously. Text categorization helps to detect the most salient information from source documents, and syntax annotation helps to mitigate the problem of generating grammatically incorrect summaries.
4) We propose an interactive key-value memory-augmented attention mechanism (IKVMA) to better keep track of attention history and salient information coverage, helping the decoder to overcome the problem of generating repetitive and incomplete summaries by automatically distinguishing which salient facets have been described and which salient facets are unexplored.
5) We conduct a series of experiments on two benchmark data sets to verify the effectiveness of HH-ATS. Experimental results show that HH-ATS obtains substantially better results than the strong baselines on both data sets.

The rest of this article is structured as follows. Section II reviews and discusses the related references. Section III provides the problem definition and the architecture of HH-ATS. In Sections IV–VI, we introduce the knowledge-aware hierarchical attention module, the multitask learning module, and the dual discriminator GAN (DD-GAN) module in detail. Section VII describes the experimental setup. Section VIII demonstrates and analyzes the experimental results. Section IV concludes this article.

II. Related Work

A. Abstractive Text Summarization

Broadly, existing text summarization studies can be divided into extractive and abstractive methods. The goal of extractive summarization is to produce a condensed summary by extracting important phrases or sentences from the source document. In contrast, abstractive summarization attempts to generate a novel summary using text generation techniques [1]. The generated summary may contain new words or phrases that are not contained in the source document. The focus of this manuscript is ATS.

So far, great efforts have been made to develop ATS by applying the seq2seq framework [1], [8]. Rush *et al.* [1] was the first work that employed a seq2seq model with an attention mechanism for ATS. Nallapati *et al.* [2] applied an

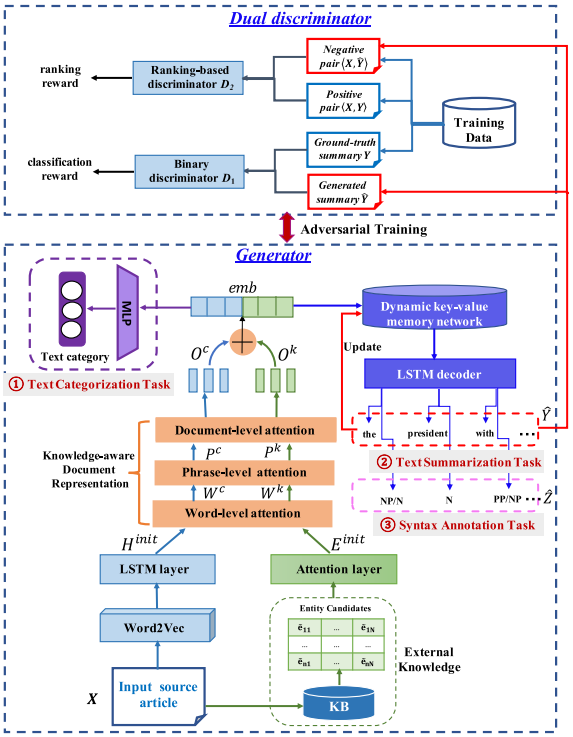

Fig. 1. Overall architecture of the proposed HH-ATS model.

attentional seq2seq model with the "large vocabulary trick" to learn hierarchical document representations and identify essential information from source documents. See *et al.* [8] proposed a "pointer-generator" model to generate summaries. At each decoding step, the model either copied a word from the source article via pointing or generated a word from the whole vocabulary.

Subsequently, several works were trying to combine the encoder–decoder LSTMs and the reinforcement learning techniques to improve further the performance of abstractive summarization [9], [10]. For instance, Paulus *et al.* [9] introduced a deep reinforced model (ML+RL version), in which a new loss function was proposed to integrate the maximum likelihood estimation (MLE) objective and the policy gradient algorithm to mitigate the exposure bias issue. Keneshloo *et al.* [11] investigated the performance of several reinforcement learning techniques (i.e., policy gradient, actor-critic, and deep Q-learning) in training the seq2seq models for a wide variety of text generation tasks, such as ATS and image captioning.

B. Multitask Learning in NLP

Multitask learning optimizes multiple tasks simultaneously, aiming to improve the generalization performance of each task [12]–[14]. For instance, Liu *et al.* [15] combined the tasks of information retrieval and multidomain classification, which could eventually benefit from the regularization effects and a large scale of cross-task data. Luong *et al.* [16] integrated the multitask learning into the seq2seq framework, sharing the parameters of both encoder and decoder across the tasks. Noticeable improvements have been achieved on the machine translation task. Liu *et al.* [17] trained some classification tasks (e.g., subjective classification and sentiment classification) simultaneously, which shared the text representation learning module. Yang *et al.* [18] presented a multitask learning algorithm for web-scale event summarization, which simultaneously optimized relevance prediction and real-time document filtering to improve text relevance. Yang *et al.* [19] employed a multitask learning method to enhance the ATS task, which trained ATS and two coupled tasks simultaneously in a unified framework. Yang *et al.* [20] developed a personalized dialog generation system, which leveraged author profiling as an auxiliary task to learn better user-aware dialog representation and capture the characteristics of users.

C. Generative Adversarial Network in Text Generation

In recent years, several works have revealed the effectiveness of GAN [21] in different text generation tasks, such as dialog generation [22], image captioning [23], and sequence generation [24]. GAN attempts to learn a generator and a discriminator alternately, which are trained with an adversarial minimax game. Eventually, the discriminator can guide the generator to produce more plausible sentences. For instance, Yu *et al.* [24] introduced a SeqGAN method to produce short sequences. The discriminator evaluated the sequence and provided feedback (reward) to guide the learning of the generator. To solve the problem that the gradient cannot pass back to the generator when the output is discrete, the policy gradient algorithm was applied to optimize the generator. Chen *et al.* [23] employed the adversarial learning to exploit the unpaired images and captions for effective image captioning. Two critic networks were proposed to evaluate the generated captions, where a domain critic measured whether the generated captions were indistinguishable from the human-written captions and a multimodal critic measured whether an image-caption pair was a valid one. Liu *et al.* [10] applied the standard GAN framework to enhance the ATS system by employing a binary text classifier as the discriminator to guide the training of the generator through an adversarial process. Yang *et al.* [25] proposed a hybrid learning model for abstractive summarization, which simulated the different stages of how humans write a summary. A standard GAN framework was used in the polishing stage.

This article is a significant extension of our previous conference article [25]. The primary differences between HH-ATS and the previous studies (including our preliminary work [25]) can be summarized as three aspects.

1) HH-ATS proposes a knowledge-aware hierarchical attention module, which leverages the external background knowledge from KB as prior knowledge to capture the important information from source documents at different levels of granularity.
2) An IKVMA is proposed to keep track of attention history and salient information coverage, helping the decoder to overcome the problems of generating repetitive and faultiness summaries by automatically distinguishing which salient facets have been described and which salient facets are unexplored.

3) We propose a DD-GAN framework to further improve the performance of text summarization. A binary discriminator and a ranking-based discriminator are devised to provide guidance to the generator, which assesses both the genuineness of a candidate summary and the relevance to its corresponding source article.

III. Our Methodology

A. Problem Definition

Assume that each input document $X = \{x_1, x_2, \ldots, x_n\}$ has a corresponding gold summary $Y = \{y_1, y_2, \ldots, y_g\}$ and a category label C, where n and g indicate the lengths of the source document and the gold summary, respectively. Each gold summary Y has a sequence of combinatory category grammar (CCG) supertags [26], denoted as $Z = \{z_1, z_2, \ldots, z_g\}$. Given the source article X, our HH-ATS model attempts to produce a target summary $\hat{Y} = \{\hat{y_1}, \hat{y_2}, \ldots, \hat{y_m}\}$, where m denotes the length of the target summary. For the text categorization task, given the source document X, the objective of HH-ATS is to predict a category label $\hat{C}$ for the input article. For syntax annotation task, given the source document X, the objective of HH-ATS is to output a sequence of CCG supertags $\hat{Z} = \{\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_m\}$ in response to the target summary.

B. Architecture of Our Approach

As stated in Section I, the human reading cognitive process is composed of three primary phases: rough reading, active reading, and postediting. Accordingly, we introduce a hierarchical human-like deep neural network to mimic the process of human reading cognition. As shown in Fig. 1, HH-ATS contains three key parts: a knowledge-aware hierarchical attention module, a multitask learning module, and a DD-GAN module. First, the knowledge-attention module prereads the input document and leverages the commonsense knowledge from KB as attention source to distinguish the essential features of the input document and produce a knowledge-aware document representation (see Section IV). Second, after initial perception (or rough reading), we propose two extended regularizations (text categorization and syntax annotation) for the seq2seq framework using multitask learning (see Section V). Third, we employ a DD-GAN framework to further improve the performance of the ATS system, which trains a generator and two discriminators with a minimax game (see Section VI). Next, we will introduce each component of HH-ATS in detail.

IV. Knowledge-Aware Hierarchical Attention Module

The knowledge-aware hierarchical attention module leverages the commonsense knowledge from KB as prior knowledge to learn knowledge-aware document encoder in our sequence-to-sequence framework, capturing the overall meaning of the source document. In particular, the semantic compositionality of the source document is fully exploited with word-level attention, phrase-level attention, and document-level attention. To prevent conceptual confusion, we utilize superscripts "c" and "k" to represent the variables that are related to context representation and knowledge representation, respectively.

A. Initial Context Representation

We first compute the initial context representation of the source document X. Each word x in document X is converted into a distributed representation $\mathbf{x} \in \mathbb{R}^{d_e}$ by an embedding layer, where d_e represents the dimension of the word embedding. Then, the hidden states of words in the source document are learned by an LSTM layer. Formally, given the word vector $\mathbf{x}_i$ in the source article, the hidden state $\mathbf{h}_i \in \mathbb{R}^{d_c}$ (d_c represents the size of each LSTM unit) is calculated as

$$\mathbf{h}_i = \text{LSTM}(\mathbf{h}_{i-1}, \mathbf{x}_i) \tag{1}$$

where $\mathbf{h}_{i-1}$ is the hidden state at index $i-1$.

Thus, given the input document X, we can obtain the initial contextual document representation $H^{\text{init}} = [\mathbf{h}_1, \ldots, \mathbf{h}_n]$, where n indicates the length of the source document X.

B. Initial Knowledge Representation

Knowledge-based document representation is learned from the candidate entity embeddings under the guidance of contextual information. We conduct entity mention detection by performing n-gram matching and obtain top-N candidate entities from KB for each entity mention in the document due to the ambiguity of each entity mention, e.g., "Apple" may refer to a company or a kind of fruit. We apply the DeepWalk [27] algorithm to learn the distributed representations for the entities in KB. Formally, we denote the candidate entities for the entity mention at index i as $\{\tilde{\mathbf{e}}_{i1}, \tilde{\mathbf{e}}_{i2}, \ldots, \tilde{\mathbf{e}}_{iN}\} \in \mathbb{R}^{N \times d_{\text{kb}}}$, where d_{kb} is the dimension of the entity embedding in KB and $\tilde{\mathbf{e}}_{ij}$ is the jth candidate entity for the ith word.

The context information can facilitate the retrieval of the candidate entities. For example, when observing the context words "Steve Jobs," we can infer that the ambiguous entity "Apple" refers to a company instead of a kind of fruit. In order to retrieve the most appropriate candidate entity from $\{\tilde{\mathbf{e}}_{i1}, \tilde{\mathbf{e}}_{i2}, \ldots, \tilde{\mathbf{e}}_{iN}\}$, we use the initial contextual document representation H^{init} as the attention source to calculate the attention weight α_{ij} for the jth candidate entity embedding $\tilde{\mathbf{e}}_{ij}$

$$\alpha_{ij} = \text{softmax}(\rho(\tilde{\mathbf{e}}_{ij}, \mu(H^{\text{init}}))) \tag{2}$$

$$\rho(\tilde{\mathbf{e}}_{ij}, \mu(H^{\text{init}})) = \tanh(U^{k1}\tilde{\mathbf{e}}_{ij} + U^{c1}\mu(H^{\text{init}}) + \mathbf{b}^{k1}) \tag{3}$$

where μ represents averaging operation, U^{c1} and U^{k1} indicate weight parameters, $\mathbf{b}^{k1}$ represents a bias term, and α_{ij} indicates the context-guided attention weight for the jth candidate entity embedding $\tilde{\mathbf{e}}_{ij}$.

After computing the attention weight for each candidate entity embedding, we can learn the knowledge representation for each entity mention by congregating the embeddings of its candidate entities in the KB with attention vector α_i

$$\mathbf{e}_i = \sum_{i=1}^{N} \alpha_{ij}\tilde{\mathbf{e}}_{ij} \tag{4}$$

where $\mathbf{e}_i$ is the knowledge representation for the ith entity in the source article and N represents the number of candidate entities for each entity mention in the source article. In this way, we learn the initial knowledge representation of the input article as $E^{\text{init}} = [\mathbf{e}_1, \ldots, \mathbf{e}_n]$.

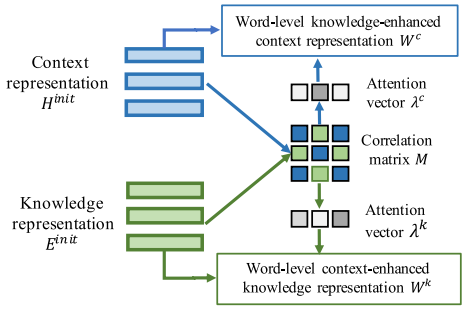

Fig. 2. Dataflow of word-level mutual attention for the source article X.

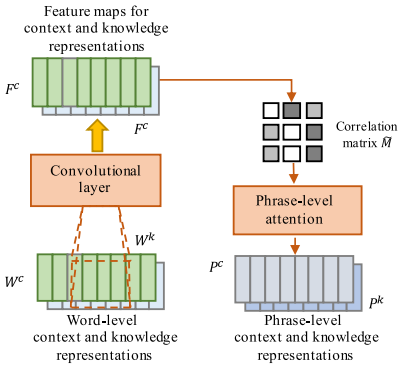

Fig. 3. Dataflow of phrase-level attention for the source article X.

C. Knowledge-Aware Document Representation

We design a hierarchical attention network to distill the crucial features from the context and knowledge representations by exploring the semantic compositionality of the source article. Specifically, we design a three-stage attention network, including word-level attention, phrase-level attention, and document-level attention to model the document semantics at three different levels, exploring the semantic compositionality of the source article and capturing more comprehensive knowledge-enhanced information.

1) Word-Level Mutual Attention: After obtaining the entity and word embeddings, a word-level mutual attention mechanism is introduced to learn the correlation between context and knowledge representations, as shown in Fig. 2. It benefits to learn a subspace to measure the correlation between the entity words and context words. Formally, we adopt the dot product between the context and knowledge representations to calculate the correlation matrix M for the input document X as

$$M = (H^{\text{init}})^T \cdot E^{\text{init}} \in \mathbb{R}^{n \times n} \tag{5}$$

where each element in $M_{i,j}$ refers to the correlation between the ith element in the context representation and the jth element in the knowledge representation.

Not all words contribute equally to the overall document representation. Thus, we employ word-level attention to identify the words that are essential to the meaning of the document. In particular, we average the values of each row and each column of M as attention sources and get the normalized weight vectors $\boldsymbol{\lambda}^c$ and $\boldsymbol{\lambda}^k$ to measure the importance of items in context and knowledge representations through the softmax function

$$\boldsymbol{\lambda}^c = \text{softmax}\left(\frac{\sum_n^{i=1} M[:, i]}{n}\right) \tag{6}$$

$$\boldsymbol{\lambda}^k = \text{softmax}\left(\frac{\sum_n^{j=1} M[j, :]}{n}\right). \tag{7}$$

After that, we compute the knowledge-enhanced context representation matrix W^c and the context-enhanced knowledge representation matrix W^k, which can be seen as the high-level document representations with salient context and knowledge features

$$W^c = \tanh(U^{c2}(H^{\text{init}} + (\mathbf{I}^c \otimes \boldsymbol{\lambda}^c) \odot E^{\text{init}})) \tag{8}$$

$$W^k = \tanh(U^{k2}(E^{\text{init}} + (\mathbf{I}^k \otimes \boldsymbol{\lambda}^k) \odot H^{\text{init}})) \tag{9}$$

where U^{c2} and U^{k2} are projection parameters, $\mathbf{I}^c = [1, \ldots, 1]^T$ indicates a d-dimensional all-ones vector, $\mathbf{I}^c \otimes \boldsymbol{\lambda}^c$ represents the Kronecker product operation between $\mathbf{I}^c$ and $\boldsymbol{\lambda}^c$, and $\odot$ refers to the elementwise multiplication.

2) Phrase-Level Attention: The important degrees of different n-gram phrases are different. Taking the sentence "Olivier Rousteing has revealed that he chose Kim and Kanye to star in Balmain's latest Campaign" as an example, the phrase "Olivier Rousteing" has much stronger information than the phrase "has revealed" in generating the target summary. To extract important n-gram phrases from the input, we adopt the n-gram convolution operation to extract local semantic features. The convolution operation involves a filter B. We assume that the feature maps for context and knowledge representations are F^c and F^k

$$F^c = \tanh(W^c * B + \mathbf{b}^p) \in \mathbb{R}^{(n-l+1) \times d_p} \tag{10}$$

$$F^k = \tanh(W^k * B + \mathbf{b}^p) \in \mathbb{R}^{(m-l+1) \times d_p} \tag{11}$$

where $\mathbf{b}^p$ is a bias matrix, l denotes the window size of the continuous words, $*$ represents the convolution operator, and d_p indicates the number of filters.

As shown in Fig. 3, a phrase-level attention mechanism is designed to learn important local n-gram chunks. We first employ the dot product between the feature maps of context and knowledge representations to calculate the correlation matrix $\tilde{M}$

$$\tilde{M} = \text{softmax}((F^c)^T F^k) \tag{12}$$

where $\tilde{M}$ is the correlation matrix between the context word chunks and the knowledge entities chunks.

We employ phrase-level attention to identify the n-grams that are essential to the meaning of the document. In particular, the knowledge and context representations are first projected into the shared subspace by linear projections. After that, the bilinear attention mechanism is employed to distinguish the important context and knowledge n-grams. Mathematically, we formulate the chunk-based context representation P^c and knowledge representation P^k as follows:

$$P^c = F^c \odot \{F^k U^{c3} \tilde{M}^T\} \tag{13}$$

$$P^k = F^k \odot \{F^c U^{k3} \tilde{M}\} \tag{14}$$

where $\odot$ denotes elementwise multiplication, U^{c3} and U^{k3} are parameters to be learned, and P^c and P^k refer to the phrase-level knowledge-enhanced context representation and context-enhanced knowledge representation, respectively.

3) Document-Level Attention: We consider the influence on the context representation from the knowledge representation and the influence on the knowledge representation from context representation, which can provide more clues to pay attention to the salient information. Inspired by [28], we devise a multiview interactive attention (MIA) network to learn comprehensive representations of the context and knowledge features. MIA leverages the interactive relation between the context and knowledge features by devising an interactive attention network. In addition, MIA explores multihead attention to learn the most crucial features from different representation subspaces.

A 2-D attention matrix is produced by MIA. Given the phrase-level knowledge-aware context representation (i.e., P^c) and context-aware knowledge representation (P^k), the attention matrix Σ^c for the document-level knowledge-enhanced context representation is computed by a neural network and is normalized through the softmax function

$$\Sigma^c = \left[\Sigma_1^c, \ldots, \Sigma_n^c\right] \tag{15}$$

$$\Sigma_i^c = \frac{\exp\left(\delta\left(\left[P_i^c; \mu(P^k)\right]\right)\right)}{\sum_{j=1}^m \exp\left(\delta\left(\left[P_j^c; \mu(P^k)\right]\right)\right)} \tag{16}$$

$$\delta\left(\left[P_i^c; \mu(P^k)\right]\right) = U^{c4} \tanh\left(U^{c5}\left[P_i^c; \mu(P^k)\right]\right) \tag{17}$$

where $\Sigma_i^c \in \mathbb{R}^\tau$ indicates the ith row of attention matrix, τ denotes the number of hops of attention, μ is the mean operation, U^{c4} and U^{c5} are parameters to be learned, and n is the length of phrase-level context representation.

In the same way, we also compute the attention matrix Σ^k for the document-level context-aware knowledge representation

$$\Sigma^k = \left[\Sigma_1^k, \ldots, \Sigma_n^k\right] \tag{18}$$

$$\Sigma_i^k = \frac{\exp\left(\delta\left(\left[P_i^k; \mu(P^c)\right]\right)\right)}{\sum_{j=1}^m \exp\left(\delta\left(\left[P_j^k; \mu(P^c)\right]\right)\right)} \tag{19}$$

where δ is defined in (17) and $\Sigma_i^k \in \mathbb{R}^r$ is the ith row of the attention matrix.

Finally, we can get the final knowledge-aware context representation O^c and the context-aware knowledge representation O^k as

$$O^c = \Sigma^c \cdot P^c \tag{20}$$

$$O^k = \Sigma^k \cdot P^k. \tag{21}$$

The knowledge-aware context representation and context-aware knowledge representation are concatenated to form the final knowledge-enhanced document representation $emb = [O^c; O^k]$ for the input article X.

V. Multitask Learning Module

To simulate the process of active reading and deeply understand a document, a multitask learning framework is proposed to simultaneously optimize the ATS task and two auxiliary tasks (i.e., text categorization and syntax annotation). In particular, the text categorization task helps to locate the most important features of the source document, and the syntax annotation task exploits word-level syntax to produce a grammatically correct summary.

A. Text Categorization Task

We use text categorization as an auxiliary task to improve the document representation learning. Text categorization and ATS tasks share the same encoder. In this way, we can make the document modeling module sensitive to the category information.

Text categorization aims at predicting a category label (e.g., "Politics" and "Sports") to each source article, which is a typical multiclass classification problem. Specifically, we use the final knowledge-enhanced document representation emb as the representation of source article X, which is then fed into a task-specific fully connected (FC) layer and a softmax layer (for probabilistic classification) to predict a category label $\hat{C}$ for article X

$$\hat{C} = \text{softmax}(U_2^{\text{text}} \cdot \tanh(U_1^{\text{text}} \cdot \text{flat}(\text{emb}) + b^{\text{text}})) \tag{22}$$

where flat is an operation that flattens matrix into vector form, U_1^{text} and U_2^{text} are learnable weight parameters, and b^{text} is a bias term.

Given an annotated training set $\{(X_{1:D}, C_{1:D})\}$, we utilize the cross entropy between the predicted category $\hat{C}$ and the ground-truth category C as our loss function for text categorization

$$\mathcal{J}_{\text{ML}}^{\text{text}}(\theta_1) = -\sum_{i=1}^{D}\sum_{j=1}^{L} C_{ij} \log(\hat{C}_{ij}) \tag{23}$$

where C_i represents the gold category of the ith document, D represents the number of documents in the training set, L represents the number of possible categories, and θ_1 represents the collective parameters for the text categorization task.

B. Shared LSTM Decoder

To mitigate the problem of generating duplicated words and incomplete sentences, we use syntax annotation as an auxiliary task, which shares the LSTM decoder with ATS task.

The decoder is implemented with an LSTM and an IKVMA, conditioning on the output vector of the encoder. Fig. 4 shows the decoding process for generating the tth word of the target summary. We use the knowledge-aware document representation (i.e., emb) as the initial state of the LSTM decoder. At time t, the LSTM decoder takes as input $\mathbf{u}_t$ (while training, $\mathbf{u}_t$ is the word embedding of y_{t-1} in the reference summary; at the testing stage, $\mathbf{u}_t$ is the word embedding of $\hat{y}_{t-1}$ produced by the decoder) and updates its hidden state $\mathbf{s}_t$ as

$$\mathbf{s}_t = \text{LSTM}(\mathbf{s}_{t-1}, [\mathbf{u}_t, \mathbf{c}_t]) \tag{24}$$

where $\mathbf{c}_t$ indicates the input context vector at time step t. In conventional attention-based methods, the input context vector $\mathbf{c}_t$ is usually computed as a weighted sum of the input

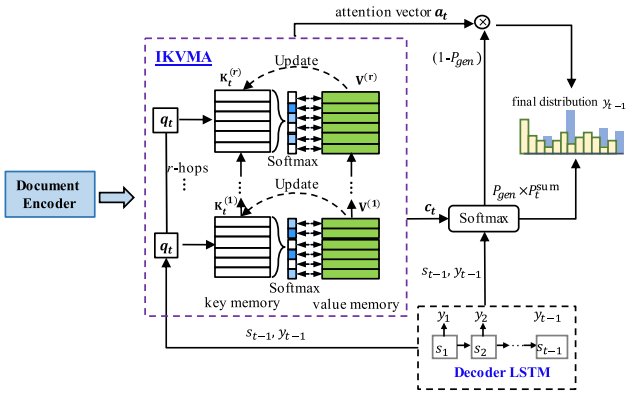

Fig. 4. Decoding process for generating the tth word of the target summary.

document representation, which ignores the attention history and the coverage information of the generated summary. To make the decoder keep track of previous attention history and attend to novel as well as appropriate salient information at each decoding step, we propose an IKVMA to help the decoder to overcome the problems of generating repetitive and faultiness summaries by automatically distinguishing which salient facets have been described and which salient facets are unexplored. We introduce the implementation details of IKVMA in Section V-B1. The address operation of IKVMA defined in (26) is used to obtain the attention weight, and then, the memory-augmented input context vector $\mathbf{c}_t$ can be computed by the read operation of IKVMA defined in (28) over the value-memory based on the learned attention vector.

1) Interactive Key-Value Memory-Augmented Attention: The IKVMA module consists of two components: a timely updated key memory $\mathbf{K} \in \mathbb{R}^{A\times d}$ to keep track of attention history and a fixed value memory $\mathbf{V} \in \mathbb{R}^{A\times d}$ to store the document features throughout the whole decoding process. Both the key memory and the value memory consist of A slots, which are initialized with the document feature vectors. At each decoding step, the ith slot in key memory stores the attention status corresponding to the ith document feature vector that is updated along with the decoding process, and the ith slot in value memory stores the representation of the ith document feature vector, denoted as $\mathbf{V}_i$. In this manner, the model can choose more appropriate salient information. Suppose that there are r rounds of memory access in each decoding step. Next, we introduce the detailed operations of IKVMA from round $r-1$ to round r.

a) Key-memory addressing: At step t, we learn a "query" vector $\mathbf{q}_t$ from the hidden state $\mathbf{s}_{t-1}$ and the previous word embedding $e(y_{t-1})$ via an LSTM

$$\mathbf{q}_t = \text{LSTM}(\mathbf{s}_{t-1}, e(y_{t-1})) \tag{25}$$

where $\mathbf{q}_t$ is used to address from the key memory. We compute the attention vector $\mathbf{a}_t^{(r)}$ over the key memory $\mathbf{K}_t^{(r-1)}$ as

$$\mathbf{a}_{t,j}^{(r)} = \text{softmax}\big(v_{t,j}^{(r)}\big) \tag{26}$$

$$v_{t,j}^{(r)} = \mathbf{v}_{\mathbf{a}}^{(r)^T} \tanh\left(\mathbf{W}_{\mathbf{a}}^{(r)}\mathbf{q}_t + \mathbf{U}_{\mathbf{a}}^{(r)}\mathbf{K}_{t,j}^{(r-1)}\right) \tag{27}$$

where $\mathbf{K}_{t,j}^{(r-1)}$ represents the jth memory slot of the state memory $\mathbf{K}_t^{(r-1)}$, $\mathbf{W}_{\mathbf{a}}^{(r)}$, $\mathbf{U}_{\mathbf{a}}^{(r)}$, and $\mathbf{v}_{\mathbf{a}}^{(r)}$ are learnable parameters, and $\mathbf{a}_{t,j}^{(r)}$ indicates the weight assigned to the jth memory slot $\mathbf{K}_{t,j}^{(r-1)}$.

b) Value-memory reading: After obtaining the attention weight $\mathbf{a}_t^{(r)}$, the attentive context document representation $\mathbf{c}_t^{(r)}$ (from round $r-1$ to round r) for generating the tth word is computed by the weighted sum of all slots in the value memory $\mathbf{V}$

$$\mathbf{c}_t^{(r)} = \sum_{j=1}^{A} \mathbf{a}_{t,j}^{(r)} \mathbf{V}_j \tag{28}$$

where $\mathbf{V}_j$ is the jth slot in static value memory.

c) Key-memory updating: The updating process of the key-memory state includes two operations: ERASE and ADD. The ERASE operation decides the content to be removed from the memory state, which is similar to the forget gate in LSTM. With the ERASE operation, the model can avoid exploring the same information for multiple times and, therefore, avoid generating repetitive words or phrases. Formally, the key-memory state after the ERASE operation is

$$\tilde{\mathbf{K}}_{t,j}^{(r)} = \mathbf{K}_{t,j}^{(r-1)}\left(1 - \omega_{t,j}^{(r)}\mathbf{F}_t^{(r)}\right) \tag{29}$$

where $\mathbf{F}_t^{(r)} = sigmoid(W_{\mathbf{F}}, \mathbf{s}_t)$ and $W_{\mathbf{F}}$ is a learnable parameter. $\omega_{t,j}^{(r)}$ indicates the weight of the jth slot of the memory state, which is computed by

$$\omega_{t,j}^{(r)} = \text{softmax}\big(\gamma_{t,j}^{(r)}\big) \tag{30}$$

$$\gamma_{t,j}^{(r)} = \mathbf{v}_{\gamma}^{(r)^T} \tanh\left(\mathbf{W}_{\gamma}^{(r)}\mathbf{q}_t + \mathbf{U}_{\gamma}^{(r)}\mathbf{K}_{t,j}^{(r-1)}\right) \tag{31}$$

where $\mathbf{W}_{\gamma}^{(r)}$, $\mathbf{U}_{\gamma}^{(r)}$, and $\mathbf{v}_{\gamma}^{(r)}$ are learnable parameters.

The ADD operation decides how much current information (new information) should be added to the visual key-memory state to track the dynamic interaction between the key memory and the decoder, which is computed as

$$\mathbf{K}_{t,j}^{(r)} = \tilde{\mathbf{K}}_{t,j}^{(r)} + \omega_{t,j}^{(r)}\tilde{\mathbf{F}}_t^{(r)} \tag{32}$$

where $\tilde{\mathbf{F}}_{i,t} = \sigma(W_{\tilde{\mathbf{F}}}, \mathbf{s}_t)$ and $W_{\tilde{\mathbf{F}}}$ is a learnable parameter.

C. Syntax Annotation and Abstractive Summarization Tasks

We use syntax annotation (also known as CCG supertag annotation) [26] as an auxiliary task, which shares the decoding module with the ATS task. Specifically, syntax annotation aims to assign lexicon categories ("noun," "noun phrase," "prepositional phrase," and "sentence") to each word in the sentence. Given a document X with n words, the syntax annotation task attempts to predict $P(Z|X)$, where $Z = \{z_1, \ldots, z_g\}$ represents the corresponding CCG supertag sequence. It is noteworthy that the generated CCG supertags and text summary have the same length.

At each decoding step, we concatenate the context vector $\mathbf{c}_t$ and the hidden state $\mathbf{s}_t$ and feed the concatenated vector to a linear function to learn decoder hidden vector O_t at time t as

$$O_t = V^o[\mathbf{s}_t, \mathbf{c}_t] + \mathbf{b}^o \tag{33}$$

where V^o is the learnable projection parameter and $\mathbf{b}^o$ is a bias term.

The generation probabilities of the tth CCG supertag [26] and the tth word can be calculated by

$$P_t^{\text{sum.}} = P(y_t|\hat{Y}_{1:t-1}; X) = \text{softmax}\left(U^{\text{sum.}} O_t + \mathbf{b}^{\text{sum}}\right) \quad (34)$$

$$P_t^{\text{syn.}} = P(z_t|\hat{Y}_{1:t-1}; X) = \text{softmax}\left(U^{\text{syn.}} O_t + \mathbf{b}^{\text{syn.}}\right) \quad (35)$$

where $U^{\text{sum.}}$ and $U^{\text{syn.}}$ denote projection parameters. $\mathbf{b}^{\text{sum.}}$ and $\mathbf{b}^{\text{syn.}}$ are bias terms. The superscripts "sum." and "syn." represent the parameters for the text summarization and supertag annotation tasks, respectively. $\hat{Y}_{1:t-1}$ represents the generated tokens up to time $t-1$. $P_t^{\text{sum.}}$ represents the probability distribution of the generated word at time t.

However, the standard seq2seq model usually struggles to process the out-of-vocabulary (OOV) words and produces a large number of "UNK" tokens. To mitigate this problem, a copy mechanism is widely used to further improve the performance of text summarization [8], [29], [30]. The pointer-generator network allows both generating words from the vocabulary and copying words with a pointer network. In particular, at each decoding step, the generation probability $P_{\text{gen}} \in [0, 1]$ is computed via (36), which is then utilized as a soft switch to select generating a word from the whole vocabulary or copying a word from the source document. Formally, at time t, the generation probability P_{gen} is calculated from the hidden state $\mathbf{s}_t$, the context representation $\mathbf{c}_t$, and the decoder input $\mathbf{u}_t$

$$P_{\text{gen}} = \sigma\left(V_c^T \mathbf{c}_t + V_s^T \mathbf{s}_t + V_u^T \mathbf{u}_t + b_{\text{gen}}\right) \quad (36)$$

where V_c, V_s, and V_u are weight parameters and b_{gen} is a bias term.

Finally, at time t, the probability of generating a candidate token w_j can be computed as

$$\bar{P}_t^{\text{sum.}}(w_j) = P_{\text{gen}} * P_t^{\text{sum.}}(w_j) + (1 - P_{\text{gen}}) * \sum \mathbf{a}_{t,j} \quad (37)$$

where $\mathbf{a}_{t,j}$ indicates the weight assigned to the jth item of the document representation at time step t, which is defined in (26). If w_j is an OOV token, then $P_t^{\text{sum.}}(w_j) = 0$; if w_j does not appear in the input document, then $\mathbf{a}_{t,j} = 0$.

We optimize the generation probability distributions of the summary and the supertag sequence by minimizing the following two separate maximum likelihood loss functions:

$$\mathcal{J}_{\text{ML}}^{\text{sum.}}(\theta_2) = -\sum_{t=1}^{m} \log\left(\bar{P}_t^{\text{sum.}}\right) \quad (38)$$

$$\mathcal{J}_{\text{ML}}^{\text{syn.}}(\theta_3) = -\sum_{t=1}^{m} \log\left(P_t^{\text{syn.}}\right) \quad (39)$$

where m is the size of the generated sequences and θ_2 and θ_3 are the parameters sets for ATS and syntax annotation tasks, respectively.

D. Joint Training

In order to strengthen the shared encoder and decoder simultaneously, we train these three relevant tasks jointly. Formally, we optimize the three tasks by minimizing the following objective function:

$$\mathcal{J}_{\text{ML}}(\Theta) = \tilde{\lambda}_1 \mathcal{J}_{\text{ML}}^{\text{text}} + \tilde{\lambda}_2 \mathcal{J}_{\text{ML}}^{\text{sum.}} + \tilde{\lambda}_3 \mathcal{J}_{\text{ML}}^{\text{syn.}} \quad (40)$$

where Θ represents the set of parameters of the HH-ATS model and $\tilde{\lambda}_1$, $\tilde{\lambda}_2$, and $\tilde{\lambda}_3$ represent the hyperparameters controlling the importance of the corresponding objective functions. We set $\tilde{\lambda}_1 = \tilde{\lambda}_2 = 0.45$ and $\tilde{\lambda}_3 = 0.1$. In the experiments, all the hyperparameters are chosen by performing the grid search with the validation data.

Policy Gradient Algorithm: In most previous work, the summarization systems are learned based on the MLE, which suffers from several learning problems. First, the training objective is different from the evaluation metrics. For example, in the summarization systems, the seq2seq model is typically learned by minimizing the cross-entropy loss. However, at the testing stage, the seq2seq model is often measured by using discrete and nondifferentiable metrics (e.g., ROUGE [31] and BLEU [32]). Second, at time t in the training stage, the LSTM decoder takes as input the gold token at time $t-1$, while at the testing stage, the token generated by the decoder at time $t-1$ is taken as the input of the decoder to generate the tth token. This exposure bias issue [33] would lead to error accumulation at the testing phase. To alleviate the aforementioned issues in decoding the target summary, we optimize the text summarization model directly toward the ROUGE-1 by maximizing the expected reward with the policy gradient algorithm [34]

$$\mathcal{J}_{\text{RL}}^{\text{sum}} = (r(\bar{y}) - r(y^s)) \sum_{t}^{m} \log\ p\left(y_t^s|Y_{1:t-1}^s; X\right) \quad (41)$$

where $\bar{y}$ represents the baseline output that is learned via greedy strategy at each decoding step, y^s represents the output obtained by sampling from $\bar{P}_t^{\text{sum.}}$, and $r(\cdot)$ is a reward function, which is implemented with GOUGE-1.

Similar to previous work [9], we first pretrain HH-ATS to optimize the multitask objective $\mathcal{J}_{\text{ML}}$ defined in (40) and then retrain HH-ATS to optimize a mixed objective function combining $\mathcal{J}_{\text{ML}}$ and $\mathcal{J}_{\text{RL}}^{\text{sum}}$

$$\mathcal{J}_{\text{mixed}}(\Theta) = \beta J_{\text{ML}} + (1 - \beta) J_{\text{RL}}^{\text{sum}} \quad (42)$$

where β represents a hyperparameter controlling the weights of $\mathcal{J}_{\text{ML}}$ and $\mathcal{J}_{\text{RL}}^{\text{sum}}$. In the experiments, we set $\beta = 0.1$.

VI. Dual Discriminator GAN Module

We employ a DD-GAN to further enhance the ATS. DD-GAN contains a generator and two discriminators, where the generator attempts to synthesize high-quality and plausible summaries so as to "fool" the two discriminators, while the two discriminators on the contrary are expected to score down the generated summaries. Their minimax game is summarized as the following objective function $\mathcal{L}$:

$$\mathcal{L} = \min_{\Theta} \max_{\Phi_1, \Phi_2} \mathbb{E}_{Y \sim P_{\text{data}}}[\log D_1(Y)] + \mathbb{E}_{\hat{Y} \sim G}[\log(1 - D_1(\hat{Y}))] + \mathbb{E}_{Y \sim P_{\text{data}}}[\log D_2(Y)] + \mathbb{E}_{\hat{Y} \sim G}[\log(1 - D_2(\hat{Y}))] \quad (43)$$

where Θ, Φ_1, and Φ_2 represent the parameter sets of generator G, binary discriminator D_1, and ranking-based discriminator D_2, respectively, and $\mathbb{E}$ indicates the mathematical expectation. Next, we will describe the generator G and two discriminators (D_1 and D_2) in detail.

A. Binary Discriminator D_1

We first propose a binary discriminator D_1, which aims at distinguishing the input summary as synthesized by generator G or written by humans. In particular, we use the convolutional neural network (CNN) [35] to encode the source article, and a max-pooling operation is then applied over the features. We pass these pooled features to an FC layer followed by a softmax layer to predict the probability of being "gold summary."

In adversarial training, we use the binary discriminator D_1 as a reward function to enhance the performance of the generator G. By fixing the parameters of generator G, we can optimize the parameters of binary discriminator D_1 by minimizing the following loss function:

$$\mathcal{J}_{D_1} = -\mathbb{E}_{Y\sim P_{\text{data}}}[\log D_1(Y)] - \mathbb{E}_{\hat{Y}\sim G}[\log(1 - D_1(\hat{Y}))]. \quad (44)$$

B. Ranking-Based Discriminator D_2

The binary discriminator D_1 usually suffers from obtaining unsatisfactory performance. It may achieve incredible high accuracy, which would make the synthetic summary have a reward around zero. This is because the binary discriminator D_1 is capable of recognizing the synthetic summary with high confidence. To make the generated summary more coherent and precise to the source article, we also propose a ranking-based discriminator D_2 that uses a ranking model to provide guidance to the generator G.

Specifically, the source article X, the human-written summary Y, and the synthetic summary $\hat{Y}$ are all projected to a distributed representation space with a CNN. The network for X is denoted as f_x, while Y and $\hat{Y}$ share a network f_y. Given these representations, the similarities of $f_x(X)$ versus $f_y(Y)$ and $f_y(\hat{Y})$ are calculated via a scoring function ϖ. Given the source article X, the probabilities of the article-summary pairs $\langle X, Y\rangle$ and $\langle X, \hat{Y}\rangle$ being correctly ranked can be computed as

$$D_2(\langle Y, \hat{Y}\rangle \mid X) = \psi(\varpi(f_x(X), f_y(Y)) - \varpi(f_x(X), f_y(\hat{Y}))) \quad (45)$$

where ψ is the sigmoid function and ϖ is a scoring function.

The objective of ranking-based discriminator D_2 is to maximize the probability that the positive pair $\langle X, Y\rangle$ is ranked higher than the negative pair $\langle X, \hat{Y}\rangle$, guiding the generator G to produce a more plausible summary. Equivalently, by fixing the parameters of the generator G, we can optimize the parameters of ranking-based discriminator D_2 by minimizing the following loss function:

$$\mathcal{J}_{D_2} = -\mathbb{E}_{\langle Y,Y^-\rangle\sim P_{\text{data}}}[\log D_2(\langle Y, Y^-\rangle)] - \mathbb{E}_{\hat{Y}\sim G}[\log(1 - D_2(\langle Y, \hat{Y}\rangle))] \quad (46)$$

where Y^- represents the negative summary randomly chosen from the training set with the ground-truth summary excluded. This optimization problem is typically solved with gradient descent since the objective function $\mathcal{J}_{D_2}$ is differentiable with respect to Φ_2.

C. Generator G

Given a source article X, the objective of the generator G is to generate a plausible and precise summary that can "fool" both discriminators. When the discriminators D_1 and D_2 are obtained and fixed, we are ready to optimize the parameters of generator G by minimizing the following objective function:

$$\mathcal{J}_G = \mathbb{E}_{\hat{Y}\sim G}\{[\log(1 - D_1(\hat{Y}))] + [\log(1 - D_2(\langle Y, \hat{Y}\rangle))]\}. \quad (47)$$

Since it is difficult to backpropagate the gradients from the discriminators to the generator G, we use the policy gradient algorithm [34] to optimize this objective function similar to (41).

VII. Experimental Setup

A. Data Sets

We conduct extensive experiments on the following two benchmark data sets.

1) CNN/Daily Mail: We measure our HH-ATS on the CNN/Daily Mail data set [36], which is a benchmark to evaluate ATS models. This data set contains news articles and corresponding multisentence human-written summaries. In total, it contains 287 226 training samples, 13 368 validation samples, and 11 490 test samples. On average, each article contains 781 tokens and each summary has 56 tokens.

2) Gigaword: This data set is automatically created by using the first sentence of each article as the source and its headline as the summary [2], [37]. Following [2], we utilize publicly available scripts[2] to preprocess the data. In total, the Gigaword data set contains approximately 3.8M training pairs, 400k validation pairs, and 400k test pairs.

We conduct data preprocessing before training our HH-ATS model. Each document is tokenized using a popular NLP toolkit NLTK.[3] Two independent vocabularies are built for the source articles and corresponding summaries, respectively. In the experiments, we merely keep 20 000 most frequent words and replace the remaining rare words with an "*UNK*" token.

For the text categorization task, we search the source web page of each article and obtain its category label. These news articles are divided into 11 categories: "Sports," "Showbiz," "Politics," "Opinion," "Tech," "Travel," "Health," "Crime," "Living," "Business," and "Other." We can also train the text classifier on more large and universe news data sets to increase the generality of the proposed model and apply it for ATS, motivated by the remarkable success of the object classifier pretrained on ImageNet [38] in various computer vision tasks.

For the syntax annotation task, the training data are annotated with CCG supertags[4] [26], where each word has a corresponding supertag. A finite set of CCG supertags are adopted, including "noun (N)," "noun phrase (NP)," "prepositional phrase (PP)," and "sentence (S)." Note that both the summary and the CCG supertag sequence have the same length.

[2]https://github.com/kyunghyuncho/dl4mt-material
[3]http://www.nltk.org
[4]https://github.com/uwnlp/EasySRL

B. Implementation Details

For each data set, we use the validation set with the grid search algorithm to choose hyperparameters automatically. Similar to the parameters setting of [8], we truncate each source article and its summary to 400 tokens and 100 tokens, respectively. We adopt Freebase [39] as our KB for both data sets. The graph embeddings are initialized through choosing the values from a normal distribution $\mathcal{N}(0, 1)$ and set the dimension of the graph embedding to be 100. The number of candidate entities for each entity mention is tuned from 1 to 10 with the step size of 1, and we finally set the number to be 5 (i.e., $N = 5$).

We utilize pretrained word2vec embeddings of size 100 [40] to initialize the word embeddings. The recurrent weight matrices are initialized as orthogonal matrices, and the other weight parameters are initialized with the normal distribution $\mathcal{N}(0, 0.01)$. We initialize the bias terms to be zero. Both the LSTM encoder and the LSTM decoder have the dimensionality of 256. For the convolutional layer of the phrase-level attention network, we set the number of convolution filters of CNN to be 256. The size of each convolution filter is set to be 2. The number of heads (i.e., τ) in the document-level attention network is set to be 3. There are two rounds of memory access for the IKVMA model in each decoding step ($r = 2$). Similar to previous work [8], we first train the proposed model with respect to a standard MLE objective (ML objective). We apply an early stopping method with a window size of three to stop the training process. In particular, the training process is terminated if the validation loss does not decrease for three continuous epochs.

After pretraining the model with ML objective, HH-ATS is switched to a mixed training objective, incorporating the ML objective $\mathcal{J}_{\text{ML}}$ and the RL objective $\mathcal{J}_{\text{RL}}^{\text{sum}}$. The scaling factor β is tuned from 0 to 0.45 with a step size of 0.05, and we finally set $\beta = 0.1$. ROUGE-1 metric is chosen as the reinforcement reward function, measuring the similarity between the partial generated summary and the ground-truth summary. The Adam optimizer [41] is applied for training, with the learning rate of 0.0001. We adopt the beam search algorithm (beam size $= 5$) at the decoding phase.

After the pretraining stage, the generative model and discriminative models in HH-ATS are further trained alternatively. In particular, the generative model updates for five times, and then, the discriminative models are retrained to keep a good pace with the generative model. HH-ATS is implemented in TensorFlow and trained on a single Tesla P100 GPU.

C. Baseline Methods

We compare our HH-ATS model with several strong competitors, which are described as follows.

1) *ABS and ABS+:* These two methods are implemented with the standard encoder–decoder RNNs with attention mechanisms and a " large vocabulary trick" [2].
2) *RAS-LSTM and RAS-Elman:* A recurrent attentive summarizer is proposed to generate summaries, which uses a convolutional attention-based encoder to help the decoder focus on the important words of the source articles [42].
3) *PGC:* This method extends the seq2seq model by allowing the model to copy words from the source article [8].
4) *DeepRL:* A policy gradient algorithm is employed to mitigate the exposure bias issue in the MLE objective [9]. Here, we adopt the ML+RL version since it can generate more informative and readable summaries.
5) *GANsum:* This is the first work that explores adversarial training to improve the overall performance of ATS [10].
6) *MATS:* A multitask learning method is employed to enhance the ATS task, which trains ATS and two coupled tasks simultaneously in a unified framework [14].
7) *HATS:* This method introduces a hybrid learning model for abstractive summarization, which simulates the different stages of how humans write a summary [25].
8) *Re*^{3}*:* This retrieve, rerank, and rewrite model is proposed in [43], which uses summaries from training set as soft templates to guide the process of generating summaries.

TABLE I

AUTOMATIC EVALUATION RESULTS ON THE CNN/DAILY MAIL DATA SET. ALL THE SCORES HAVE A 95% CONFIDENCE INTERVAL OF AT MOST ±0.25

Methods	ROUGE-1	ROUGE-2	ROUGE-L
ABS	35.46	13.30	32.65
ABS+	35.63	13.75	33.01
RAS-LSTM	37.46	15.11	34.45
RAS-Elman	38.25	16.28	35.43
PGC	39.53	17.28	36.38
DeepRL (RL)	41.16	15.82	39.08
DeepRL (ML+RL)	39.87	15.82	36.90
GANsum	39.92	17.65	36.71
MATS	40.74	18.14	37.15
HATS	42.16	19.17	38.35
HH-ATS (RL)	42.93	19.85	38.64
HH-ATS (ML+RL)	**43.16**	**20.32**	**39.14**

VIII. EXPERIMENTAL RESULTS

A. Automatic Evaluation

Similar to previous studies [2], we adopt ROUGE-1, ROUGE-2, and ROUGE-L [31] to verify the effectiveness of HH-ATS, which are the most widely used automatic metrics in evaluating text summarization systems. Specifically, ROUGE-1 and ROUGE-2 measure the quality of a summary by calculating overlapping n-grams between the candidate summary and the gold summary. ROUGE-L computes the longest common sequence (LCS) between the candidate summary and the gold summary. ROUGE-1 (unigram) and ROUGE-2 (bi-gram) primarily consider informativeness, whereas ROUGE-L measures the readability of the generated summary.

To account for the randomness in the experiments, we run our HH-ATS model five times on each test set and report the average ROUGE scores in Tables I and II. Our HH-ATS model substantially outperforms the baseline methods by a significant margin. Specifically, PGC consistently performs better than ABS. This may be because the copy mechanism

TABLE II
AUTOMATIC EVALUATION RESULTS ON THE GIGAWORD DATA SET. ALL THE SCORES HAVE A 95% CONFIDENCE INTERVAL OF AT MOST ±0.25

Methods	ROUGE-1	ROUGE-2	ROUGE-L
ABS	29.55	11.32	26.42
ABS+	29.78	11.89	26.97
RAS-LSTM	32.55	14.70	30.03
RAS-Elman	33.78	15.97	31.15
PGC	33.44	16.09	31.43
DeepRL (RL)	35.82	16.64	32.45
DeepRL (ML+RL)	35.16	16.75	31.68
GANsum	35.04	16.55	31.96
MATS	35.56	16.97	32.94
HATS	36.78	18.65	33.96
Re^3	37.04	19.03	34.46
HH-ATS (RL)	38.06	19.28	35.82
HH-ATS (ML+RL)	**38.43**	**19.75**	**36.11**

used in PGC can deal with the OOV words. DeepRL and GANsum perform better than PGC since they employ the policy gradient algorithm to mitigate the exposure bias issue and optimize the evaluation metric directly. HH-ATS performs even better than the strong compared methods by exploring the knowledge-aware hierarchical attention, the IKVMA, and the DD-GAN.

B. Ablation Study

To measure the impact of each component on the performance of HH-ATS, we conduct ablation study for HH-ATS in terms of removing the commonsense knowledge from KB (w/o KB), text categorization (w/o text), syntax generation (w/o syntax), and GAN framework (w/o GAN). In addition, we also ablate multiple components of HH-ATS. Since it is difficult to obtain all the factor combinations, we mainly ablate the three phases (i.e., knowledge-based attention module, multitask module, and GAN framework). In particular, we conduct the ablation test by: 1) removing text categorization and syntax annotation (w/o text+syntax); 2) removing KB, text categorization, and syntax annotation (w/o KB+text+syntax); 3) removing KB and GAN (w/o KB+GAN); and 4) removing text categorization, syntax annotation, and GAN (w/o text+syntax+GAN).

The ablation results are provided in Tables III and IV. In general, all the parts contribute a great improvement to HH-ATS. From Tables III and IV, we can observe that the ROUGE and human evaluation results decrease sharply when discarding the GAN framework. This is within our expectation since the RL reward provided by the two discriminators can guide HH-ATS to produce more informative and plausible summaries. In addition, commonsense knowledge from KB also contributes to the effectiveness of HH-ATS, which indicates that the prior knowledge from KB helps to learn more comprehensive document representations. However, the improvement of ROUGE scores by integrating syntax annotation is relatively limited. This may be explained by the fact that the issue of generating duplicated words and incomplete sentences has little impact on the automatic evaluation metrics. In contrast, syntax annotation contributes a great improvement in terms of human evaluation.

TABLE III
ABLATION TEST RESULTS ON THE CNN/DAILY MAIL DATA SET

Methods	ROUGE-1	ROUGE-2	ROUGE-L
HH-ATS	**43.16**	**20.32**	**39.14**
w/o KB	42.52	19.56	38.25
w/o text	41.74	19.08	38.21
w/o syntax	42.83	19.82	38.60
w/o GAN	41.69	19.05	37.86
w/o text+syntax	41.32	18.64	37.97
w/o KB+text+syntax	40.43	18.10	37.15
w/o KB+GAN	40.61	18.26	37.31
w/o text+syntax+GAN	40.15	17.90	36.84

TABLE IV
ABLATION TEST RESULTS ON THE GIGAWORD DATA SET

Methods	ROUGE-1	ROUGE-2	ROUGE-L
HH-ATS	**38.43**	**19.75**	**36.11**
w/o KB	37.73	19.28	35.36
w/o text	37.85	19.05	35.42
w/o syntax	38.21	19.37	35.80
w/o GAN	37.52	18.59	34.75
w/o text+syntax	37.45	18.71	35.07
w/o KB+text+syntax	36.70	17.83	33.85
w/o KB+GAN	36.12	17.63	33.63
w/o text+syntax+GAN	35.75	17.37	33.24

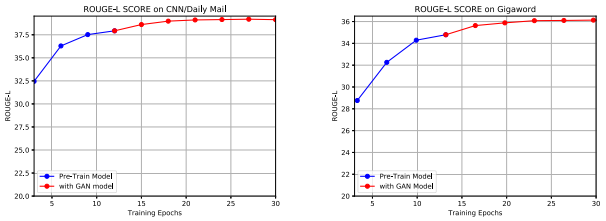

Fig. 5. Learning curves of HH-ATS in terms of ROUGE-L.

C. Learning Curve of GAN

To better understand the learning process of HH-ATS, we illustrate the training curves of HH-ATS in Fig. 5. Due to the limited space, we merely report the learning curves corresponding to ROUGE-L on the two data sets. ROUGE-1 and ROUGE-2 exhibit a similar trend. As shown in Fig. 5, at the pretraining phase, HH-ATS gets converged in 12 epochs on the CNN/Daily data set and 13 epochs on the Gigaword data set. The ROUGE-L values are further boosted on both data sets when employing the DD-GAN framework, which verifies that the discriminators can provide effective rewards to guide the generator in generating more plausible summaries.

D. Human Evaluation

We also perform a manual evaluation to measure the informativeness and fluency of the generated summaries. Similar to [44], we randomly choose 200 articles from test data and ask three human annotators to evaluate the generated summaries by considering both the informativeness (if the summary identifies salient information from the original article) and fluency (if the summary is fluent with a proper grammatical structure). From the informativeness perspective, the annotators need to

TABLE V
HUMAN EVALUATION RESULTS ON THE CNN/DAILY MAIL DATA SET

Methods	Informativeness	Fluency
ABS	2.46	2.35
ABS+	2.54	2.47
RAS-LSTM	2.75	2.70
RAS-Elman	2.82	2.73
PGC	2.83	3.14
DeepRL (ML+RL)	3.16	2.95
GANsum	3.04	3.02
MATS	3.07	3.09
HATS	3.17	3.13
Re^3	3.15	3.09
HH-ATS (ML+RL)	**3.34**	**3.25**

TABLE VI
HUMAN EVALUATION RESULTS ON THE GIGAWORD DATA SET

Methods	Informativeness	Fluency
ABS	2.68	2.73
ABS+	2.72	2.74
RAS-LSTM	28.6	2.78
RAS-Elman	2.81	2.85
PGC	3.04	3.12
DeepRL (ML+RL)	3.21	3.09
GANsum	3.19	3.18
MATS	3.17	3.15
HATS	3.25	3.21
Re^3	3.23	3.17
HH-ATS (ML+RL)	**3.41**	**3.32**

read the entire article and its summary carefully and attempt to understand the primary information. If the original article and its summary contain similar key information, then this summary will be given a higher score. From the fluency perspective, if the summary is not readable (e.g., has grammatical mistakes), a lower score will be given to the summary. The annotators need to assign each summary a score of 1 (bad), 2 (poor), 3 (not bad), 4 (satisfactory), and 5 (good) for informativeness and fluency separately.

Tables V and VI summarize the human evaluation results of all models. HH-ATS outperforms other methods on both informativeness and fluency, which indicates that our model is capable of capturing comprehensive salient information from the source articles and generating high-quality natural language summaries.

E. Case Study

To evaluate HH-ATS qualitatively, we report some summaries generated by different models. Due to the limited space, we select an article from the test set of the CNN/Daily Mail data set and illustrate the generated summaries by different models for comparison. We report the results in Table VII. We observe that HH-ATS tends to produce a more informative and specific summary given the source article. First, the knowledge-attention network, which mimics the rough reading process, learns the commonsense knowledge form KB as prior knowledge to distinguish the important features from the source document and decide the focus of the summary. For example, HH-ATS can successfully focus on the words "Olivier Rousteing" that are contained in KB. After discarding the rough reading (commonsense knowledge from KB), the generated summary loses salient information, as shown in Table VII. Second, the articles in different categories may have significantly different summary styles. Table VII shows a case study of HH-ATS by removing the text categorization task. From the results, we can observe that HH-ATS w/o text categorization is prone to generating a low-quality summary that misses the salient entities in the article. Third, syntax knowledge plays an essential role in text generation. Enforcing syntax conformance alleviates the problems of duplicated words and incomplete sentences. As we can see in Table VII, HH-ATS w/o syntax annotation is prone to generating a low-quality summary with an improper grammatical structure. In contrast, the HH-ATS model, which shares the decoder with the syntax annotation task, generates a more plausible and fluent summary. Fourth, the quality of the

TABLE VII
EXAMPLE OF SOURCE ARTICLE FROM SHOW CATEGORY AND ITS SUMMARIES GENERATED BY DIFFERENT MODELS. THE WORDS IN RED INDICATE THE INCOMPLETE OR REDUNDANT WORDS

Category: Show
Article (truncated): "they are one of the world's most famous couples-and have quickly gained respect among the fashion elite. and now, one esteemed designer has revealed why kim kardashian and kanye west have the midas touch. olivier rousteing has revealed that he chose kim and kanye to star in balmain's latest campaign because they 'represent a family for the new world'. scroll down for video. fashion's most well-connected designer, olivier rousteing, has revealed why he snapped kim kardashian and kanye west up to front his balmain campaign. (.....) olivier-who regularly dresses kim, 34, and her siblings for the red carpet-explained that when kendall jenner and kim wear his clothes, they look like a 'fashion army'. kim and kanye this week made trips to france and armenia with their daughter, north west. the trip to the religious mecca reportedly included north being baptised in the country where her late father's side of the family originated from. kim kardashian , kanye west and north visit the geghard monastery in armenia and take in the sights. kim, kanye and north have become a fashionable family. pictured here with alia wang, aimie wang and nicki minaj at the alexander wang show in february 2014."
Reference summary: "olivier rousteing has revealed why he chose kim and kanye for balmain. designer says the couple are among the most talked-about people. fashionable couple love wearing matching designs by balmain designer."
Summary by HH-ATS: "olivier rousteing has revealed why he chose kim and kanye to star in balmain's latest campaign because they represent a family for the new world. french designer says the couple are among the most talked-about people."
Summary by HH-ATS w/o KB: "kanye west kim kardashian rocking balmain. the 29-year-old creative director has revealed that he was inspired to feature the couple - who have a 22-month-old daughter north - in the label's spring/summer 2015 men's campaign"
Summary by HH-ATS w/o text categorization: "olivier rousteing has seen kim kardashian and kanye west. kim kardashian and kanye west have worn his clothes."
Summary by HH-ATS w/o syntax annotation: "olivier rousteing has revealed why kim and kanye west his balmain campaign. french designer the couple couple are among the most talked-about."
Summary by HH-ATS w/o GAN: "olivier rousteing has revealed why he chose kim and kanye to star in balmain's latest campaign because they represent a family for the new world. the 29-year-old creative director has revealed that he was inspired to feature the couple-who have a 22-month-old daughter north - in the label's spring/summer 2015 men's campaign."

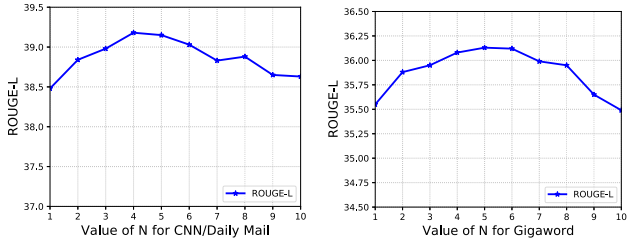

Fig. 6. Experimental results of HH-ATS by varying the value of N.

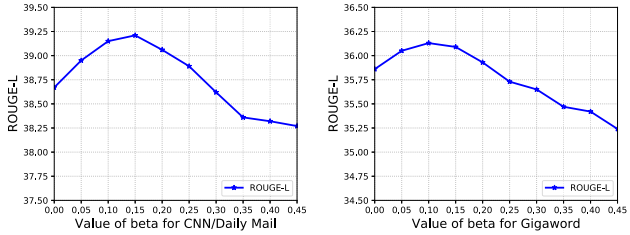

Fig. 7. Experimental results of HH-ATS by varying the value of β.

generated summary can be further improved with DD-GAN by deleting, rephrasing, and adding words into the candidate summary.

F. Hyperparameter Analysis

In this section, we investigate the impact of two important hyperparameters on the overall performance of HH-ATS, including the number of candidate entities from KB for each entity mention (i.e., N) and the scaling hyperparameter β defined in (42).

1) Effect of Hyperparameter N: N represents the number of entity candidates from KB for each entity mention in the source article. In this experiment, we analyze the impact of N on the overall performance of HH-ATS by varying its value from 1 to 10 with a step size of 1. The ROUGE-L scores on CNN/Daily Mail and Gigaword data sets are reported in Fig. 6. HH-ATS obtains the best ROUGE-L scores when $N = 4$ or 5. As N increases from 1 to 10, the ROUGE-L scores grow sharply until we achieve the optimal values, after which the ROUGE-L scores decrease slightly.

2) Effect of Hyperparameter β: The scaling hyperparameter β defined in (42) decides the weights of the MLE and RL training. By varying the value of β, we can analyze the relative importance of the two objectives in ATS. The special case of $\beta = 0$ means that no MLE (ML) objective is employed in (42). We vary the values of β from 0 to 0.45 with a step size of 0.05 and compute the ROUGE-L values on the CNN/Daily Mail and Gigaword data sets. Experimental results are summarized in Fig. 7. We can achieve the best ROUGE-L scores when $\beta = 0.1$ or 0.15 on the data sets. It implies that as the value of β increases from 0 to 0.45, the ROUGE-L scores increase until the optimal values, after which the ROUGE-L scores begin to decrease. Evidently, the RL objective can further improve the performance of ATS.

IX. Conclusion and Future Work

In this article, we introduce an HH-ATS, which mimics how humans approach this task. HH-ATS contains three primary components (i.e., a knowledge-based hierarchical attention module, a multitask module, and a DD-GAN framework), which correspond to the three phases of human reading cognition process (i.e., rough reading, active reading, and poste-diting). Experimental results demonstrate that the proposed HH-ATS model achieves higher ROUGE scores than state-of-the-art baseline methods. In addition, the human evaluation also shows that HH-ATS can generate summaries with better informativeness and fluency.

In the future, we plan to explore different automatic evaluation metrics that may match the human judgments better. In addition, we would like to explore the graph convolutional network or dependence parser to capture crucial long-distance patterns.

References

[1] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 379–389.

[2] R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," in *Proc. 20th SIGNLL Conf. Comput. Natural Lang. Learn.*, 2016, pp. 280–290.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] P. G. Avery and M. F. Graves, "Scaffolding young learners' reading of social studies texts," *Social Stud. Young Learner*, vol. 9, no. 4, pp. 10–14, 1997.

[5] E. L. Toprak and G. Almacioglu, "Three reading phases and their applications in the teaching of english as a foreign language in reading classes with young learners," *J. Lang. Linguistic Stud.*, vol. 5, no. 1, pp. 20–36, 2009.

[6] M. Pressley and P. Afflerbach, *Verbal Protocols Reading: The Nature Constructively Responsive Reading*. Evanston, IL, USA: Routledge, 2012.

[7] C. Tarchi *et al.*, "Comprehending and recalling from text: The role of motivational and cognitive factors," *Educ. Res.*, vol. 27, no. 3, p. 600, 2017.

[8] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 1073–1083.

[9] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," in *Proc. ICLR*, 2018, pp. 1–12.

[10] L. Liu, Y. Lu, M. Yang, Q. Qu, J. Zhu, and H. Li, "DepthLGP: Learning embeddings of out-of-sample nodes in dynamic networks," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 8109–8110.

[11] Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy, "Deep reinforcement learning for sequence-to-sequence models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2469–2489, Jul. 2019.

[12] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.

[13] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 160–167.

[14] Y. Lu, L. Liu, Z. Jiang, M. Yang, and R. Goebel, "A multi-task learning framework for abstractive text summarization," in *Proc. AAAI*, 2019, pp. 9987–9988.

[15] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-Y. Wang, "Representation learning using multi-task deep neural networks for semantic classification and information retrieval," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2015, pp. 912–921.

[16] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," in *Proc. ICLR*, 2016, pp. 1–10.

[17] H. Zhang, L. Xiao, Y. Wang, and Y. Jin, "A generalized recurrent neural architecture for text classification with multi-task learning," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 2873–2879.

[18] M. Yang *et al.*, "Mares: Multitask learning algorithm for Web-scale real-time event summarization," *World Wide Web*, vol. 22, no. 2, pp. 499–515, 2019.

[19] M. Yang, X. Wang, Y. Lu, J. Lv, Y. Shen, and C. Li, "Plausibility-promoting generative adversarial network for abstractive text summarization with multi-task constraint," *Inf. Sci.*, vol. 521, pp. 46–61, Jun. 2020.

[20] M. Yang, W. Huang, W. Tu, Q. Qu, Y. Shen, and K. Lei, "Multitask learning and reinforcement learning for personalized dialog generation: An empirical study," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 5, 2020, doi: 10.1109/TNNLS.2020.2975035.

[21] I. Goodfellow *et al.,* "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[22] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," 2017, *arXiv:1701.06547*. [Online]. Available: http://arxiv.org/abs/1701.06547

[23] T.-H. Chen, Y.-H. Liao, C.-Y. Chuang, W.-T. Hsu, J. Fu, and M. Sun, "Show, adapt and tell: Adversarial training of cross-domain image captioner," in *Proc. ICCV*, vol. 2, 2017, pp. 521–530.

[24] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Proc. AAAI*, vol. 2017, pp. 2852–2858.

[25] M. Yang, Q. Qu, W. Tu, Y. Shen, Z. Zhao, and X. Chen, "Exploring human-like reading strategy for abstractive text summarization," in *Proc. AAAI*, 2019, pp. 7362–7369.

[26] S. Clark, "Supertagging for combinatory categorial grammar," in *Proc. 6th Int. Workshop Adjoining Grammar Rel. Frameworks*, 2002, pp. 19–24.

[27] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 701–710.

[28] D. Ma, S. Li, X. Zhang, and H. Wang, "Interactive attention networks for aspect-level sentiment classification," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 4068–4074.

[29] C. Gulcehre, S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio, "Pointing the unknown words," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 140–149.

[30] J. Gu, Z. Lu, H. Li, and V. O. K. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1631–1640.

[31] C.-Y. Lin and E. Hovy, "Manual and automatic evaluation of summaries," in *Proc. Workshop Autom. Summarization*, 2002, pp. 899–902.

[32] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics*, 2001, pp. 311–318.

[33] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. ICLR*, 2016, pp. 1–4.

[34] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. NIPS*, 2000, pp. 1057–1063.

[35] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1746–1751.

[36] K. M. Hermann *et al.*, "Teaching machines to read and comprehend," in *ICNIPS*. Cambridge, MA, USA: MIT Press, 2015, pp. 1693–1701.

[37] D. Graff, J. Kong, K. Chen, and K. Maeda, "English gigaword," *Linguistic Data Consortium*, vol. 4, no. 1, p. 34, 2003.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[39] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 1247–1250.

[40] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 2013, pp. 3111–3119.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[42] S. Chopra, M. Auli, and A. M. Rush, "Abstractive sentence summarization with attentive recurrent neural networks," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2016, pp. 93–98.

[43] Z. Cao, W. Li, S. Li, and F. Wei, "Retrieve, rerank and rewrite: Soft template based neural summarization," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 152–161.

[44] J. Cheng and M. Lapata, "Neural summarization by extracting sentences and words," in *Proc. 54th Annu. Meeting Assoc. Comput.*, 2016, pp. 484–494.

Min Yang received the Ph.D. degree from The University of Hong Kong, Hong Kong, in 2017.

She is currently an Associate Professor with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. Her current research interests include machine learning and natural language processing.

Chengming Li (Member, IEEE) received the B.S. and M.S. degrees in computer application technology from the Dalian University of Technology, Dalian, China, in 2009 and 2011, respectively, and the Ph.D. degree from the Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan, in 2015.

His research interests include data mining, information-centric networking, network security, and big data.

Ying Shen (Member, IEEE) received the Ph.D. degree in medical and biomedical information science from the University of Paris Ouest Nanterre La Défense, Nanterre, France, in 2015.

She is currently an Associate Professor with Sun Yat-sen University, Guangzhou, China. Her research interests include medical informatics and natural language processing.

Qingyao Wu (Member, IEEE) received the M.S. and Ph.D. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 2009 and 2013, respectively.

He is currently a Professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. His research interests include transfer learning and multilabel learning.

Zhou Zhao (Member, IEEE) received the B.S. and Ph.D. degrees in computer science from The Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2010 and 2015, respectively.

He is currently an Associate Professor with the College of Computer Science, Zhejiang University, Hangzhou, China. His research interests include machine learning, data mining, and information retrieval.

Xiaojun Chen (Member, IEEE) received the Ph.D. degree from the Harbin Institute of Technology, Harbin, China, in 2011.

He is currently an Assistant Professor with the College of Computer Science and Software, Shenzhen University, Shenzhen, China. His research interests include machine learning, clustering, feature selection, and massive data mining.