Joint Modeling of Characters, Words, and Conversation Contexts for Microblog Keyphrase Extraction

Yingyi Zhang
Department of Information Management, School of Economics and Management, Nanjing University of Science and Technology, Nanjing, Jiangsu, China. E-mail: yingyizhang@njust.edu.cn

Chengzhi Zhang*
Department of Information Management, School of Economics and Management, Nanjing University of Science and Technology, Nanjing, Jiangsu, China. E-mail: zhangcz@njust.edu.cn

Jing Li
Tencent AI Lab, Shenzhen, Guangdong, China. E-mail: ameliajli@tencent.com

Millions of messages are produced on microblog platforms every day, leading to the pressing need for automatic identification of key points from the massive texts. To absorb salient content from the vast bulk of microblog posts, this article focuses on the task of microblog keyphrase extraction. In previous work, most efforts treat messages as independent documents and might suffer from the data sparsity problem exhibited in short and informal microblog posts. On the contrary, we propose to enrich contexts via exploiting conversations initialized by target posts and formed by their replies, which are generally centered around relevant topics to the target posts and therefore helpful for keyphrase identification. Concretely, we present a neural keyphrase extraction framework, which has 2 modules: a conversation context encoder and a keyphrase tagger. The conversation context encoder captures indicative representation from their conversation contexts and feeds the representation into the keyphrase tagger, and the keyphrase tagger extracts salient words from target posts. The 2 modules were trained jointly to optimize the conversation context encoding and keyphrase extraction processes. In the conversation context encoder, we leverage hierarchical structures to capture the word-level indicative representation and message-level indicative representation hierarchically. In both of the modules, we apply character-level representations, which enables the model to explore morphological features and deal with the out-of-vocabulary problem caused by the informal language style of microblog messages. Extensive comparison results on real-life data sets indicate that our model outperforms state-of-the-art models from previous studies.

Introduction

Microblog platforms have become popular outlets for individuals to voice opinions and exchange information, such as Twitter. Tweets are the posts produced by users on Twitter, which contains abundant opinions and content they are interested in. It has many downstream applications. First is user portrait construction. Depending on user portrait, platforms can recommend friends and advertising users are interested in. Since tweets are advantageous at reflecting the interests and opinions of users, constructing portraits by tweets may improve the accuracy of recommendation. Second is event forecasting and tracking. Tweets have been leveraged to conduct disease predicting (Chew & Eysenbach, 2010), election trend predicting (Zheng, Liu, Wu, & Tan, 2018), and social event tracking (Ahmad, Pogorelov, Riegler, Conci, & Halvorsen, 2019). However, the huge volume of user-generated data produced daily on these platforms has resulted in an explosive growth of data, exceeding the reading and understanding capacity of human beings, not to mention discovering the gist information and making sense of them. As a result, there exists a growing demand for techniques to automatically recognize key excerpts from massive microblog posts. Among those techniques, keyphrase extraction approaches serve as an effective alternative, which aims to recognize salient phrases in one or multiple words to reflect the key focus of a given collection (Turney, 2000). Keyphrase extraction has proved useful to downstream applications such as information retrieval (Choi, Croft, & Kim, 2012) and event tracking (Ribeiro, Gershman, de Matos, Neto, & Carbonell, 2017). Particularly, keyphrases have played an important role in knowledge mining on tweets. For instance, Chew and

*Corresponding author

Received December 27, 2018; revised April 10, 2019; accepted May 28, 2019

 DOI: 10.1002/asi.24279

Eysenbach (2010) use the keyphrases in tweets to track H1N1 disease. In their study, they leverage the keyphrase to link tweets on the same topic or event together. In addition, Efron (2010) used keyphrases to assist information retrieval in Twitter.

To date, most studies on related fields have focused on the usage of ranking-based models (Bellaachia & Al-Dhelaan, 2012; Marujo et al., 2015) or sequence-tagging models (Zhang, Wang, Gong, & Huang, 2016) via treating messages as independent documents. However, because of the severe data sparsity issue, it is arguable that these methods are suboptimal in recognizing key content from *short* and *informal* microblog messages. There are two main reasons that data sparsity exhibits: the feature limitation problem caused by the short length, and the out-of-vocabulary (OOV) problem resulting from the informal language style.

To address the feature limitation problem, this article exploits the conversations context initiated by the target posts. In the conversation context, replying messages conveying personal opinions on previously discussed points can be utilized to enrich context for those original short messages (Chang, Wang, Mei, & Liu, 2013; Li, Liao, Gao, He, & Wong, 2016). To illustrate the usefulness of conversation contexts in keyphrase extraction, Table 1 displays the target post about the Pittsburgh Steelers, an American football team competing in the National Football League (NFL). Given the 15 words in the target post, it is difficult to understand why the keyphrase should be "Steelers." If looking at the replying messages, we can observe that they form a conversation centered around the topic raised in the target post. It can be easily identified that crucial words are mentioned multiple times in the conversation, which therefore provide useful clues to the keyphrase identification. For instance, content reflecting the discussion focus, such as the names of other teams ("Packers" and "Skins") and other topic-related words ("championships" and "athletes") can effectively indicate the keyphrase "Steelers." Also, the keyword "Steelers" reoccurs in [R3] and [R5]. Thus, it is vital to capture salient words from the conversation context. It can also be observed that there exist some meaningless messages, such as [R1] and [R6], which do not contain any crucial words. If we treat these messages as being as important as others, they will create interference. Hence, capturing salient messages from a conversation context will be helpful. Such context information embedded in the conversation is nevertheless ignored in most previous studies (Chang et al., 2013; Li, Gao, Wei, Peng, & Wong, 2015). These studies apply various filtering strategies to select crucial messages from a conversation context, such as support vector machine (SVM), convention neural network (CNN), and the attention mechanism. At the same time, these strategies merely capture the message-level features and do not have the ability to learn the word-level and message-level information synchronously. The hierarchical structure is one of the technologies that can jointly capture the word-level and the messages-level features. It has proven to be useful in tasks such as dialog predicting (Serban, Sordoni, Bengio, Courville, & Pineau, 2016) and document classification (Yang et al., 2016). Hence, this article proposes a conversation context encoder with hierarchical structures to encode conversation context in word-level and sentence-level synchronously.

As for the OOV problem, caused by the microblog users' informal writing styles and the prominence of nonstandard words (for example, abbreviations and misspelled words), we argue that the morphological representations should be effectively captured. Since this article utilizes a pretrained word embedding to initialize word representations, in this study an OOV word is a word that does not exist in the pretrained word embedding. To better explain the OOV problem, Table 2 displays the statistic information of the OOV words in our two data sets. As Table 2 shows, in both Twitter data sets OOV words make up 50% of the total vocabulary. Such high-proportioned OOV words indicate that alleviating the OOV problem is of utmost importance. The character-level word embedding, one of the techniques that learn the character information of each word, can help alleviate this problem. It has proven useful in name entity recognition (Kuru, Can, & Yuret, 2016) and sentiment entity recognition (Jebbara & Cimiano, 2017).

In this article, we propose a novel neural keyphrase extraction framework consisting of two modules: a conversation context encoder and a keyphrase tagger. By having a conversation context encoder, we are able to capture indicative representation from the conversation context to alleviate the feature

TABLE 1. A sample Twitter conversation about "Steelers." [Ri]: The i-th message in conversation ordered by their posting time.

The target post for keyphrase extraction:
Win or lose this *Super Bowl*, the **Steelers** are the preeminent franchise in *NFL* history.
Messages forming a conversation:
[R1]: If you leave out the first 50 years, yeah.
[R2]: Fair point, but I actually considered that. I believe modern-era *championships* should be weighed much more heavily.
[R3]: That's because you are a Modern Era Guy. *Steelers* did nothing as a franchise for 38 years. Cannot overlook that.
[R4]: Well I cannot deny all bias, but the *competition* is exponentially greater since merger. Way more *teams*, way better *athletes.*
[R5]: But again, if the *Packers* win the *Super Bowl* it'll be their 13th *NFL* title. The *Steelers* are trying to win No. 7.
[R6]: Your argument is sound. Not fully convinced, but you have weakened my conviction.
[R7]: Those definitely count. The *Skins* are 5 time *champs*!;)

Note: **Steelers**: the keyphrase to be detected; Italic words: content words concerning the key topic in targeting post and indicative of the keyphrase.

DOI: 10.1002/asi

TABLE 2. The out-of-vocabulary statistic information on two data sets.

Data set	Vocab.	OOV-Vocab.	OOV-Percentage (%)
Daily-Life	85,699	42,753	49.89
Election-Trec	69,434	39,857	57.40

limitation problem, and then the indicative representation is fed into the keyphrase tagger, which is employed to extract salient words from target posts. The conversation context encoder consolidates a hierarchical structure to capture word-level features and message-level features in a conversation context. Conversation context encoders with hierarchical structures are called hierarchical encoders in this article. We apply three kinds of hierarchical encoders: BiLSTM hierarchical encoders, Attention-based BiLSTM hierarchical encoders (Yang et al., 2016), and Memory network hierarchical encoders. In both the conversation context encoder and the keyphrase tagger, in order to alleviate the OOV problem, we apply the character-level word embedding to learn character-level features.

The experimental results show that on real-life Twitter data sets, keyphrase extraction frameworks with hierarchical encoders and character-level word embeddings yield higher F1 scores than other models. Quantitative and qualitative analyses indicate that hierarchical encoders are useful to capture the word-level features and sentence-level features from the conversation context. Besides, the character-level word embedding can be leveraged to alleviate the OOV problem. The contribution of this study lies in three areas: one, to the best of our knowledge, we are the first to jointly encode word-level information and message-level information from conversation context to help keyphrase extraction. Moreover, we are the first to leverage character information to overcome the OOV problem in keyphrase extraction. Third, we propose a memory network hierarchical encoder in a new type.

Related Work

Previous works on keyphrases extraction mainly focused on formal texts like news reports and scientific articles. Existing keyphrase extraction models can be categorized as ranking-based models and tagging-based models.

Ranking-based methods mainly include models based on graph ranking and word frequency. As for graph ranking methods, the TextRank model proposed by Mihalcea and Tarau (2004) ranks keywords based on the co-occurrence graphs among words in a single document. A range of optimization methods are proposed to improve the performance of TextRank (Liu, Li, Zheng, & Sun, 2009). For instance, Wan and Xiao (2008) extended the context of a document by employing neighbor documents to provide more knowledge to TextRank. Liu et al. (2009) changed the weight of nodes in the graph with a topic-related degree of words by Latent Dirichlet Allocation topic algorithm. To improve the accuracy of weight assignment of the relationship between words, Martinez-Romo, Araujo, and Fernandez (2016) proposed a strategy in which the relationship between words is measured by the significant co-occurrence and the relationship in WordNet. To test the significant co-occurrence of two words, a null model-based statistical hypothesis test was employed. As for word frequency methods, the term frequency-inverse document frequency (TF-IDF) model proposed by Salton and Buckley (1988) ranks keywords based on the word frequency in a document and corpus and has proved efficient in information retrieval (Jones, 2004) and content summarization (Kireyev, 2009).

Tagging models focus on using manually crafted features for binary classifiers to predict keyphrases. Keyphrase extraction algorithm (KEA) is a well-known model based on the Naive Bayes algorithm with two features: the TF-IDF score and the position of the first occurrence of a phrase (Witten, Paynter, Frank, Gutwin, & Nevill-Manning, 1999). KEA has been demonstrated to be effective in extracting keyphrases (Jones & Paynter, 2002). To harness more features to improve the tagging models, Tang, Li, Wang, and Cai (2004) presented an approach based on Bayesian decision theory with new selected features: mutual information and word linkage. Medelyan and Witten (2014) proposed the KEA++ algorithm based on KEA. Besides the two features used in KEA, KEA++ employed another two features: word length and node degree. Ercan and Cicekli (2007) selected lexical chains related features and to apply a decision tree to extract keyphrases. Conditional Random Fields (CRF) is another frequently used algorithm in keyphrase extraction. It was first utilized to indicate keyphrases from a document by Zhang et al. (2008) and gained good performance. Yu, Xuan, and Zheng (2012) integrated the document structure in the CRF-based keyphrase extraction model. In their model, words have various weights in different parts of a document when training. Our models are in the line of tagging approaches, and provide an alternative that incorporates additional knowledge from conversations.

Recently, keyphrase extraction methods have been extended to social media texts (Bellaachia & Al-Dhelaan, 2012; Marujo et al., 2015; Zhang et al., 2016; Zhao et al., 2011); for example, Twitter and Sina Weibo. Marujo et al. (2015) extracted a keyphrase from a single text. This model is based on the MAUI toolkit (2010), which trains a decision tree over a large set of manually engineered features; for example, TF-IDF score, Brown clustering information, and word vectors. However, feature engineering is time-consuming and labor-intensive. To overcome these drawbacks, some neural network models, which can learn features from a training corpus automatically, were proposed recently and have proven to be effective in keyphrase extraction. For instance, Zhang et al. (2016) proposed a neural network model to extract a keyphrase from a single tweet. In this model, keyphrase extraction is regarded as a sequence-labeling task and they only use the internal information in tweets. In the light of Derczynski et al. (2015), extracting information is particularly challenging in microblogs since they have a number of genre-specific characteristics, such as short messages and noisy content, and extracting a keyphrase from single tweets suffers from the severe data sparsity

DOI: 10.1002/asi

problems. To overcome these limitations, Zhang, Li, Song, and Zhang (2018) utilized conversation context generated by replies messages to help identify keyphrases from tweets. However, their models have two major limitations. First, all replies are fed into the context encoders at the same time, which will omit the message-level features in the conversation context and the word-level features in the messages. Second, user-generated content is in the informal language style and may contain misspelled words and rare words, which lead to a severe OOV problem. To alleviate the first limitation, this article proposes keyphrase extraction models with hierarchical structure-based conversation context encoders. The hierarchical structure has the ability to capture the word-level features of messages and message-level features of conversation context. To overcome the OOV problem, the character-level word embedding is leveraged to learn the morphological representations by capturing character-level features from words.

Keyphrase Extraction Neural Network Model

Our keyphrase extraction frameworks consist of two modules: a keyphrase tagger and a conversation context encoder. The keyphrase tagger aims to identify keyphrases from a target post. The conversation context encoder captures the salient content in conversation context, which would help indicate keyphrases from the target post. To capture both word-level features and message-level features from conversation contexts, we integrate the hierarchical structure with conversation context encoders, which are called hierarchical encoders. To learn the character-level information from words and thereby alleviate the OOV problem, we combine the character-level word embedding into both the keyphrase tagger and the conversation context encoder. The entire framework is learned synchronously with the given target posts and their corresponding conversation context. In prediction, the keyphrase tagger identifies keyphrases in a post with the help of representations generated by the conversation context encoder. Figure 1 shows the overall structure of our keyphrase extraction framework. As shown in the Figure 1, in the conversation context x_i^r, each character $x_{i,t,w,c}^r$ of the word $x_{i,t,w}^r$ and the word $x_{i,t,w}^r$ are fed into the character-level word embedding. Then the output $\tilde{v}_{i,t,w}^r$ of the character-level word embedding are fed into the hierarchical conversation context encoder. In the target post x_i, each character $x_{i,t,w,c}$ of the word $x_{i,t,w}$ and the word $x_{i,t,w}$ are fed into the character-level word embedding. Then the output $\tilde{v}_{i,t,w}^r$ of the character-level word embedding are concatenated with the output e_i^r of the conversation context encoder and is then fed into the keyphrase tagger.

In the following sections, we first introduce the hierarchical conversation context encoders and character-level word embeddings. Then the keyphrase taggers are described.

Hierarchical Conversation Context Encoder

We apply three hierarchical encoders, that is, BiLSTM hierarchical encoder (H-BiLSTM), attention-based BiLSTM hierarchical encoder (H-Att (BiLSTM)), and memory network hierarchical encoder (H-MemNN). Three encoders have various functions. First, H-Att (BiLSTM) and H-MemNN

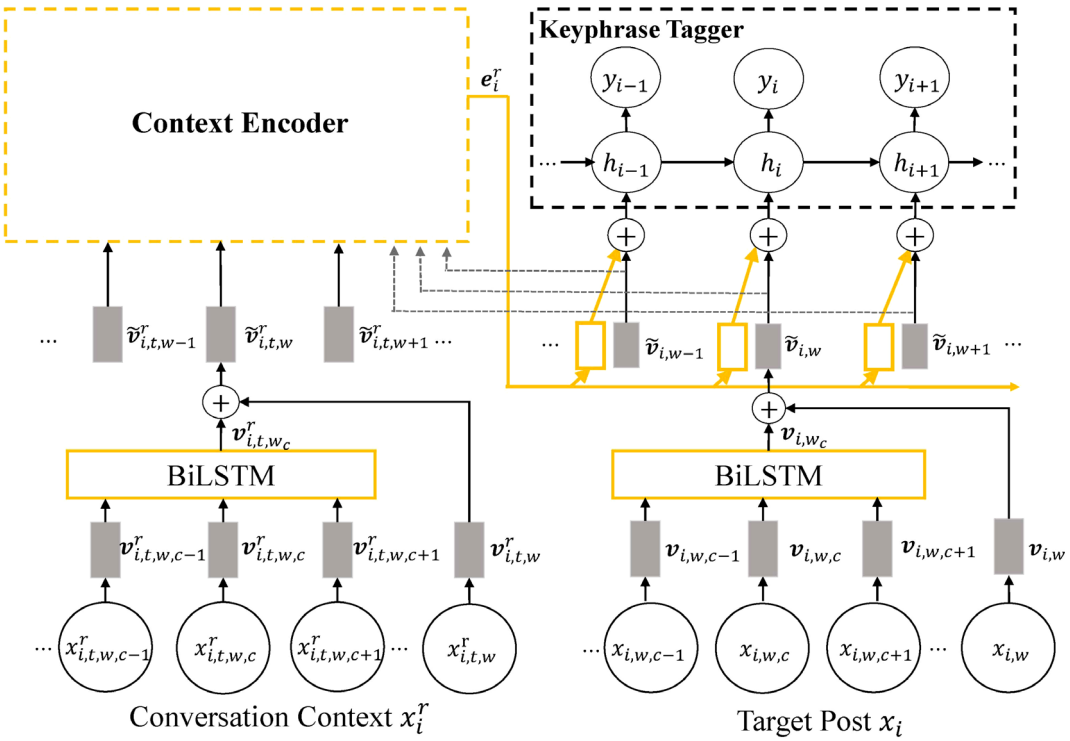

FIG. 1. The framework of keyphrase extraction. [Color figure can be viewed at wileyonlinelibrary.com]

DOI: 10.1002/asi

explores salient words and salient messages from a conversation context that describe the main focus of the conversation context, which helps indicate keyphrases from a target post, while H-BiLSTM encoders regard all the words and messages equally important in a conversation context. Furthermore, H-MemNN encoders explicitly exploit the affinity of target posts and the conversation context in matching each other, while H-Att (BiLSTM) encoders implicitly highlight certain context without taking target posts into account.

BiLSTM hierarchical encoder. Word Modeling. In H-BiLSTM, the word representation is captured in a message with a BiLTSM encoder, namely, BiLSTM word-level encoders. As Figure 2 shows, formally, given the message $x^r_{i,t}$, each word $x^r_{i,t,w}$ is represented as a vector $\tilde{\mathbf{v}}^r_{i,t,w}$ mapped by an embedding layer. Vector $\tilde{\mathbf{v}}^r_{i,t,w}$ is then fed into the word-level encoder. As BiLSTM has two opposite directions, the content representation of $x^r_{i,t}$, denoted by $\mathbf{h}^r_{i,t}$, takes the concatenation of the last states from both directions, which come from two ends of a given turn.

Message Modeling. To learn message representations of conversation context x^r_i, our model employs another BiLSTM, namely, BiLSTM message-level encoders. The t-th state of the message-level encoder takes the representation of the t-th message $x^r_{i,t}$ as an input. Similar to a word modeling process, we concatenate the last states from both directions of the message-level encoder. Then the output e^r_i of the conversation context is fed into the keyphrase tagger.

Attention based BiLSTM hierarchical encoder. Word Modeling. H-Att (BiLSTM) puts an attention mechanism on the BiLSTM model for "soft addressing" important words in a message, namely, the attention-based word-level encoder. In this article, we use the feed-forward attention, as shown in Figure 3. The encoder is thus represented as:

$$\tilde{\mathbf{v}}^r_{i,t} = \sum_{w=1}^{|x^r_{i,t}|} \alpha^r_{i,t,w} \mathbf{h}^r_{i,t,w} \tag{1}$$

where $\alpha^r_{i,t,w}$ is the attention coefficient obtained for word $x^r_{i,t,w}$, which implicitly reflects its importance, and $\alpha^r_{i,t,w}$ is computed via a softmax over the hidden states by:

$$\alpha^r_{i,t,w} = softmax\left(a\left(\mathbf{h}^r_{i,t,w}\right)\right) \tag{2}$$

where $a(.)$ is a learnable function formulated as:

$$a\left(\mathbf{h}^r_{i,t,w}\right) = \tanh\left(\mathbf{W}_a \mathbf{h}^r_{i,t,w}\right) \tag{3}$$

which takes input only from on $\mathbf{h}^r_{i,t,w}$. $\mathbf{W}_a$ are parameters of the function $a(.)$ to be learned.

Message Modeling. To learn message representation of the conversation context $\mathbf{x}^r_i$, we use another BiLSTM with an attention mechanism on it to "soft address" salient messages in the conversation context, namely, attention-based message-level encoders. The t-th state of the message-level encoder takes the representation of the t-th message $\mathbf{x}^r_{i,t}$ as input. And similar to the word modeling process, we use the feedforward attention and the encoder is thus represented as:

$$\mathbf{e}^r_i = \sum_{t=1}^{|\mathbf{x}^r_i|} \alpha^r_{i,t} \mathbf{h}^r_{i,t} \tag{4}$$

Memory network hierarchical encoder. Word Modeling. The word representation is captured in a message level with a memory network word-level encoder. As show in Figure 4, formally, each embedded context sequence $\mathbf{v}^r_{i,t} = <\mathbf{v}^r_{i,t,1}, \mathbf{v}^r_{i,t,2}, \ldots, \mathbf{v}^r_{i,t,|x^r_{i,t}|}>$ is stored into memory $\mathbf{M}_{i,t}$. We then yield the match between embedded target

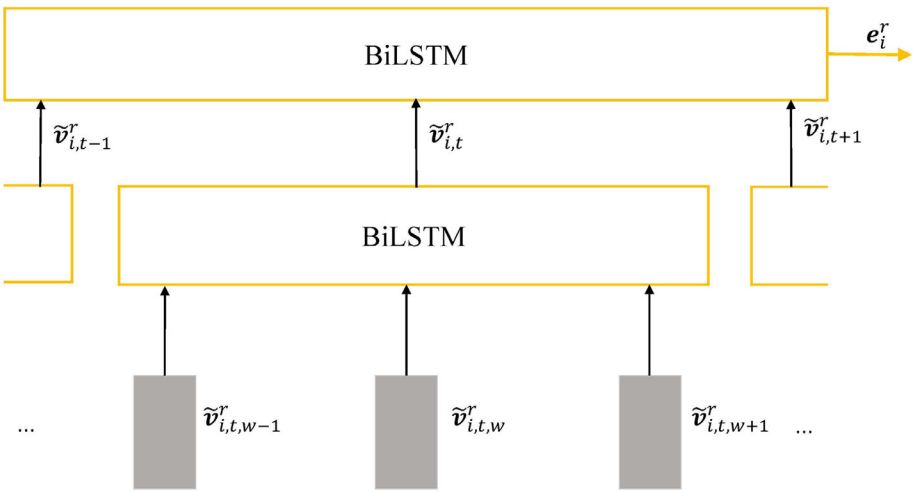

FIG. 2. The structure of the BiLSTM hierarchical conversation context encoder. [Color figure can be viewed at wileyonlinelibrary.com]

DOI: 10.1002/asi

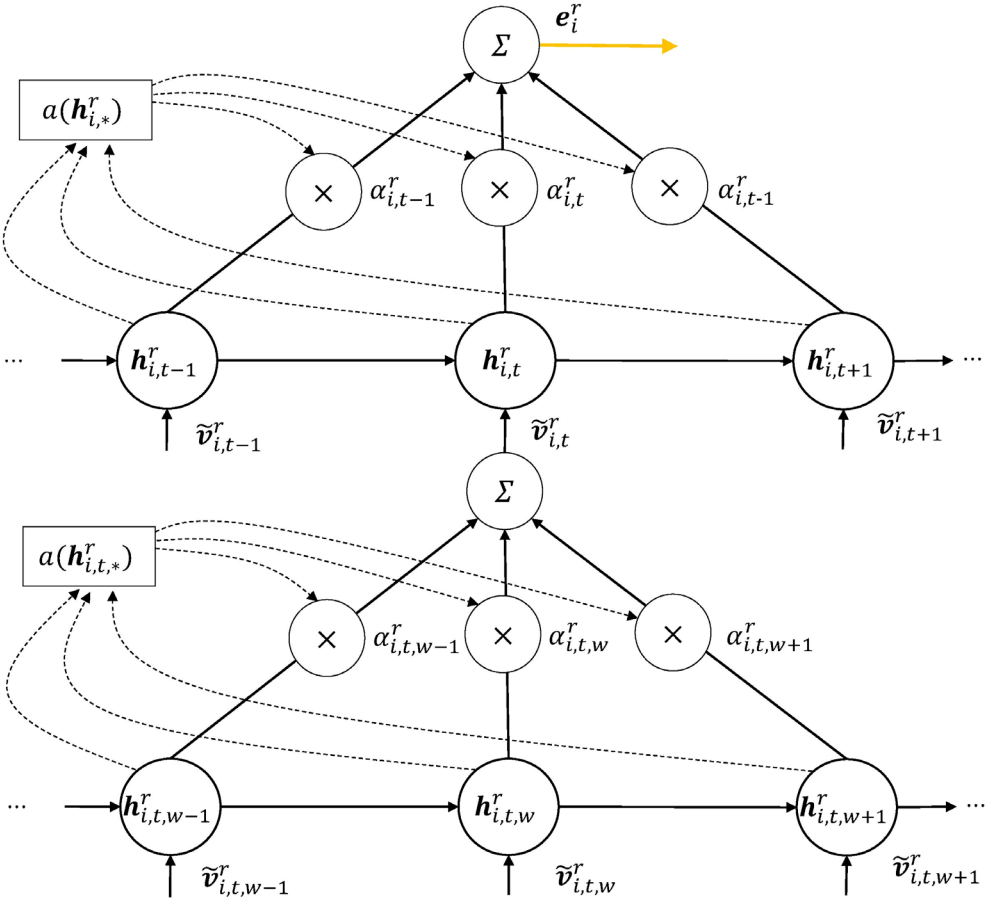

FIG. 3. The structure of attention-based BiLSTM hierarchical conversation context encoder. [Color figure can be viewed at wileyonlinelibrary.com]

post $\mathbf{v}_{i,t}$ and context memory $\mathbf{M}_{i,t}$ by their inner product activated by softmax, where $\mathbf{P}_{i,t}$ captures the similarity between the *w-th* word in message $x_{i,t}^r$ and the *w′-th* word in target post x_i. To transform input $x_{i,t}^r$ into an aligned form to make it be able to be added with $\mathbf{P}_{i,t}$, we include another embedding matrix $\mathbf{C}_{i,t}$. The sum of $\mathbf{C}_{i,t}$ and matching matrix $\mathbf{P}_{i,t}$ serves as the encoded representation for each message in the conversation context.

Message Modeling. To learn message representation of conversation context x_i^r, we use another memory network, named the memory network message-level encoder. Before being fed to the message-level encoder, the outputs of the word-level encoder requires a dimensionality reduction processing. In this article, we use the average dimensionality reduction method, which computes the mean of elements along the dimension of the target post. After dimension reducing, each embedded message sequence $\mathbf{v}_i^r = <\mathbf{v}_{i,1}^r, \mathbf{v}_{i,2}^r, \ldots, \mathbf{v}_{i,|x_i^r|}^r>$ is stored into memory $\mathbf{M}_i$. We then yield the match between embedded target post $\mathbf{v}_i$ and context memory $\mathbf{M}_i$ by their inner product activated by softmax, where $\mathbf{P}_i$ captures the similarity between the t-th turns in conversation context x_i^r and the t-th message in target post x_i. To transform context input x_i^r into an aligned form to make it able to be added with $\mathbf{P}_i$, we include another embedding matrix $\mathbf{C}_i$. Similar to the attention encoder, the MemNN encoder aims to generate a representation, which addresses the important messages in the conversation context that helps extracting keyphrases from target post x_i. The sum of $\mathbf{C}_i$ and matching matrix $\mathbf{P}_i$ serves as the encoded representation for conversation context.

Character-Level Word Embedding

A particular challenge involved in keyphrase extraction from social media is from the fact that user-generated contents can be of low quality and contain misspelled words and rare words, and so on. Character information can bring relevant information that regular word embedding (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) lacks. Hence, character information might be beneficial in the keyphrase extraction task, allowing a model to be robust to OOV words.

We utilize the character-level word embedding model proposed by Jebbara and Cimiano (2017), which jointly uses character representations and word representations and has proven efficient and effective. We take the embedding layers in conversation context encoder modules as an example. Given the character sequence $\mathbf{c}_{i,w}^r = <x_{i,w,1}^r, x_{i,w,2}^r, \ldots, x_{i,w,|x_{i,w}^r|}^r>$ of a word $x_{i,w}^r$, each character $x_{i,w,c}^r$ is transformed to its corresponding character embedding vector $\mathbf{v}_{i,w,c}^r$ using a character embedding matrix $W^c \in \mathbb{R}^{dc \times |V\ c|}$:

DOI: 10.1002/asi

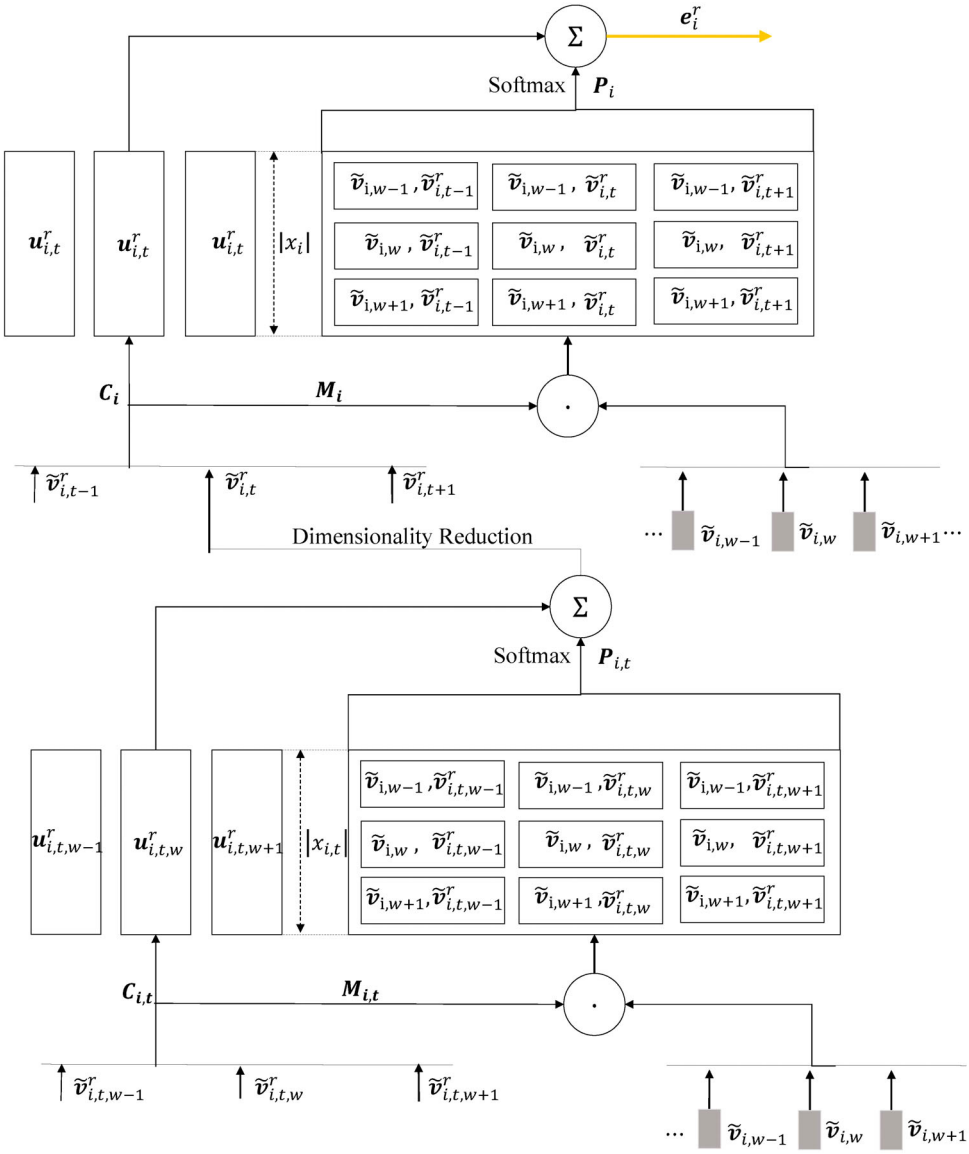

FIG. 4. Memory network hierarchical conversation context encoder. [Color figure can be viewed at wileyonlinelibrary.com]

$$\mathbf{v}^r_{i,w,c} = \mathbf{W}^c \mathbf{e}^{x^r_{i,w,c}} \tag{5}$$

Analogously to the procedure for word embedding, $|\mathbf{V}^c|$ is the character vocabulary and $\mathbf{e}^{x^r_{i,w,c}}$ is a one-hot vector of size $|\mathbf{V}^c|$ representing the character $\mathbf{x}^r_{i,w,c}$. $\mathbf{d}^c$ represents the dimension of the character vector. The sequence of character embeddings is passed through a BiLSTM layer to obtain a representation $\mathbf{h}^c$ for the character sequence. Lastly, the hidden state $\mathbf{h}^c$ is transformed to the final character-level word embedding using a linear feedforward layer:

$$\mathbf{v}^r_{i,w_c} = \mathbf{W}^{cw}\mathbf{h}^c + \mathbf{b}^{cw} \tag{6}$$

With $\mathbf{W}^{cw} \in \mathbb{R}^{dc \times 2.dc}$ and $\mathbf{b}^{cw} \in \mathbb{R}^{dc}$.

To incorporate the word embedding with character-level word embedding, each word is passed through the word embedding layer to obtain $\mathbf{v}^r_{i,w} = \left\{\mathbf{v}^r_{i,1}, \mathbf{v}^r_{i,2}, \ldots, \mathbf{v}^r_{i,|\mathbf{x_i}|}\right\}$. The character-level word embeddings $\mathbf{v}^r_{i,w_c}$ are then concatenated with the word level embeddings $\mathbf{v}_{i,w}$:

$$\tilde{\mathbf{v}^r} = \left[\mathbf{v^r_{i,1}} : \mathbf{v^r_{i,1_c}}\right], \left[\mathbf{v^r_{i,2}} : \mathbf{v^r_{i,2_c}}\right], \ldots, \left[\mathbf{v^r_{i,|x_i|}} : \mathbf{v^r_{i,|x_{i_c}|}}\right] \tag{7}$$

TABLE 3. Definitions of different $y_{i,w}$.

Tag	Meaning
SINGLE	$x_{i,w}$ is a one-word keyphrase (keyword).
BEGIN	$x_{i,w}$ is the first word of a keyphrase.
MIDDLE	$x_{i,w}$ is part of a keyphrase but it is neither the first nor the last word of the keyphrase.
END	$x_{i,w}$ is the last word of a keyphrase
NOT	$x_{i,w}$ is not a keyword or part of a keyphrase.

DOI: 10.1002/asi

The sequence $\tilde{v^r}$ then pass through the hierarchical conversation context encoders of the framework. Since $\tilde{v^r}$ contains character-level and word-level information, the following layers can theoretically make use of the abundant information, such as morphological representations, to improve the performance of keyphrase extraction.

Keyphrase Tagger

Keyphrase taggers are utilized to indicate a keyphrase from target posts. We follow Zhang et al. (2016) to cast keyphrase extraction into the sequence-tagging task. As shown in Figure 1, our keyphrase taggers are built on an input feature map **I(.)**, which embed each word $x_{i,w}$ in a target post into a dense vector format, that is, $I(x_{i,w}) = \mathbf{v}_{i,w}$. In the keyphrase tagger, two sequence-tagging patterns are employed, that is, single-layer tagging patterns and joint-layer tagging patterns. These two patterns are described in detail in the following paragraphs.

Single-Layer Tagging Patterns. Formally, given a target microblog post x_i formulated as word sequence $<x_{i,1}, x_{i,2}, \ldots, x_{i,|x_i|}>$, where $|x_i|$ denotes the length of x_i, we aim to produce a tag sequence $<y_{i,1}, y_{i,2}, \ldots, y_{i,|x_i|}>$, where $y_{i,w}$ indicates whether $x_{i,w}$ is part of a keyphrase. In detail, $y_{i,w}$ has five possible values:

$$y_{i,w} \in \{SINGLE, BEGIN, MIDDLE, END, NOT\} \quad (8)$$

Table 3 lists the definition of each value. In the light of the research by Zhang et al. (2016), it has shown that keyphrase extraction methods with this 5-value tagset perform better than those with binary outputs, that is, only marked with yes or no for a word to be part of a keyphrase. To predict words tags, we use Bi-LSTM, a state-of-the-art neural network.

Joint-Layer Tagging Pattern. In addition to single-layer tagging pattern, we also use the joint-layer tagging pattern proposed by Zhang et al. (2016), which is a state-of-the-art keyphrase tagger in previous works. As a multitask learner, joint-layer taggers tackle two tasks with two types of outputs, $y^1_{i,w}$ and $y^2_{i,w}$. $y^1_{i,w}$ has a binary tagset, which indicates whether word $x_{i,w}$ is part of a keyphrase or not. $y^2_{i,w}$ employs the 5-value tagset defined in Table 3.

Data Set

Data Collection

Our experiments were conducted on two data sets: the Election-Trec data set and the Daily-Life data set. Both data sets were collected from Twitter. The Election-Trec data set is constructed based on the TREC2011 microblog track[1] and the Election data set[2] shared by Zeng et al. (2018). For the TREC2011 data set, to recover conversations, we used the tweet search API[3] to retrieve full information of a tweet with its "in reply to status id" included. Recursively, we searched the "in reply to" tweet until the entire conversation was recovered. Note that we do not consider retweet relations, that is, reposting behaviors on Twitter, because retweets provide limited extra textual information for the reason that Twitter did not allow users to add comments in retweets until 2015. The Election data set is directly used, since it contains the full conversation of each Twitter already. The Daily-Life data set was collected from January to April of 2018 using tweet's streaming API with a small set of daily life keywords. Then the conversations were recovered using a tweet search API in a recursive way.

Data Processing

For keyphrase annotation, we follow Zhang et al. (2016) to use microblog hashtags as gold-standard keyphrases. They evaluated the quality of the hashtags by randomly selecting 1,000 tweets and choosing three volunteers. Every tweet was assigned a score of 2 (perfectly suitable), 1(suitable), or 0 (unsuitable) to indicate whether the hashtag of the tweet was a good keyphrase for it. The results showed that 90.2% were suitable and 66.1% were perfectly suitable. This demonstrated that the hashtag can be regarded as the ground truth keyphrase of tweets. In Twitter, the hashtag is designed to provide a way for users to embed metadata in their posts, which indicate the specific semantic domain of the post (Kapanova & Fidanova, 2017; Small, 2011). In previous studies, the hashtag was utilized to help conduct information retrieval (Chew & Eysenbach, 2010) and event tracking (Efron, 2010). Thus, we employ the hashtag in the tweet as the ground truth keyphrases. To guarantee the quality of our constructed data set, we filtered all microblog posts by two rules: first, there is only one hashtag per post; second, the hashtag is inside a post, that is, containing neither the first nor the last word of a post. Then we removed all the "#" symbols in hashtags before keyphrase extraction. For both Twitter data sets, we randomly sampled 80% for training, 10% for development, and the other 10% for testing. Table 4 reports the statistics of the two data sets. We preprocessed both Twitter data sets with the Twitter NLP tool[4] for tokenization.

Since there are no spaces between words in hashtags, we use some strategies to segment hashtags. There are two kinds of hashtags in the data sets. One is the "multi-word" that contains both capitals and lowercase, the other is the "single-word" in all lowercase or capitals. Table 5 shows the statistic information of "single-word" and "multi-word" in two Twitter data sets. If a hashtag is a "multi-word," we segment hashtags with two patterns, first is (*capital*) ⋆ (*lowercase*)+, which represents one capital followed by one or more lowercases, second is (*capital*)+, which

[1] https://trec.nist.gov/data/tweets/

[2] http://www.ccs.neu.edu/home/luwang/datasets/microblog_conversation.zip

[3] http://developer.twitter.com/en/docs/tweets/search/api-reference/get-saved_searches-show-id

[4] http://www.cs.cmu.edu/ark/TweetNLP/

DOI: 10.1002/asi

TABLE 4. Statistics of two data sets.

Data set	# of annot. Msgs	# of msgs in context	Context length	Vocab
Daily-Life				
Train	12,827	5.18	97.01	79,558
Vali	1,610	5.10	97.46	19,924
Test	1,610	5.10	98.47	19,859
Election-Trec				
Train	24,210	4.01	70.77	64,640
Vali	3,027	4.18	75.46	18,764
Test	3,027	4.22	75.66	18,956

Note: Train, Dev, and Test denotes training, development, and test set, respectively. # of annot. Msgs: number of messages with keyphrase annotation, each containing conversation context. # of msgs in context: average count of message in conversation context. Context length: average count of words in the conversation context. Vocab: vocabulary size.

represents one or more capitals. When doing hashtag segmentation, the first pattern is utilized first and then the second pattern is applied. For instance, for the hashtag "LondonMarathon," the first pattern is utilized and the hashtag is divided into two words "london" and "marathon." For the hashtag "MTPDaily," the first pattern is utilized first to match the word "daily," and then the second pattern is applied to match the word "mtp." Meanwhile, if a hashtag is a "single-word," it also may consist of more than one word. If a "single-word" consists of one word, it is called "one-word single-word." Otherwise, it is called "self-invented word," which means the word is invented by users. The proportion of "self-invented word" is small in data sets. For instance, in the Daily-life data set 96.52% hashtags are "multi-words" and "one-word single-word." In "The Impact of Character Embedding on OOV Words", qualitative analyses demonstrate that the character-level word embedding has the ability to identify "self-invented word."

Experimental Setup

Training Setup

For keyphrase taggers based on BiLSTM, we follow Zhang et al. (2016) and set their state size for each direction to 150. Joint-layer taggers employ the same hyperparameters according to Zhang et al. (2016). The character-level word embedding layer employs the same hyperparameters according to Jebbara and Cimiano (2017). The state size of the conversation context encoders share the same settings with keyphrase taggers. In training, the entire keyphrase extraction framework uses cross-entropy loss and the RMSprop optimizer (Graves, 2013) for parameter updating.

We initialized input feature map $\mathbf{I}$ for target post and $\mathbf{I}^r$ for the conversation context by embeddings pretrained on a large-scale external microblog collection from Twitter. Twitter embeddings were trained on 99 M tweets with 27B tokens and 4.6 M words in the vocabulary.

TABLE 5. Statistics of hashtags in two data sets.

Data set	Single-word	Multi-word
Daily-Life		
Train	4,623	8,204
Vali	576	1,034
Test	601	1,009
Election-Trec		
Train	14,686	9,524
Vali	1,793	1,234
Test	1,821	1,206

Note: Train, Dev, and Test denotes training, development, and test set, respectively. Single-word: number of target posts with the "single-word" hashtag. Multi-word: number of target posts with the "multi-word" hashtag.

BaseLine Model

Conditional Random Field. CRF (Lafferty, McCallum, & Pereira, 2001) is a type of discriminative undirected probabilistic graphical model and can process a sequence-labeling task. Keyphrase extraction is regarded as a sequence labeling task when using the CRF algorithm.

Word Embedding Model Without Conversation Context Encoders. This model only has the keyphrase tagger module. Moreover, it merely considers the word embedding information from pretrained word embedding. To analyze whether conversation context encoders and character-level word embeddings can enhance the performance of keyphrase extraction on social media, this article utilized this model as the baseline.

Word Embedding Models With Nonhierarchical Conversation Context Encoders. This model consists of two modules: a conversation context encoder and a keyphrase tagger. The conversation context encoder is a nonhierarchical encoder (Zhang et al., 2018), which merely captures word-level features from a conversation context. Formally, given the conversation context x_i^r, each word $x_{i,w}^r$ is represented as a vector $v_{i,w}^r$ by an embedding layer. Then the $v_{i,w}^r$ is fed into the word-level encoder and the output e_i^r is obtained directly. Moreover, it merely considers word embedding information from pretrained word embedding in both the conversation context encoder and the keyphrase tagger. This kind of model includes word embedding model with BiLSTM nonhierarchical encoders (BiLSTM), word embedding model with attention-based BiLSTM nonhierarchical encoders (Att (BiLSTM)), and word embedding model with memory network nonhierarchical encoders (MemNN).

Our Model

Word Embedding Model With Hierarchical Conversation Context Encoder. This model consists of two modules: a hierarchical conversation context encoder and a keyphrase tagger. The hierarchical conversation context encoder captures word-level features and sentence-level features from a conversation context synchronously. In addition, it only considers word embedding information in both conversation context encoders and keyphrase taggers. This model is utilized to prove the effectivity of the hierarchical conversation context encoder. This kind of model includes word embedding

DOI: 10.1002/asi

models with BiLSTM hierarchical encoders (H-BiLSTM), word embedding models with attention-based BiLTSM hierarchical encoders (H-Att (BiLSTM)), and word embedding models with memory network hierarchical encoders (H-MemNN).

Character-Level Word Embedding Model With Nonhierarchical Conversation Context Encoder. This model consists of two modules: a nonhierarchical conversation context encoder and a keyphrase tagger. The nonhierarchical conversation context encoder merely captures word-level features from conversation context. In both conversation context encoder and keyphrase tagger, we use character-level word embedding to learn morphological representations by capturing character-level features. This model is utilized to analyze whether character-level word embedding would promote the performance of keyphrase extraction. Three models in this type are proposed: character-level word embedding models with BiLSTM nonhierarchical encoders (BiLSTM-C), character-level word embedding models with attention-based BiLSTM nonhierarchical encoders (Att (BiLSTM)-C), and character-level word embedding models with memory network nonhierarchical encoders (MemNN-C).

Character-Level Word Embedding Model With Hierarchical Conversation Context Encoder. This model consists of two modules: a hierarchical conversation context encoder and a keyphrase tagger. The hierarchical conversation context encoder has the ability to capture word-level features and message-level features from a conversation context. In both hierarchical conversation context encoders and keyphrase taggers, we utilize character-level word embedding to capture character-level features from conversation contexts and target posts. Hence, this model captures character-level, word-level, and message-level features from conversation context to assist keyphrase extraction from target posts. It is utilized to prove the effectivity of the combination of hierarchical structure encoders and character-level word embedding. We propose three models in this type: character-level word embedding models with BiLSTM hierarchical encoders (H-BiLSTM-C), character-level word embedding models with attention-based BiLSTM hierarchical encoders (H-Att (BiLSTM)-C), and character-level word embedding models with memory network hierarchical encoders (H-MemNN-C).

Results

This section presents the quantitative and qualitative results of our neural keyphrase extraction models. Among them, this article first presents the results of the proposed neural network keyphrase extraction models. Then we compare the performances of three encoders: BiLSTM hierarchical encoders, attention-based BiLSTM hierarchical encoders, and memory network hierarchical encoders. Moreover, we further state the functions of the hierarchical conversation context encoders and character-level word embeddings, respectively. In the end, we discuss the performances of models in the situations when the target posts do not have conversation contexts.

TABLE 6. The P, R, F1 scores(%) of CRF keyphrase extraction model on Daily-Life and Election-Trec data set.

Data set	P	R	F1
Daily-Life	79.48	53.66	64.07
Election-Trec	80.66	45.69	58.34

Overall Analysis

In this section we compare our proposed models with three baseline models: CRF, word embedding model without conversation context encoders, and word embedding model with nonhierarchical conversation context encoders. The results of the CRF baseline models are reported in Table 6. The results of our proposed models and other baseline models are reported in Table 7. From Tables 6 and 7, we have these observations.

Neural Networks Keyphrase Taggers Yield Better F1 Than CRF. As shown in Table 7, the F1 scores of neural networks keyphrase models fluctuate between 58.90% and 72.37% in the Election-Trec data set. In the Daily-Life data set, the F1 scores fluctuate between 66.83% and 74.56%. Those F1 scores are higher than those of the CRF models shown in Table 6. These observations indicate that the neural network- based models are better choices for extracting keyphrase from tweets.

The Conversation Context Is Useful For Keyphrase Extraction. As the results reported in Table 7, models with conversation context encoders uniformly outperform those without conversation context encoders. This indicates that capturing features from a conversation context by conversation context encoders can alleviate the data sparsity problem by enriching the context of target posts.

Models With Hierarchical Encoders Have Better Performance Than Those With Nonhierarchical Encoders. As shown in Table 7, models with hierarchical conversation context encoders, such as H-BiLSTM, H-Att (BiLSTM), and H-MemNN, have better performances than nonhierarchical models, for example, BiLSTM, Att (BiLSTM), and MemNN. Hence, jointly capturing word-level and message-level features from a conversation context is effective.

Character-Level Features Is Useful for Keyphrase Extraction on Social Media. As shown in Table 7, models with character-level word embeddings, such as BiLSTM-C, Att (BiLSTM)-C, and MemNN-C, yield better F1 scores than word embedding based baseline models; for example, BiLSTM, Att (BiLSTM), and MemNN. Thus, capturing character-level features from words can alleviate the OOV problem and thereby improve the performance of keyphrase extraction.

The models Jointly Capturing Character-Level, Word-Level and Message-Level Features Yield the Best F1 Scores. The models achieve the best F1 scores when they consist of the character-level word embedding and the hierarchical conversation context encoder, such as H-BiLSTM-C, H-Att (BiLSTM)-C, and H-MemNN-C. This is due to that these models synchronously capture character-level features to alleviate

DOI: 10.1002/asi

TABLE 7. Comparisons of the P, R, F1 scores (%) of various neural network models on Daily-Life data set and Election-Trec data set.

	Daily-Life						Election-Trec					
	Single-layer			Joint-layer			Single-layer			Joint-layer		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Baseline												
No encoder	65.62	68.07	66.83	74.41	67.10	70.57	71.20	50.23	58.90	68.57	63.08	65.71
BiLSTM	74.57	64.47	69.15	74.43	68.70	71.45	74.54	53.75	62.46	72.40	60.50	65.92
Att (BiLSTM)	76.66	62.98	69.15	75.41	69.83	**72.51**	73.51	55.94	63.53	70.58	62.82	66.47
MemNN	72.27	67.67	69.89	74.15	69.83	71.92	67.65	60.63	63.94	70.23	63.41	66.64
Our model												
H-BiLSTM	73.70	66.83	70.10	73.37	70.00	71.65	68.88	63.89	66.29	71.81	64.64	68.03
H-Att(BiLSTM)	77.36	66.23	**71.33**	76.90	70.43	73.52	69.81	65.69	**67.69**	74.96	63.37	68.68
H-MemNN	74.39	66.17	70.04	75.66	72.12	**73.85**	75.78	60.83	67.49	72.23	66.93	**69.47**
BiLSTM-C	73.87	67.79	70.70	76.53	70.73	73.52	68.46	65.49	66.94	72.33	63.34	67.54
Att(BiLSTM)-C	74.58	70.01	**72.23**	78.51	69.41	73.68	70.63	64.38	67.36	75.65	62.52	68.46
MemNN-C	75.76	67.43	71.35	78.35	70.01	**73.94**	71.18	64.84	**67.86**	75.58	64.91	**69.84**
H-BiLSTM-C	78.46	66.77	72.14	77.89	70.91	74.24	70.37	64.61	67.36	74.93	64.74	69.47
H-Att(BiLSTM)-C	75.77	69.35	**72.42**	78.84	70.55	74.47	77.78	62.69	**69.42**	76.42	68.72	**72.37**
H-MemNN-C	76.48	68.62	72.33	77.96	71.45	**74.56**	72.84	64.48	68.41	76.09	66.86	71.18

Note: The left half reports results of single-layer taggers. The right half reports results of joint-layer taggers. Each column: results of the same tagger with different encoders. Each row: results of different taggers with the same encoder. No Encoder: word embedding model without conversation context encoder.

the OOV problem and capture word-level and message-level features from a conversation context to enrich the context of target posts.

Comparison of Three Hierarchical Encoders

To compare the performance of models with three hierarchical encoders: BiLSTM hierarchical encoders, attention-based BiLSTM hierarchical encoders, and memory network hierarchical encoders, this article analyzes the F1 scores of H-BiLSTM, H-Att(BiLSTM), and H-MemNN presented in Table 7. Following are our observations.

Highlighting Salient Words and Messages in Conversation Context Help With Capturing the Main Features of the Conversation Context. As Table 7 shows, H-Att (BiLSTM) and H-MemNN yield better F1 scores than H-BiLSTM. H-Att (BiLSTM) and H-MemNN have capabilities to indicate salient words and salient messages from a conversation context that describe the main focus of the conversation context, while H-BiLSTM regards all the words and messages equally important in a conversation context.

TABLE 8. Outputs of joint-layer keyphrase taggers combined with various content encoders give the example illustrated in Table 1.

	Extracted keyphrase
Gold-standard	*Steelers*
No encoder	*NULL*
Our model	
MemNN	*Super Bowl*
MemNN-C	*Super Bowl*
H-MemNN	*Steelers*
H-MemNN-C	*Steelers*

Note: "NULL": No encoder models produce any keyphrases.

Models With Memory Network Hierarchical Conversation Context Encoders Yield Approximately the Best Results. As shown in Table 7, H-MemNN with a joint layer keyphrase tagger yield a better performance than H-BiLSTM and H-Att (BiLSTM) models in both the Daily-Life and Election-Trec data set. It originates from that memory networks are designed to allow the encoders to calculate the similarity between the conversation context and the target post, which can exploit the affinity of conversation contexts and target posts.

Qualitative Analysis of Hierarchical Encoders

To qualitatively analyze why hierarchical conversation context encoders generally perform better in comparison, we conducted a case study on the simple instance in Table 8. Recall that the keyphrase should be "Steelers." We compare the keyphrases produced by a range of joint-layer keyphrase taggers with memory network conversation context encoders, given in Table 8. Interestingly, baseline models without encoders do not extract any keyphrase. Models with memory network nonhierarchical encoders (MemNN and MemNN-C) extract the phrase "Super Bowl." This may due to that these models tend to identify words that frequently appear in the conversation context, for example, "Packers," "Super Bowl," which are relevant to the topic of the "Super Bowl" national football competition. Of all the keyphrase taggers, only two of the models with the hierarchical encoders identified the correct keyphrase.

To explain the reason why keyphrase taggers with memory network hierarchical encoders (H-MemNN and H-MemNN-C) have the ability to extract accurate keyphrases from target posts, this article utilizes a heatmap to visualize the similarity matrix of words in this target post and messages in the corresponding conversation context, from which we can analyze the information captured by those encoders in detail. As shown in Figure 5,

DOI: 10.1002/asi

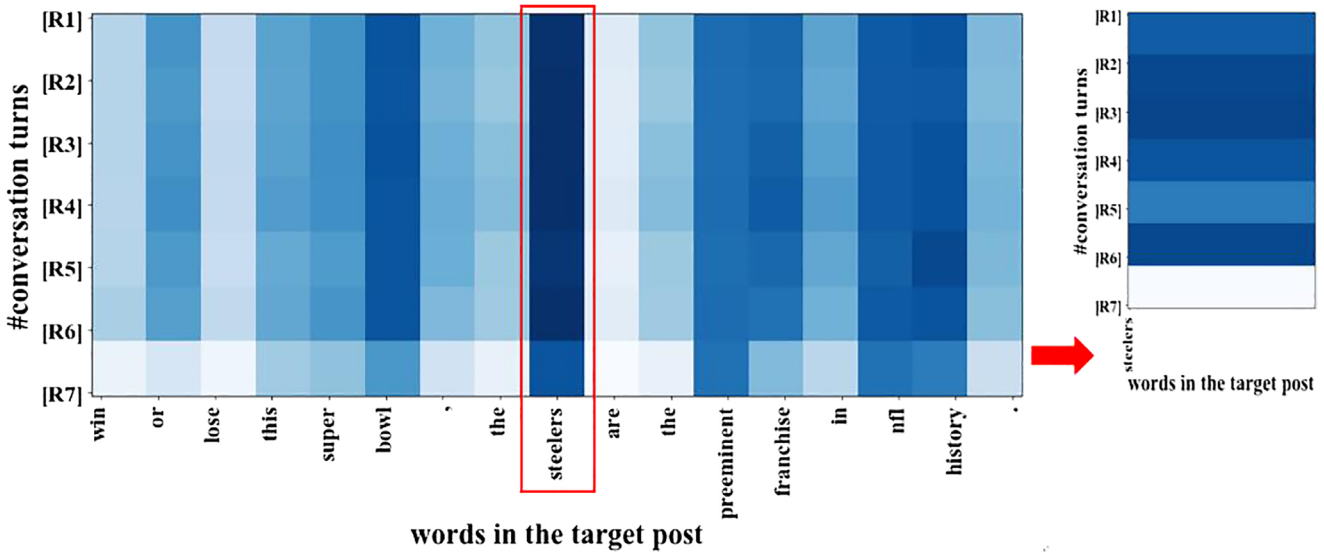

FIG. 5. The heatmap of the context representation generated by memory network hierarchical encoders. The horizontal axis refers to words in the target post, while the vertical axis refers to messages in the conversation context. The red box indicates the keyphrase to be detected. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 9. The P, R, F1 scores (%) of keyphrase taggers with word embedding and character-level word embedding on the subdata set that hashtags of the target posts contains out-of-vocabulary words.

	Daily-Life			Election-Trec		
	Joint-layer Taggers			Joint-layer Taggers		
	P	R	F1	P	R	F1
H-Att (BiLSTM)	73.78	70.07	71.88	71.48	59.66	65.03
H-Att (BiLSTM)-C	75.21	69.98	72.50	72.49	65.93	69.06
H-MemNN	75.79	69.50	72.51	69.50	63.33	66.27
H-MemNN-C	75.75	70.56	73.06	72.41	64.38	68.16

TABLE 10. The example that the hashtag in the tweet contains rare words.

Target Post	*In what may be the last Boston Marathon of her illustrious career is ready to leave it all on the line!*
Gold-standard Keyphrase	*Boston Marathon*
Context Encoder	
H-Att (BiLSTM)	*Boston*
H-Att (BiLSTM)-C	*Boston Marathon*

TABLE 11. The example that the hashtag in the tweet contains abbreviations.

Target Post	*30 min to hcsm! If you are interested in health communications & social media, you are welcome to join. Just add the hashtag to your tweets!*
Gold-standard Keyphrase	*hcsm*
Context Encoder	
H-Att (BiLSTM)	*NULL*
H-Att (BiLSTM)-C	*hcsm*

TABLE 12. The example that the hashtag in the tweet contains self-invented words.

Target Post	*MEN Did you get Big Sur animalcollective tickets?*
Gold-standard Keyphrase	*animalcollective*
Context Encoder	
H-Att (BiLSTM)	*NULL*
H-Att (BiLSTM)-C	*animalcollective*

it is noticeable that the word "steelers" is more relevant with all the turns in the conversation context than other words since a darker color represents a higher similarity in the heatmap. Moreover, it is obvious that the similarity values of a word are different with various messages. To analyze the similarity values of the word "steelers" in detail, another heatmap (The max and min values of this heatmap are adjusted to magnify the difference) is utilized to visualize the similarities between the word "steelers" and messages in the conversation context. As shown in the right side of Figure 5, the message [R3] is the most relevant to the word "steelers." This observation is the same with the analyzation in section 0 that the message [R3] contains the keyword "steelers."

DOI: 10.1002/asi

TABLE 13. The P, R, F1 scores (%) of keyphrase taggers with word embedding and character-level word embedding on the subdata set that keyphrases of the target posts contains out-of-vocabulary words.

	Daily-Life						Election-Trec					
	Single-layer			Joint-layer			Single-layer			Joint-layer		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BiLSTM	77.36	61.00	68.21	73.45	68.20	70.73	73.93	53.58	62.13	70.30	62.00	65.89
Att(BiLSTM)	73.60	64.18	68.57	75.03	66.83	70.69	74.54	53.75	62.46	68.29	63.99	66.07
MemNN	73.65	65.10	**69.11**	74.90	67.25	**70.87**	70.88	56.46	**62.85**	69.83	63.34	**66.43**
BiLSTM-C	73.40	67.51	70.34	75.84	67.68	71.52	75.86	59.46	66.67	72.31	61.68	66.57
Att(BiLSTM)-C	75.65	68.51	**71.90**	75.30	70.55	72.85	68.90	65.50	67.15	74.57	62.56	68.04
MemNN-C	75.28	66.73	70.74	76.61	70.00	**73.16**	74.01	62.13	**67.55**	72.46	66.18	**69.18**

The Impact of Character Embedding on OOV Words

To further demonstrate the efficiency of character-level word embeddings, this article analyzes the performance of models on the sub dataset that the keyphrase in the target post contains OOV words. In this article, OOV words are the words that cannot be found in the pretrained word embeddings. For a "multi-word" keyphrase, if it contains an OOV word, it is defined as the OOV keyphrase. For a "single-word" keyphrase, if it is an OOV word, it will be defined as the OOV keyphrase. There are 1,009 tweets and 1,211 tweets whose keyphrases is OOV keyphrases in the Daily-Life test data set and Election-Trec test data set, respectively. Table 9 shows the comparison of models with character-level word embeddings and those with word embeddings on two data sets. As shown in Table 9, models with character-level word embeddings yield better F1 scores than those with word embeddings only. This observation indicates that character-level word embeddings have the ability to alleviate the severe OOV problem in the keyphrase extraction task on social media.

Rare words, abbreviations, and self-invented words are common problems in the user-generated content on social media. To analyze whether character-level word embedding can solve those three problems efficiently, Tables 10–12 present instances of rare words, abbreviations, and self-invented words, respectively. With the character-level word embedding, the keyphrase taggers identify the accurate words that are same with gold-standard keyphrases, while the taggers with word embeddings merely extract the incorrect words. Those observations confirm that the character-level word embedding has capabilities to alleviate various OOV problems.

Experiment on the Data Set Without Conversation Context

Although we have shown in the previous section that conversation context is useful for training effective models for keyphrase extraction on microblog posts, it is necessary to consider that conversation context might be unavailable to some microblog posts that do not spark any repost or reply message. Under this circumstance, the models trained on messages with conversation context might be affected in extracting the keyphrases for messages without conversation context. To study whether conversation context is critical in the testing process, we assume that the conversations are only available for training data, while all the target posts in the test set have no context to be leveraged. To this end, we apply the models trained for the experiment previously on the test posts without using their conversation context. In prediction, context encoders of the trained model stake the target posts instead of conversations input, which do not have hierarchical structures. Hence, we do not conduct the experiments on models with hierarchical encoders. The results are reported in Table 13, where models with conversation context encoders yield better F1 scores than their counterparts without such encoders, whether providing conversation to test data or not. This observation indicates that encoding conversations in training data helps in learning effective keyphrase extraction models, which is beneficial to detect keyphrases in a microblog post with or without its conversation context. In addition, by comparing Table 13 with Table 7, we find that, for each model with a context encoder, higher F1 scores are observed when the conversation context is used in the testing process. This observation confirms that the conversation context of target posts helps in indicating keyphrases in prediction.

Conclusion

This article proposes a neural keyphrase extraction framework for microblog posts. The framework consists of two modules: a conversation context encoder and a keyphrase tagger. Conversation context encoders are utilized to encode conversation context to help keyphrase taggers indicate salient phrases. The keyphrase tagger is employed to extract keyphrases from target posts. To leverage the structure of conversation, the hierarchical encoder is employed to learn the word-level and message-level information from a conversation context. To alleviate the OOV problem in user-generated content on social media platforms, we utilized the character-level word embedding to capture both character-level and word-level features in both conversation context encoders and keyphrase taggers. The experimental results show that the keyphrase extraction models with hierarchical encoders and character-level word embeddings yield better performance than other models.

DOI: 10.1002/asi

In this study, keyphrases were extracted from target posts and the conversation context was utilized to assist keyphrase extraction. It should be acknowledged that conversation context contains information that is related to multiple topics. Extracting keyphrases on different topics from the conversation context will broaden the content coverage of keyphrases. Thus, in future work, we will explore models to effectively identify topics from the conversation context and thereby extract keyphrases of various topics from the conversation context and the target post synchronously.

The models proposed in this article can be extended to other tasks. First, the model may be helpful for extracting keyphrases from academic articles. Generally, academic articles consist of titles, abstracts, texts, and references. In the references, titles provided are likely to contain information related to abstracts and texts. Intuitively, encoding titles of references by conversation context encoders may assist the keyphrase taggers to indicate keyphrases from the abstract and text. Moreover, the models can also be utilized in a range of information extraction tasks on social media, such as name entity recognition.

Acknowledgments

This work was supported by Major Projects of National Social Science Fund (No.16ZDA224).

References

Ahmad, K., Pogorelov, K., Riegler, M., Conci, N., & Halvorsen, P. (2019). Social media and satellites. Multimedia Tools and Applications, 78(3), 2837–2875.

Bellaachia, A. & Al-Dhelaan, M. (2012). NE-Rank: A novel graph-based keyphrase extraction in Twitter. In Proceedings of the IEEE/WIC/ACM International Conferences on Web Intelligence, WI (pp. 372–379), Macau, China: IEEE Computer Society.

Chang, Y., Wang, X., Mei, Q., & Liu, Y. (2013). Towards Twitter context summarization with user influence models. In Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, WSDM (pp. 527–536), Rome, Italy: ACM.

Chew, C., & Eysenbach, G. (2010). Pandemics in the age of Twitter: Content analysis of tweets during the 2009 H1N1 outbreak. PLoS One, 5 (11), e14118.

Choi, J., Croft, W.B., & Kim, J. (2012). Quality models for microblog retrieval. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM (pp. 1834–1838), Maui, HI, USA: ACM.

Derczynski, L., Maynard, D., Rizzo, G., van Erp, M., Gorrell, G., Troncy, R., … Bontcheva, K. (2015). Analysis of named entity recognition and linking for tweets. Information Processing & Management, 51 (2), 32–49.

Efron, M. (2010). Hashtag retrieval in a microblogging environment. In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR (pp. 787–788), Geneva, Switzerland: ACM.

Ercan, G., & Cicekli, I. (2007). Using lexical chains for keyword extraction. Information Processing & Management, 43(6), 1705–1714.

Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint, arXiv/1308.0850.*

Jebbara, S. & Cimiano, P. (2017). Improving opinion-target extraction with character-level word embeddings. In Proceedings of the First Workshop on Subword and Character Level Models in NLP (pp. 159–167), Copenhagen, Denmark: Association for Computational Linguistics.

Jones, K.S. (2004). A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 60(5), 493–502.

Jones, S., & Paynter, G.W. (2002). Automatic extraction of document keyphrases for use in digital libraries: Evaluation and applications. Journal of the Association for Information Science and Technology, 53(8), 653–677.

Kapanova, K.G. & Fidanova, S. (2017). Generalized nets: A new approach to model a hashtag linguistic network on Twitter. In Proceedings of the Annual Meeting of the Bulgarian Section of SIAM, BGSIAM (pp. 211–221), Bulgaria: Springer.

Kireyev, K. (2009). Semantic-based estimation of term informativeness. In Proceedings of the Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, NAACL (pp. 530–538), Boulder, Colorado, USA: The Association for Computational Linguistics.

Kuru, O., Can, O.A., & Yuret, D. (2016). Charner: Character-level named entity recognition. In Proceedings of the 26th International Conference on Computational Linguistics, COLING (pp. 911–921), Osaka, Japan: The Association for Computational Linguistics.

Lafferty, J.D., McCallum, A., & Pereira, F.C.N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the Eighteenth International Conference on Machine Learning, ICML (pp. 282–289), Williamstown, MA, USA: Morgan Kaufmann.

Li, J., Gao, W., Wei, Z., Peng, B., & Wong, K. (2015). Using content-level structures for summarizing microblog repost trees. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP (pp. 2168–2178), Lisbon, Portugal: The Association for Computational Linguistics.

Li, J., Liao, M., Gao, W., He, Y., & Wong, K. (2016). Topic extraction from microblog posts using conversation structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL (pp. 2114–2123), Berlin, Germany: The Association for Computer Linguistics.

Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP, August, A Meeting of SIGDAT, A Special Interest Group of the ACL (pp. 257–266), Singapore: The Association for Computer Linguistics.

Martinez-Romo, J., Araujo, L., & Fernandez, A.D. (2016). Semgraph: Extracting keyphrases following a novel semantic graph-based approach. Journal of the Association for Information Science and Technology, 67(1), 71–82.

Marujo, L., Ling, W., Trancoso, I., Dyer, C., Black, A.W., Gershman, A., … Carbonell, J.G. (2015). Automatic keyword extraction on Twitter. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL (pp. 637–643), Beijing, China: The Association for Computer Linguistics.

Medelyan, O., & Witten, I.H. (2014). Domain-independent automatic keyphrase indexing with small training sets. Journal of the Association for Information Science and Technology, 59(7), 1026–1040.

Mihalcea, R. & Tarau, P. (2004). TextRank: Bringing order into text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, EMNLP, A Meeting of SIGDAT, A Special Interest Group of the ACL, Held in Conjunction with ACL (pp. 404–411), Barcelona, Spain: The Association for Computer Linguistics.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems, NIPS (pp. 3111–3119), Lake Tahoe, Nevada, United States: ACM.

Ribeiro, R., Gershman, A., de Matos, D.M., Neto, J.P., & Carbonell, J.G. (2017). Event-based summarization using a centrality-as-relevance model. Knowledge & Information Systems, 50(3), 945–968.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. Information Processing & Management, 24(5), 513–523.

Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.C., & Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (pp. 3776–3784), Phoenix, Arizona, USA: AAAI Press.

Small, T.A. (2011). What the hashtag? Information Communication and Society, 14(6), 872–895.

Tang, J., Li, J., Wang, K., & Cai, Y. (2004). Loss minimization based keyword distillation. In Proceedings of the Advanced Web Technologies and Applications, 6th Asia-Pacific Web Conference, APWeb (pp. 572–577), Hangzhou, China: Springer.

Turney, P.D. (2000). Learning algorithms for keyphrase extraction. Information Retrieval, 2(4), 303–336.

Wan, X. & Xiao, J. (2008). Single document keyphrase extraction using neighborhood knowledge. In Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI (pp. 855–860), Chicago, Illinois, USA: AAAI Press.

Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., & Nevill-Manning, C.G. (1999). KEA: Practical automatic keyphrase extraction. In Proceedings of the Fourth ACM Conference on Digital Libraries (pp. 254–255), Chicago, Illinois, USA: AAAI Press.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A.J., & Hovy, E.H. (2016). Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL (pp. 1480–1489), San Diego California, USA: The Association for Computational Linguistics.

Yu, F., Xuan, H., & Zheng, D. (2012). Key-phrase extraction based on a combination of CRF model with document structure. In Proceedings of the Eighth International Conference on Computational Intelligence and Security, CIS (pp. 406–410), Guangzhou, China: IEEE Computer Society.

Zeng, X., Li, J., Wang, L., Beauchamp, N., Shugars, S., & Wong, K. (2018). Microblog conversation recommendation via joint modeling of topics and discourse. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL (pp. 375–385), New Orleans, Louisiana, USA: Association for Computational Linguistics.

Zhang, C., Wang, H., Liu, Y., Wu, D., Liao, Y., & Wang, B. (2008). Automatic keyword extraction from documents using conditional random fields. The Journal of Computer Information Systems, 4(3), 1169–1180.

Zhang, Q., Wang, Y., Gong, Y., & Huang, X. (2016). Keyphrase extraction using deep recurrent neural networks on Twitter. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP (pp. 836–845), Austin, Texas, USA: The Association for Computational Linguistics.

Zhang, Y., Li, J., Song, Y., & Zhang, C. (2018). Encoding conversation context for neural keyphrase extraction from microblog posts. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL (pp. 1676–1686), New Orleans, Louisiana, USA: Association for Computational Linguistics.

Zhao, W.X., Jiang, J., He, J., Song, Y., Achananuparp, P., Lim, E., & Li, X. (2011). Topical keyphrase extraction from Twitter. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL (pp. 379–388), Portland, Oregon, USA: The Association for Computer Linguistics.

Zheng, X., Liu, G., Wu, J., & Tan, Y. (2018). Big data would not lie: Prediction of the 2016 Taiwan election via online heterogeneous information. EPJ Data Science, 7(1), 32.

DOI: 10.1002/asi