Topic-Aware Neural Keyphrase Generation for Social Media Language

Yue Wang[1*] **Jing Li**[2†] **Hou Pong Chan**[1] **Irwin King**[1] **Michael R. Lyu**[1] **Shuming Shi**[2]
[1]Department of Computer Science and Engineering
The Chinese University of Hong Kong, HKSAR, China
[2]Tencent AI Lab, Shenzhen, China
[1]{yuewang, hpchan, king,lyu}@cse.cuhk.edu.hk
[2]{ameliajli,shumingshi}@tencent.com

Abstract

A huge volume of user-generated content is daily produced on social media. To facilitate automatic language understanding, we study keyphrase prediction, distilling salient information from massive posts. While most existing methods *extract* words from source posts to form keyphrases, we propose a sequence-to-sequence (seq2seq) based neural *keyphrase generation* framework, enabling absent keyphrases to be created. Moreover, our model, being *topic-aware*, allows joint modeling of corpus-level latent topic representations, which helps alleviate the data sparsity that widely exhibited in social media language. Experiments on three datasets collected from English and Chinese social media platforms show that our model significantly outperforms both extraction and generation models that do not exploit latent topics.[1] Further discussions show that our model learns meaningful topics, which interprets its superiority in social media keyphrase generation.

1 Introduction

As social media continues its worldwide expansion, the last decade has witnessed the revolution of interpersonal communication. While empowering individuals with richer and fresher information, the flourish of social media also results in millions of posts generated on a daily basis. Facing a sheer quantity of texts, language understanding has become a daunting task for human beings. Under this circumstance, there exists a pressing need for developing automatic systems capable of absorbing massive social media texts and figuring out what is important.

Source post with keyphrase "*super bowl*":
$[S]$: Somewhere, a wife that is not paying attention to the *game*, says "I want the *team* in *yellow pants* to *win*."
Relevant tweets:
$[T_1]$: I been a *steelers fan* way before *black* & *yellow* and this *super bowl*!
$[T_2]$: I will bet you the *team* with *yellow pants wins*.
$[T_3]$: Wiz Khalifa song '*black* and *yellow*" to spur the *pittsburgh steelers* and Lil Wayne is to sing "*green* and *yellow*' for the *packers*.

Table 1: Sample tweets tagged with "*super bowl*" as their keyphrases. *Blue and italic words* can indicate the topic of super bowl.

In this work, we study the prediction of **keyphrases**, generally formed with words or phrases reflecting main topics conveyed in input texts (Zhang et al., 2018). Particularly, we focus on producing keyphrases for social media language, proven to be beneficial to a broad range of applications, such as instant detection of trending events (Weng and Lee, 2011), summarizing public opinions (Meng et al., 2012), analyzing social behavior (Ruths and Pfeffer, 2014), and so forth.

In spite of the substantial efforts made in social media keyphrase identification, most progress to date has focused on *extracting* words or phrases from source posts, thus failing to yield keyphrases containing absent words (i.e., words do not appear in the post). Such cases are indeed prominent on social media, mostly attributed to the informal writing styles of users therein. For example, Table 1 shows a tweet S tagged with keyphrase "*super bowl*" by its author, though neither "*super*" nor "*bowl*" appears in it.[2] In our work, distinguishing from previous studies, we approach social media keyphrase prediction with a *sequence*

[*]This work was partially done when Yue Wang was an intern at Tencent AI Lab.

[†]Jing Li is the corresponding author.

[1]Our data and code are publicly released in https://github.com/yuewang-cuhk/TAKG

[2]Following common practice (Zhang et al., 2016, 2018), we consider author-annotated hashtags as tweets' keyphrases.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2516–2526
Florence, Italy, July 28 - August 2, 2019.

generation framework, which is able to create absent keyphrases beyond source posts.

Our work is built on the success of deep keyphrase generation models based on neural sequence-to-sequence (seq2seq) framework (Meng et al., 2017). However, existing models, though effective on well-edited documents (e.g., scientific articles), will inevitably encounter the data sparsity issue when adapted to social media. It is essentially due to the informal and colloquial nature of social media language, which results in limited features available in the noisy data. For instance, only given the words in S (Table 1), it is difficult to figure out why "*super bowl*" is its keyphrase. However, by looking at tweets T_1 to T_3, we can see "*yellow pants*" is relevant to "*steelers*", a *super bowl* team. As "*yellow*" and "*pants*" widely appear in tweets tagged with "*super bowl*', it becomes possible to identify "*super bowl*" as S's keyphrase.

Here we propose a novel *topic-aware neural keyphrase generation model* that leverages latent topics to enrich useful features. Our model is able to identify topic words, naturally indicative of keyphrases, via exploring post-level word co-occurrence patterns, such as "*yellow*" and "*pants*" in S. Previous work have shown that corpus-level latent topics can effectively alleviate data sparsity in other tasks (Zeng et al., 2018; Li et al., 2018). The effects of latent topics, nevertheless, have never been explored in existing keyphrase generation research, particularly in the social media domain. To the best of our knowledge, *our work is the first to study the benefit of leveraging latent topics on social media keyphrase generation*. Also, our model, taking advantage of the recent advance of neural topic models (Miao et al., 2017), enables end-to-end training of latent topic modeling and keyphrase generation.

We experiment on three newly constructed social media datasets. Two are from English platform Twitter and StackExchange, and the other from Chinese microblog Weibo. The comparison results over both extraction and generation methods show that our model can better produce keyphrases, significantly outperforming all the comparison models without exploiting latent topics. For example, on Weibo dataset, our model achieves 34.99% F1@1 compared with 32.01% yielded by a state-of-the-art keyphrase generation model (Meng et al., 2017). We also probe into our outputs and find that meaningful latent topics can be learned, which can usefully indicate keyphrases. At last, a preliminary study on scientific articles shows that latent topics work better on text genres with informal language style.

2 Related Work

Our work is mainly in the line of two areas: keyphrase prediction and topic modeling. We introduce them in turn below.

Keyphrase Prediction. Most previous efforts on this task adopt supervised or unsupervised approaches based on *extraction* — words or phrases selected from source documents to form keyphrases. Supervised methods are mostly based on sequence tagging (Zhang et al., 2016; Gollapalli et al., 2017) or binary classification using various features (Witten et al., 1999; Medelyan et al., 2009). For unsupervised methods, they are built on diverse algorithms, including graph ranking (Mihalcea and Tarau, 2004; Wan and Xiao, 2008), document clustering (Liu et al., 2009, 2010), and statistical models like TF-IDF (Salton and McGill, 1986).

Our work is especially in the line of social media keyphrase prediction, where extractive approaches are widely employed (Zhang et al., 2016, 2018). On the contrary, we predict keyphrases in a *sequence generation* manner, allowing the creation of absent keyphrases. Our work is inspired by seq2seq-based keyphrase generation models (Meng et al., 2017; Chen et al., 2018, 2019a,b), which are originally designed for scientific articles. However, their performance will be inevitably compromised when directly applied to social media language owing to the data sparsity problem. Recently, Wang et al. (2019) propose a microblog hashtag generation framework, which explicitly enriches context with user responses. Different from them, we propose to leverage corpus-level latent topic representations, which can be learned without requiring external data. Its potential usefulness on keyphrase generation has been ignored in previous research and will be extensively studied here.

Topic Modeling. Our work is closely related with topic models that discover latent topics from word co-occurrence in document level. They are commonly in the fashion of latent Dirichlet allocation (LDA) based on Bayesian graphical models

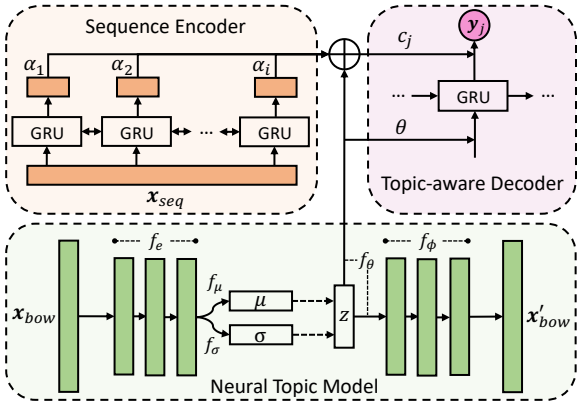


Figure 1: Our topic-aware neural keyphrase generation framework (§3).

(Blei et al., 2003). These models, however, rely on the expertise involvement to customize model inference algorithms. Our framework exploits the recently proposed neural topic models (Miao et al., 2017; Srivastava and Sutton, 2017) to infer latent topics, which facilitate end-to-end training with other neural models and do not require model-specific derivation. It has proven useful for citation recommendation (Bai et al., 2018) and conversation understanding (Zeng et al., 2019). In particular, Zeng et al. (2018) propose to jointly train topic models and short text classification, which cannot fit our scenario due to the large diversity of the keyphrases (Wang et al., 2019). Different from them, our latent topics are learned together with language generation, whose effects on keyphrase generation have never been explored before in existing work.

3 Topic-Aware Neural Keyphrase Generation Model

In this section, we describe our framework that leverages latent topics in neural keyphrase generation. Figure 1 shows our overall architecture consisting of two modules — a neural topic model for exploring latent topics (§3.1) and a seq2seq-based model for keyphrase generation (§3.2).

Formally, given a collection $\mathcal{C}$ with $|C|$ social media posts $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|C|}\}$ as input, we process each post $\mathbf{x}$ into bag-of-words (BoW) term vector $\mathbf{x}_{bow}$ and word index sequence vector $\mathbf{x}_{seq}$. $\mathbf{x}_{bow}$ is a V-dim vector over the vocabulary (V being the vocabulary size). It is fed into the neural topic model following the BoW assumption (Miao et al., 2017). $\mathbf{x}_{seq}$ serves as the input for the seq2seq-based keyphrase generation model.

Below we first introduce our two modules and then describe how they are jointly trained (§3.3).

3.1 Neural Topic Model

Our neural topic model (NTM) module is inspired by Miao et al. (2017) based on variational auto-encoder (Kingma and Welling, 2013), which consists of an encoder and a decoder to resemble the data reconstruction process.

Specifically, the input $\mathbf{x}_{bow}$ is first encoded into a continuous latent variable $\mathbf{z}$ (representing $\mathbf{x}$'s topic) by a BoW encoder. Then the BoW decoder, conditioned on $\mathbf{z}$, attempts to reconstruct $\mathbf{x}$ and outputs a BoW vector $\mathbf{x}'_{bow}$. Particularly, the decoder simulates topic model's generation process. We then describe their division of labor.

BoW Encoder. The BoW encoder is responsible for estimating prior variables μ and σ, which will be used to induce intermediate topic representation $\mathbf{z}$. We adopt the following formula:

$$\mu = f_\mu(f_e(\mathbf{x}_{bow})),\ log\,\sigma = f_\sigma(f_e(\mathbf{x}_{bow})), \quad (1)$$

where $f_*(\cdot)$ is a neural perceptron with an ReLU-activated function following Zeng et al. (2018).

BoW Decoder. Analogous to LDA-style topic models, it is assumed that there are K topics underlying the given corpus $\mathcal{C}$. Each topic k is represented with a topic-word distribution ϕ_k over the vocabulary, and each post $\mathbf{x} \in \mathcal{C}$ has a topic mixture denoted by θ, a K-dim distributional vector. Specifically in neural topic model, θ is constructed by Gaussian softmax (Miao et al., 2017). The decoder hence takes the following steps to simulate how each post $\mathbf{x}$ is generated:

- Draw latent topic variable $\mathbf{z} \sim \mathcal{N}(\mu, \sigma^2)$
- Topic mixture $\theta = softmax(f_\theta(\mathbf{z}))$
- For each word $w \in \mathbf{x}$
  - Draw $w \sim softmax(f_\phi(\theta))$

Here $f_*(\cdot)$ is also a ReLU-activated neural perceptron for inputs. In particular, we employ the weight matrix of $f_\phi(\cdot)$ as the topic-word distributions $(\phi_1, \phi_2, ..., \phi_K)$. In the following, we adopt the topic mixture θ as the topic representations to guide keyphrase generation.

3.2 Neural Keyphrase Generation Model

Here we describe how we generate keyphrases with a topic-aware seq2seq model, which incorporates latent topics (learned by NTM) in its generation process. Below comes more details.

Overview. The keyphrase generation module (KG model) is fed with source post $\mathbf{x}$ in its word sequence form $\mathbf{x}_{seq} = \langle w_1, w_2, ..., w_{|\mathbf{x}|} \rangle$ ($|\mathbf{x}|$ is the number of words in $\mathbf{x}$). Its target is to output a word sequence $\mathbf{y}$ as $\mathbf{x}$'s keyphrase. Particularly, for a source post with multiple gold-standard keyphrases, we follow the practice in Meng et al. (2017) to pair its copies with each of the gold standards to form a training instance.

To generate keyphrases for source posts, the KG model employs a seq2seq model. The **sequence encoder** distills indicative features from an input source post. The decoder then generates its keyphrase, conditioned on the encoded features and the latent topics yielded by NTM (henceforth **topic-aware sequence decoder**).

Sequence Encoder. We employ a bidirectional gated recurrent unit (Bi-GRU) (Cho et al., 2014) to encode the input source sequence. Each word $w_i \in \mathbf{x}_{seq}$ ($i = 1, 2, ..., |\mathbf{x}|$) is first embedded into an embedding vector ν_i, and then mapped into forward and backward hidden states (denoted as $\overrightarrow{\mathbf{h}_i}$ and $\overleftarrow{\mathbf{h}_i}$) with the following defined operations:

$$\overrightarrow{\mathbf{h}_i} = f_{GRU}(\nu_i, \mathbf{h}_{i-1}), \tag{2}$$

$$\overleftarrow{\mathbf{h}_i} = f_{GRU}(\nu_i, \mathbf{h}_{i+1}). \tag{3}$$

The concatenation of $\overrightarrow{\mathbf{h}_i}$ and $\overleftarrow{\mathbf{h}_i}$, $[\overrightarrow{\mathbf{h}_i}; \overleftarrow{\mathbf{h}_i}]$, serves as w_i's hidden state in encoder, denoted as $\mathbf{h}_i$. Finally, we construct a memory bank: $\mathbf{M} = \langle \mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{|\mathbf{x}|} \rangle$, for decoder's attentive retrieval.

Topic-Aware Sequence Decoder. In general, conditioned on the memory bank $\mathbf{M}$ and latent topic θ from NTM, we define the process to generate its keyphrase $\mathbf{y}$ with the following probability:

$$Pr(\mathbf{y} \,|\, \mathbf{x}) = \prod_{j=1}^{|\mathbf{y}|} Pr(y_j \,|\, \mathbf{y}_{<j}, \mathbf{M}, \theta), \tag{4}$$

where $\mathbf{y}_{<j} = \langle y_1, y_2, ..., y_{j-1} \rangle$. And $Pr(y_j|\mathbf{y}_{<j}, \mathbf{M}, \theta)$, denoted as p_j, is a word distribution over vocabulary, reflecting how likely a word to fill in the j-th slot in target keyphrase. Below we describe the procedure to obtain p_j.

Our sequence decoder employs a unidirectional GRU layer. Apart from the general state update, the j-th hidden state $\mathbf{s}_j$ is further designed to take input $\mathbf{x}$'s topic mixture θ into consideration:

$$\mathbf{s}_j = f_{GRU}([\mathbf{u}_j; \theta], \mathbf{s}_{j-1}), \tag{5}$$

where $\mathbf{u}_j$ is the j-th embedded decoder input[3] and $\mathbf{s}_{j-1}$ is the previous hidden state. Here $[;]$ denotes the concatenation operation.

The decoder also looks at $\mathbf{M}$ (learned by sequence encoder) and puts an attention on it to capture important information. When predicting the j-th word in keyphrase, the attention weights on $w_i \in \mathbf{x}_{seq}$ is defined as:

$$\alpha_{ij} = \frac{\exp(f_\alpha(\mathbf{h}_i, \mathbf{s}_j, \theta))}{\sum_{i'=1}^{|\mathbf{x}|} \exp(f_\alpha(\mathbf{h}_{i'}, \mathbf{s}_j, \theta))}, \tag{6}$$

where

$$f_\alpha(\mathbf{h}_i, \mathbf{s}_j, \theta) = \mathbf{v}_\alpha^T \, tanh(\mathbf{W}_\alpha[\mathbf{h}_i; \mathbf{s}_j; \theta] + \mathbf{b}_\alpha). \tag{7}$$

Here $\mathbf{v}_\alpha$, $\mathbf{W}_\alpha$, and $\mathbf{b}_\alpha$ are trainable parameters. $f_\alpha(\cdot)$ measures the semantic relations between the i-th word in the source and the j-th target word to be predicted. Such relations are also calibrated with the input's latent topic θ in order to explore and highlight topic words. We hence obtain the topic sensitive context vector $\mathbf{c}_j$ with:

$$\mathbf{c}_j = \sum_{i=1}^{|\mathbf{x}|} \alpha_{ij} \mathbf{h}_i. \tag{8}$$

Further, conditioned on $\mathbf{c}_j$, we generate the j-th word over the global vocabulary according to:

$$p_{gen} = softmax(\mathbf{W}_{gen}[\mathbf{s}_j; \mathbf{c}_j] + \mathbf{b}_{gen}). \tag{9}$$

In addition, we adopt copy mechanism (See et al., 2017) following Meng et al. (2017), which allows keywords to be directly extracted from the source input. Specifically, we adopt a soft switcher $\lambda_j \in [0, 1]$ to determine whether to copy a word from source as the j-th target word:

$$\lambda_j = sigmoid(\mathbf{W}_\lambda[\mathbf{u}_j; \mathbf{s}_j; \mathbf{c}_j; \theta] + \mathbf{b}_\lambda), \tag{10}$$

with $\mathbf{W}_\lambda$ and $\mathbf{b}_\lambda$ being learnable parameters. Topic information θ is also injected here to guide the switch decision.

Finally, we obtain distribution p_j for predicting the j-th target word with the formula below:

$$p_j = \lambda_j \cdot p_{gen} + (1 - \lambda_j) \cdot \sum_{i=1}^{|\mathbf{x}|} \alpha_{ij}, \tag{11}$$

where attention scores $\{\alpha_{ij}\}_{i=1}^{|\mathbf{x}|}$ serve as the extractive distribution over the source input.

[3] We take the previous word from gold standards in training by teacher forcing and from the predicted word in test.

3.3 Jointly Learning Topics and Keyphrases

Our neural framework allows end-to-end learning of latent topic modeling and keyphrase generation. We first define objective functions for the two modules respectively.

For NTM, the objective function is defined based on negative variational lower bound (Blei et al., 2016). Here due to space limitation, we omit the derivation details already described in Miao et al. (2017), and directly give its loss function:

$$\mathcal{L}_{NTM} = D_{KL}(p(\mathbf{z}) \,||\, q(\mathbf{z} \,|\, \mathbf{x})) - \mathbb{E}_{q(\mathbf{z} \,|\, \mathbf{x})}[p(\mathbf{x} \,|\, \mathbf{z})], \tag{12}$$

where the first term is the Kullback-Leibler divergence loss and the second term reflects the reconstruction loss. $p(\mathbf{z})$ denotes a standard normal prior. $q(\mathbf{z} \,|\, \mathbf{x})$ and $p(\mathbf{x} \,|\, \mathbf{z})$ represent the process of BoW encoder and BoW decoder respectively.

For KG model, we minimize the cross entropy loss over all training instances:

$$\mathcal{L}_{KG} = -\sum_{n=1}^{N} \log(Pr(\mathbf{y}_n \,|\, \mathbf{x}_n, \theta_n)), \tag{13}$$

where N denotes the number of training instances and θ_n is $\mathbf{x}_n$'s latent topics induced from NTM.

Finally, we define the entire framework's training objective with the linear combination of $\mathcal{L}_{NTM}$ and $\mathcal{L}_{KG}$:

$$\mathcal{L} = \mathcal{L}_{NTM} + \gamma \cdot \mathcal{L}_{KG}, \tag{14}$$

where the hyper-parameter γ balances the effects of NTM and KG model. Our two modules can be jointly trained with their parameters updated simultaneously. For inference, we adopt beam search and generate a ranking list of output keyphrases following Meng et al. (2017).

4 Experiment Setup

Datasets. We conduct experiments on three social media datasets collected from two English online platforms, **Twitter** and **StackExchange**, and a Chinese microblog website, **Weibo**. Twitter and Weibo are microblogs encouraging users to freely post with a wide range of topics, while StackExchange, an online Q&A forum, are mainly for question asking (with a title and a description) and seeking answers from others.

The Twitter dataset contains tweets from TREC 2011 microblog track.[4] For Weibo dataset, we first tracked the real-time trending hashtags in Jan-Aug 2014,[5] and then used them as keywords to search posts with hashtag-search API.[6] And the StackExchange dataset is randomly sampled from a publicly available raw corpus.[7]

Source posts	# of posts	Avg len per post	# of KP per post	Source vocab
Twitter	44,113	19.52	1.13	34,010
Weibo	46,296	33.07	1.06	98,310
StackExchange	49,447	87.94	2.43	99,775
Target KP	|KP|	Avg len per KP	% of abs KP	Target vocab
Twitter	4,347	1.92	71.35	4,171
Weibo	2,136	2.55	75.74	2,833
StackExchange	12,114	1.41	54.32	10,852

Table 2: Data statistics of source posts (on the top) and target keyphrases (on the bottom). Avg len: the average number of tokens. KP: keyphrases. Abs KP: absent keyphrases. |KP|: the number of distinct keyphrases.

For the target keyphrases, we employ user-annotated hashtags for Twitter and Weibo following Zhang et al. (2016), and author-assigned tags (e.g., "*artificial-intelligence*") for StackExchange. Posts without such keyphrase tags are hence removed from the datasets. Particularly, for StackExchange, we concatenate the question title together with its description as the source input. For Twitter and Weibo source posts, we retain tokens in hashtags (without # symbols) for those appearing in the middle of posts, since they generally act as semantic elements and thus considered as present keyphrases (Zhang et al., 2016). For those appearing before or after a post, we remove the entire hashtags and regard them as absent keyphrases as is done in Wang et al. (2019).

For model training and evaluation, we split the data into three subsets with 80%, 10%, and 10%, corresponding to training, development, and test set. The statistics of the three datasets are shown in Table 2. As can be seen, over 50% of the keyphrases do not appear in their source posts, thus extractive approaches will fail in dealing with these posts. We also observe that StackExchange exhibits different keyphrase statistics compared to either Twitter or Weibo, with more keyphrases appearing in one post and more diverse keyphrases.

[4] http://trec.nist.gov/data/tweets/

[5] http://open.weibo.com/wiki/Trends/

[6] http://www.open.weibo.com/wiki/2/

[7] https://archive.org/details/stackexchange

Preprocessing. For Twitter dataset, we employed Twitter preprocessing toolkit in Baziotis et al. (2017) for source post and hashtag (keyphrase) tokenization. Chinese Weibo data was preprocessed with Jieba toolkit[8] for word segmentation, and English StackExchange data with natural language toolkit (NLTK) for tokenization.[9]

We further take the following preprocessing steps for each of the three datasets: First, posts with meaningless keyphrases (e.g., single-character ones) were filtered out; also removed were non-alphabetic (for English data) and retweet-only (e.g., "*RT*") posts. Second, links, mentions (@username), and digits were replaced with generic tags "*URL*", "*MENT*", and "*DIGIT*" following Wang et al. (2019). Third, a vocabulary was maintained, with 30K most frequent words for Twitter, and 50K for Weibo and StackExchange each. For BoW vocabulary of the input $\mathbf{x}_{bow}$ for NTM, stop words and punctuation were removed.

Parameter Settings. We implement our model based on the pytorch framework in Paszke et al. (2017). For NTM, we implement it following the design[10] in Zeng et al. (2018) and set topic number K to 50. The KG model is set up mostly based on Meng et al. (2017). For its sequence encoder, we adopt two layers of bidirectional GRU and one layer of unidirectional GRU for its decoder. The hidden size of the GRU is 300 (for bi-GRU, 150 for each direction). For the embedding, its size is set to 150 and values are randomly initialized. We apply Adam (Kingma and Ba, 2014) with initial learning rate as $1e-3$. In training process, gradient clipping $= 1.0$ is conducted to stabilize the training. Early-stopping strategy (Caruana et al., 2001) is adopted based on the validation loss. Before joint training, we pretrain NTM for 100 epochs and KG model for 1 epoch as the convergence speed of NTM is much slower than the KG model. We empirically set the $\gamma = 1.0$ for balancing NTM and KG loss (Eq. 14) and iteratively update the parameters in each module and then their combination in turn.

Comparisons. In comparison, we first consider a simple baseline selecting majority keyphrases (henceforth MAJORITY) — the top K keyphrases ranked by their frequency in training data are used as the keyphrases for all test instances. We also compare with the following extractive baselines, where n-grams ($n = 1, 2, 3$) in source posts are ranked by TF-IDF scores (henceforth TF-IDF), TextRank algorighm (Mihalcea and Tarau, 2004) (henceforth TEXTRANK), and KEA system (Witten et al., 1999) (henceforth KEA). We also compare with a neural state-of-the-art keyphrase extraction model based on sequence tagging (Zhang et al., 2016) (henceforth SEQ-TAG). In addition, we take the following state-of-the-art keyphrase generation models into consideration: seq2seq model with copy mechanism (Meng et al., 2017) (henceforth SEQ2SEQ-COPY) and its variation SEQ2SEQ without copy mechanism, SEQ2SEQ-CORR (Chen et al., 2018) exploiting keyphrase correlations, and TG-NET (Chen et al., 2019b) jointly modeling of titles and descriptions (thereby only tested on StackExchange).

[8] https://github.com/fxsjy/jieba
[9] https://www.nltk.org/
[10] https://github.com/zengjichuan/TMN

5 Experimental Results

In the experiment, we first evaluate our performance on keyphrase prediction (§5.1). Then, we study whether jointly learning keyphrase generation can in turn help produce coherent topics (§5.2). At last, further discussions (§5.3) are presented with an ablation study, a case study, and an analysis for varying text genres.

5.1 Keyphrase Prediction Results

In this section, we examine our performance in predicting keyphrases for social media. We first discuss the main comparison results, followed by a discussion for present and absent keyphrases.

Popular information retrieval metrics macro-average F1@K and mean average precision (MAP) are adopted for evaluation. Here for Twitter and Weibo, most posts are tagged with one keyphrase on average (Table 2), thus F1@1 and F1@3 are reported. For StackExchange, we report F1@3 and F1@5, because on average, posts have 2.4 keyphrases. MAP is measured over the top 5 predictions for all three datasets. For keyphrase matching, we consider keyphases after stemmed by Porter Stemmer following Meng et al. (2017).

Main Comparison Discussion. Table 3 shows the main comparison results on our three datasets, where higher scores indicate better performance. From all three datasets, we observe:

• ***Social media keyphrase prediction is challenging.*** As can be seen, all simple baselines give

Model	Twitter			Weibo			StackExchange		
	F1@1	F1@3	MAP	F1@1	F1@3	MAP	F1@3	F1@5	MAP
Baselines									
MAJORITY	9.36	11.85	15.22	4.16	3.31	5.47	1.79	1.89	1.59
TF-IDF	1.16	1.14	1.89	1.90	1.51	2.46	13.50	12.74	12.61
TEXTRANK	1.73	1.94	1.89	0.18	0.49	0.57	6.03	8.28	4.76
KEA	0.50	0.56	0.50	0.20	0.20	0.20	15.80	15.23	14.25
State of the arts									
SEQ-TAG	22.79±0.3	12.27±0.2	22.44±0.3	16.34±0.2	8.99±0.1	16.53±0.3	17.58±1.6	12.82±1.2	19.03±1.3
SEQ2SEQ	34.10±0.5	26.01±0.3	41.11±0.3	28.17±1.7	20.59±0.9	34.19±1.7	22.99±0.3	20.65±0.2	23.95±0.3
SEQ2SEQ-COPY	36.60±1.1	26.79±0.5	43.12±1.2	32.01±0.3	22.69±0.2	38.01±0.1	31.53±0.1	27.41±0.2	33.45±0.1
SEQ2SEQ-CORR	34.97±0.8	26.13±0.4	41.64±0.5	31.64±0.7	22.24±0.5	37.47±0.8	30.89±0.3	26.97±0.2	32.87±0.6
TG-NET	-	-	-	-	-	-	32.02±0.3	27.84±0.3	34.05±0.4
Our model	**38.49**±0.3	**27.84**±0.0	**45.12**±0.2	**34.99**±0.3	**24.42**±0.2	**41.29**±0.4	**33.41**±0.2	**29.16**±0.1	**35.52**±0.1

Table 3: Main comparison results displayed with average scores (in %) and their standard deviations over the results with 5 sets of random initialization seeds. Boldface scores in each column indicate the best results. Our model significantly outperforms all comparisons on all three datasets ($p < 0.05$, paired t-test).

poor performance. This indicates that predicting keyphrases for social media language is a challenging task. It is impossible to rely on simple statistics or rules to yield good results.

• ***Seq2seq-based keyphrase generation models are effective.*** Compared to the extractive baselines and SEQ-TAG, seq2seq-based models perform much better. It is because social media's informal language style results in a large amount of absent keyphrases (Table 2), which is impossible for extractive methods to make correct predictions. We also find SEQ2SEQ-COPY better than SEQ2SEQ, suggesting the effectiveness to combine source word extraction with word generation when predicting keyphrases.

• ***Latent topics are consistently helpful for indicating keyphrases.*** It is observed that our model achieves the best results, significantly outperforming all comparisons by a large margin. This shows the usefulness of leveraging latent topics in keyphrase prediction. Interestingly, compared with StackExchange, we achieve larger improvements for Twitter and Weibo, both exhibiting more informal nature and prominent word order misuse. For such text genres, latent topics, learned under BoW assumption, are more helpful.

Also, the following interesting points can be observed by comparing results across datasets:

• ***Keyphrase generation is more challenging for StackExchange.*** When MAP scores of seq2seq-based methods are compared over the three datasets, we find that the scores on StackExchange are generally lower. It is probably attributed to the data characteristics of more diverse keyphrases and larger target vocabulary (Table 2).

• ***Twitter and Weibo data is noisier.*** We notice that TF-IDF, TEXTRANK, and KEA perform much worse than MAJORITY, while the opposite is observed on StackExchange. It is because Twitter and Weibo, as microblogs, contain shorter posts (Table 2) and exhibit more informal language styles. In general, models relying on simple word statistics would suffer from such noisy data.

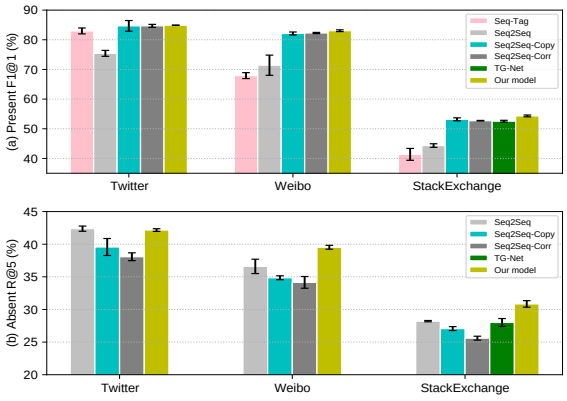


Figure 2: The prediction results for present (on the top) and absent keyphrases (on the bottom, R@5: recall@5). For present cases, from left to right shows the results of SEQ-TAG, SEQ2SEQ, SEQ2SEQ-COPY, SEQ2SEQ-CORR, TG-NET (only for StackExchange), and our model. For absent cases, models (except SEQ-TAG) are shown in the same order.

Present and Absent Keyphrase Prediction. We further discuss how our model performs in producing present and absent keyphrases. The comparison results with all neural-based models are shown in Figure 2. Here F1@1 is adopted for evaluating the prediction of present keyphrases and recall@5 for absent keyphrases.

Datasets	Twitter	StackExchange
LDA	41.12	35.13
BTM	43.12	43.52
NTM	43.82	43.04
Our model	**46.28**	**45.12**

Table 4: C_V topic coherence score comparison on our two English datasets. Higher scores indicate better coherence. Our model produces the best scores.

The results indicate that our model consistently outperforms comparison models in predicting either absent or present keyphrases. Also, interestingly, copy mechanism seems to somehow sacrifice the performance on absent keyphrase generation for correctly extracting the present ones. Such side effects, however, are not observed on our model. It is probably attributed to our ability to associate posts with corpus-level topics, hence enabling absent keywords from other posts to be "copied". This observation also demonstrates the latent topics can help our model to better decide whether to copy (Eq. 10).

5.2 Latent Topic Analysis

We have shown latent topics useful for social media keyphrase generation in §5.1. Here we analyze whether our model can learn meaningful topics.

Coherence Score Comparison. We first evaluate topic coherence with an automatic C_V measure. Here we employ Palmetto toolkit[11] (Röder et al., 2015) on the top 10 words from each latent topic following Zeng et al. (2018). The results are only reported on English Twitter and StackExchange because Palmetto does not support Chinese. For comparisons, we consider LDA (implemented with a gensim LdaMulticore package[12]), BTM[13] (Yan et al., 2013) (a state-of-the-art topic model specifically for short texts), and NTM (Miao et al., 2017). For LDA and BTM, we run Gibbs sampling with $1,000$ iterations to ensure convergence. From the results in Table 4, we observe that our model outperforms all the comparison topic models by large margins, which implies that jointly exploring keyphrase generation can in turn help produce coherent topics.

[11] https://github.com/dice-group/Palmetto/

[12] https://pypi.org/project/gensim/

[13] https://github.com/xiaohuiyan/BTM

Sample Topics. To further evaluate whether our model can produce coherent topics qualitatively, we probe into some sample words (Table 5) reflecting the topic "*super bowl*" discovered by various models from Twitter. As can be seen, there are mixed non-topic words [14] in LDA's, BTM's, and NTM's sample topic. Compared with them, our inferred topic looks more coherent. For example, "*steeler*" and "*packer*", names of *super bowl* teams, are correctly included into the cluster.

LDA	bowl super quote steeler jan watching egypt playing glee girl
BTM	bowl super anthem national christina aguilera fail word brand playing
NTM	super bowl eye protester winning watch halftime ship sport mena
Our model	bowl super yellow green packer steeler nom commercial win winner

Table 5: Top 10 terms for latent topics "*super bowl*". Red and underlined words indicate non-topic words.

5.3 Further Discussions

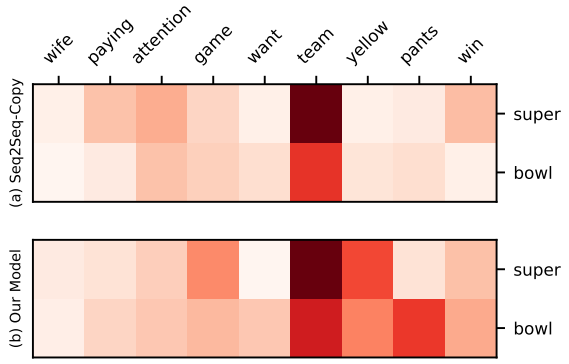
Ablation Study. We compare the results of our full model and its four ablated variants to analyze the relative contributions of topics on different components. The results in Table 6 indicate the competitive effect of topics on decoder attention and that on hidden states, but combining them both help our full model achieve the best performance. We also observe that pre-trained topics only bring a small boost, indicated by the close scores yielded by our model (*separate train*) and SEQ2SEQ-COPY. This suggests that the joint training is crucial to better absorb latent topics.

Model	Twitter	Weibo	SE
SEQ2SEQ-COPY	36.60	32.01	31.53
Our model (*separate train*)	36.75	32.75	31.78
Our model (*w/o topic-attn*)	37.24	32.42	32.34
Our model (*w/o topic-state*)	37.44	33.48	31.98
Our full model	**38.49**	**34.99**	**33.41**

Table 6: Comparison results of our ablation models on three datasets (SE: StackExchange) — *separate train*: our model with pre-trained latent topics; *w/o topic-attn*: decoder attention without topics (Eq. 7); *w/o topic-state*: decoder hidden states without topics (Eq. 5). We report F1@1 for Twitter and Weibo, F1@3 for StackExchange. Best results are in bold.

[14] Non-topic words refer to words that cannot clearly indicate the corresponding topic, including off-topic words more likely to reflect other topics.

Case Study. We feed the tweet S in Table 1 into both SEQ2SEQ-COPY and our model. Eventually our model correctly predicts the keyphrase as "*super bowl*" while SEQ2SEQ-COPY gives a wrong prediction "*team follow back*" (posted to ask other to follow back). To analyze the reason behind, we visualize the attention weights of two models in Figure 3. It can be seen that both models highlight the common word "*team*", which frequently appears in "*team follow back*"-tagged tweets. By joint modeling of latent topics, our model additionally emphasizes topic words "*yellow*" and "*pants*", which are signals indicating a super bowl team *steeler* (also reflected in the 1^{st} topic) and thus helpful to correctly generate "*super bowl*" as its keyphrase. Without such topic guidance, SEQ2SEQ-COPY wrongly predicts a common but unrelated term "*team follow back*".



1^{st} Topic	steeler national team packer win

Figure 3: Attention visualization for the sample post in Table 1. Only non-stopwords are selected. The table below shows the top five words for the 1^{st} topic.

Topic-Aware KG for Other Text Genres. We have shown the effectiveness of latent topics on social media keyphrase generation. To examine how they affect in identifying keyphrases for well-edited language, we also experiment on the traditional scientific article datasets (Meng et al., 2017), but limited improvements are observed. Latent topics can better help keyphrase generation on social media, probably because there are larger proportion of keyphrases with absent words (Figure 4), where latent topics can cluster relevant posts and enrich the source contexts. Another possible reason lies in that social media language exhibits prominent arbitrary word orders. Thus latent topics, learned under BoW assumption, can better provide useful auxiliary features.

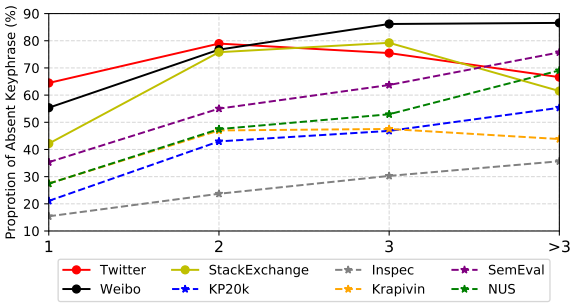


Figure 4: Proportion of absent n-gram keyphrases (n: $1, 2, 3, > 3$). The dashed lines with '*' marks: the five scientific article datasets used in Meng et al. (2017).

6 Conclusion and Future Work

We have presented a novel social media keyphrase generation model that allows the joint learning of latent topic representations. Experimental results on three newly constructed social media datasets show that our model significantly outperforms state-of-the-art methods in keyphrase prediction, meanwhile produces more coherent topics. Further analysis interprets our superiority to discover key information from noisy social media data.

In the future, we will explore how to explicitly leverage the topic-word distribution to further improve the performance. Also, our topic-aware neural keyphrase generation model can be investigated in a broader range of text generation tasks.

Acknowledgements

This work is supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14208815 and No. CUHK 14210717 of the General Research Fund). We thank ACL reviewers for their insightful suggestions on various aspects of this work.

References

Haoli Bai, Zhuangbin Chen, Michael R. Lyu, Irwin King, and Zenglin Xu. 2018. Neural relational topic models for scientific article analysis. In *Proceedings of ACM International Conference on Information and Knowledge Management*.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. 2016. Variational inference: A review for statisticians. *CoRR*, abs/1601.00670.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.

Rich Caruana, Steve Lawrence, and C Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Proceedings of Neural Information Processing Systems*.

Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. In *Proceedings of Empirical Methods in Natural Language Processing*.

Wang Chen, Hou Pong Chan, Piji Li, Lidong Bing, and Irwin King. 2019a. An integrated approach for keyphrase generation via exploring the power of retrieval and extraction. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R. Lyu. 2019b. Title-guided encoding for keyphrase generation. In *Proceedings of AAAI Conference on Artificial Intelligence*.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of Empirical Methods in Natural Language Processing*.

Sujatha Das Gollapalli, Xiaoli Li, and Peng Yang. 2017. Incorporating expert knowledge into keyphrase extraction. In *Proceedings of AAAI Conference on Artificial Intelligence*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Jing Li, Yan Song, Zhongyu Wei, and Kam-Fai Wong. 2018. A joint model of conversational discourse and latent topics on microblogs. *Journal of Computational Linguistics*.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of Empirical Methods in Natural Language Processing*.

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of Empirical Methods in Natural Language Processing*.

Olena Medelyan, Eibe Frank, and Ian H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of Empirical Methods in Natural Language Processing*.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of Association for Computational Linguistics*.

Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Sujian Li, and Houfeng Wang. 2012. Entity-centric topic-oriented opinion summarization in twitter. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining*.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering discrete latent topics with neural variational inference. In *Proceedings of International Conference on Machine Learning*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of Empirical Methods in Natural Language Processing*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *Proceedings of Neural Information Processing Systems*.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of ACM International Conference on Web Search and Data Mining*.

Derek Ruths and Jürgen Pfeffer. 2014. Social media for large studies of behavior. *Journal of Science*.

Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of Association for Computational Linguistics*.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. *arXiv preprint arXiv:1703.01488*.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of AAAI Conference on Artificial Intelligence*.

Yue Wang, Jing Li, Irwin King, Michael R. Lyu, and Shuming Shi. 2019. Microblog hashtag generation via encoding conversation contexts. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jianshu Weng and Bu-Sung Lee. 2011. Event detection in twitter. In *Proceedings of AAAI conference on weblogs and social media*.

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. KEA: practical automatic keyphrase extraction. In *Proceedings of ACM conference on Digital Libraries*.

Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *Proceedings of international conference on World Wide Web*.

Jichuan Zeng, Jing Li, Yulan He, Cuiyun Gao, Michael R. Lyu, and Irwin King. 2019. What you say and how you say it: Joint modeling of topics and discourse in microblog conversations. *Transactions of Association for Computational Linguistics*.

Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R. Lyu, and Irwin King. 2018. Topic memory networks for short text classification. In *Proceedings of Empirical Methods in Natural Language Processing*.

Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of Empirical Methods in Natural Language Processing*.

Yingyi Zhang, Jing Li, Yan Song, and Chengzhi Zhang. 2018. Encoding conversation context for neural keyphrase extraction from microblog posts. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.