IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 16, 2019, accepted June 1, 2019, date of publication June 14, 2019, date of current version July 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2923057

User-Oriented Paraphrase Generation With Keywords Controlled Network

DAOJIAN ZENG[1], HAORAN ZHANG[2], LINGYUN XIANG[1], JIN WANG[1], AND GUOLIANG JI[3]
[1]School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410004, China
[2]School of Information Science, University of Illinois Urbana–Champaign, Urbana, IL 61801, USA
[3]China Three Gorges Corporation, Beijing 100038, China

Corresponding author: Haoran Zhang (haoranz6@illinois.edu)

This work was supported in part by the National Natural Science Foundation of China under Grant 61602059, and in part by the Hunan Provincial Natural Science Foundation of China under Grant 2017JJ3334 and Grant 2019JJ50655.

ABSTRACT Paraphrase generation can help with both downstream tasks in natural language processing (NLP) and human writing in our daily life. Most of the prevalent neural models focus on the former usages and generate uncontrolled paraphrase while they ignore the subtleties of users' requirement. In addition, the existing tools for users are usually rule-based which is unnatural due to the complexity of the paraphrase nature. To this end, we propose a keyword controlled network (KCN) which can be used as an assistant paraphrase generation tool. The KCN works in an interactive manner and generates different paraphrases given different keywords. The model is based on a Sequence-to-Sequence (Seq2Seq) framework integrated with copy mechanism. Given the source sentence and the keywords, two encoders transform them into vector representations. Then, the representations are fused together and used for the decoder. The decoder with attention mechanism either copies the words from the keywords or generates words from the whole dictionary. In the training stage, as the source sentence and the target sentence are all valid paraphrases, the model is trained to generate each given different keywords, which simulates the behaviors of users. The extensive experiments on three datasets show that our method outperforms baselines in the automatic evaluation (0.06 absolute improvement in BLEU) and the generated paraphrases meet user expectation in the human evaluation.

INDEX TERMS Copy mechanism, keyword, paraphrase generation, Seq2Seq.

I. INTRODUCTION

Paraphrase generation is not only a helpful module in NLP systems but also an essential part of our daily life. While most of the researchers focus on paraphrase identification [1], there is less effort spent on paraphrase generation [2]. According to the usage, paraphrase generation can be categorized as model-oriented and user-oriented.

- model-oriented paraphrase: a data pre-processing trick before inference for NLP models such as Question Answering [3]; a data augmentation trick in training their NLP models thoroughly [4].
- user-oriented paraphrase: an assistant reading tool to comprehend passages easier or an assistant writing tool to write passages with specific style they need [5].

The recent paraphrase generation models mainly focus on model-oriented and pay less attention to user-oriented. Most of the existing user-oriented tools are rule-based systems [6] which have fatal drawbacks. Firstly, rule-based systems cannot capture the subtleties of natural language considering the context [7]. For example, in Fig. 1(a): Although ''address'' is the synonym of ''deal with'' in some cases, it is not proper given the context ''speech''. Secondly, natural language rules are too complicated to be manually designed for real-world usage. For example, pragmatic paraphrase and referential paraphrase are the hardest phenomena but happened everywhere [8], that is, two sentences refer to the same fact. For example, in Fig. 1(b) and Fig. 1(c), paraphrase generation not only requires word-syntax manipulation but also involves context comprehension and common sense.

Recently, thanks to the success of deep learning in NLP, many neural-based models are proposed for paraphrase generation [9]–[13]. These pure data-driven models do not suffer from problems aforementioned. However, an important issue remains unresolved: The generated paraphrases still not conform to users' requirements because they only

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Tong.

He is due to **address** a speech.	Close the door please.	**Trump** sends new tweets.
He is due to **deal with** a speech.	There is air flow.	**The president** sends new tweets.
(a)	(b)	(c)

FIGURE 1. Examples of many real-world paraphrases. (a)The problem of rule-based systems. Substitution from *address* to *deal with* changes the meaning. (b, c) Two kinds of real-world paraphrase. Referential paraphrase and pragmatic paraphrase are very common in our daily life and MSCOCO dataset. (a) Problem of rule-based paraphrase. (b) Pragmatic Paraphrase. (c) Referential Paraphrase.

generate stiffly uncontrolled outputs. An exception, Syntactically Controlled Paraphrase Networks (SCPN) [14] is the closest model to users' need, which combines both the inputs of the source sentence and the target syntax template, and outputs a syntactically controlled paraphrase. Nevertheless, the syntax templates require users to possess linguistic knowledge, and use specific syntactic annotation to describe the expected outputs. For example, users have to define their expected output in the form of `(S(NP)(,)(ADVP)(,)(VP))`. From users' perspective, it is hard to come up with the template in advance. In addition, the qualities of the results highly depend on the appropriateness of the syntax templates.

To overcome these difficulties, we present a Keywords Controlled Network (KCN) which meets the requirements of users. The form of the outputs is determined by the keywords sequence, instead of the syntax template. The motivation of KCN stems from a real-world scenario: the TOEFL integrated writing task asks participators to read a passage containing three points, then to listen to a lecture which comments the three points respectively. The answers should keep the information intact and be stated in a logical, clear way. To do so, most of the participators take unstylish notes with keywords and symbols firstly, then replace some words, change the orders, add some conjunctions to the notes, and finally accomplish the writing according to the notes. When they prepare for the test, students might practice many times, whose writing changes as the notes change. In other words, the form of the writing is determined by notes, and the meaning depends on listening and reading.

Inspired by this manual paraphrase process, KCN has two inputs, the source sentence to be paraphrased and the keywords controlling the target superficial form as what the note does in the TOEFL writing. The model is based on Seq2Seq [15] structure with two inputs. Two encoders transform these two inputs into vector representations. Then, the representations are fused together and used by decoder. The decoder with attention mechanism either generates words from the whole dictionary or copies words from keywords sequence. Because we separate the meaning and the form of a sentence, KCN should predict the source sentence itself given the source keywords. To this end, in the training stage, we introduce two modes to generate outputs according to different keywords, which can boost the robustness of the model. Autoencoder-mode is shown in the upper of Fig. 2. Given the source sentence as input, KCN takes the keywords from source sentence and uses the keywords to reconstruct source sentence. Paraphrase-mode is depicted in the lower of Fig. 2: KCN takes the keywords from target paraphrase to

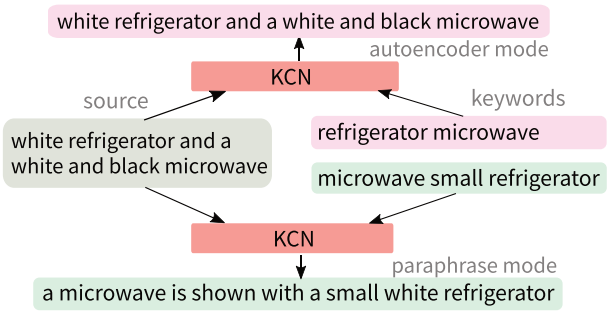


FIGURE 2. The training stage of KCN. The upper is autoencoder-mode, and the lower is paraphrase mode. The model generates different paraphrase according to different keywords.

make some changes. By this way, KCN can be more sensitive to different keywords inputs, just like the novel training strategies in Computer Vision [16], [17], Natural Language Processing [18], [19] and Indoor Localization [20].

In experiments, the qualitative evaluation shows that KCN outperforms the state-of-the-art models by a huge margin and covers most of the keywords. In the human evaluation, evaluators found that the model is able to generate many different paraphrases depending on different keywords hints.

Our contribution lies in three aspects:

- KCN focuses on controlled paraphrase generation and innovatively uses additional keywords inputs to determine the superficial form of the generate paraphrase.
- KCN introduces autoencoder-mode and paraphrase-mode for training. This new training strategy makes it generalize to different keyword hints.
- KCN is user-friendly since it does not require syntax templates. It has the potentials for real-world usage as writing assistant.

II. RELATED WORK

The definition of paraphrase has been controversial for decades in NLP. The datasets and models vary from one to another depending on their different definitions. In the following subsections, additionally to introducing the datasets and the models, we also illustrate the idea of using lexical constraints in other areas and its relation to KCN.

A. PARAPHRASE DEFINITION AND DATASETS

A common definition is bidirectional entailment [21], which is used by MSRP [22] to construct a thousand-pairs paraphrase corpus. However, exact bidirectional entailment rarely exists and constraints the diversities of paraphrasing. Thus, they use a loose bidirectional entailment criterion, which triggers intense disagreement among annotators.

P4P corpus [23] categorizes paraphrase into morpholexicon-based changes, structure-based changes, semantics-based changes and miscellaneous changes. Linguistics researchers further give extensive typologies and examples for paraphrase [8]. Some of them even involve external knowledge, such as coreference and facts. However, these complex typologies make it hard to annotate large-scale dataset for neural-network-based researches. PARANMT50M [24], [25] constructs a very big paraphrase corpus, leveraging a paraphrase property that sentences generated by back-translation might be paraphrases. MSCOCO is also a popular big dataset which stems from image caption. In paraphrase area, two sentences in MSCOCO are paraphrases if they refer to the same image, which can be called pragmatic paraphrase.

B. PARAPHRASE GENERATION

Models of paraphrase generation also vary from the views to paraphrase. The method proposed in [26] is the earliest we've known discussing paraphrase generation, who envisions the most widely used method in industry nowadays. That is, a paraphrase is generated by detecting and replacing the keywords or keyphrases by a lookup dictionary. The algorithm in [27] transforms the source sentence into Finite State Automata and generate the target paraphrase according to it. Statistical machine translation based model [28] can also do well in paraphrase generation, but fails in some syntactical paraphrase.

Recently, a variety of neural network based paraphrase generations have made a great breakthrough. Residual LSTM [9] is the first model employing neural machine translation techniques in paraphrase generation area. After that, VAE-SVG [10] leverages the variational autoencoder to tackle the problem of the limited diversities. Inspired by the success of copynet [29], PtrNet [30] in Question Answering, CoRe [12] fuses copy mechanism and restrict rewriting to model the daily life paraphrase scenario. RbM-SL [11] goes further, uses inverse reinforcement learning and adversarial training to evaluate and generate paraphrase in high quality. SCPN [14] considers additional syntax information in paraphrase generation. Rather than Seq2Seq used by the above methods, WEAN [13] purposes a neural way to find the proper target word to replace. DGEN [31] uses soft attention to replace words or phrases for paraphrase from the source sentence by an off-the-shelf dictionary. However, these methods ignore all the syntactic and pragmatic variation of paraphrase.

C. CONSTRAINED DECODING

Interactive Machine Translation (IMT) is the closed area to KCN. IMT is considered when the outputs of the machine translation systems are forced to contain given words, which is very useful in domains with exclusive words. In the pre-deep-learning era, it is conducted via prefix-suffix searching [32], [33], however, they are not compatible with the current neural-based Seq2Seq model. Recently, based on many new beam-search techniques, many Neural IMT methods have been proposed. On the one hand, the method proposed in [34] leverages the finite-state machine to realize constraint beam search in the neural model, containing both constraints selection and placement. On the other hand, GBS [35] enforces that every constrained word appears in the output (without selection). However, the complexity of their grid beam search is $O(NkC)$.[1] Although DBA [36] improves the complexity of GBS to $O(1)$, it still needs 0.6 GPU seconds to decode single sentence. Different from them, KCN uses soft constraint and decode the outputs just like common neural machine translation.

III. METHODOLOGY

The user-oriented paraphrase is formulated as keywords controlled paraphrase generation. In this section, we present an innovative KCN model that incorporates multi-encoder into Seq2Seq to fulfill this task. The overview of KCN is shown in Fig. 3. It illustrates the procedure that handles one example. Two separated encoders transform the source sentence and keywords sequence into fix-dimension context vectors. The decoder part takes keywords sequence and source sentence into consideration and integrates copy mechanism to generate the paraphrase. We describe the task definition, encoder, decoder and training strategy in detail below.

A. TASK DEFINITION

The prevalent models of paraphrase generation use dataset $\langle s^s, s^t \rangle$, where s^s is the source sentence and s^t is the target sentence. The task is automatic generation of paraphrase s^t from a given sentence s^s. In contrast, the keywords controlled paraphrase requires not only the source sentence but also the provided keywords to generate the corresponding sentence. The keywords controlled paraphrase generation is defined as follows:

$$KCN(s^s, k^u) = s^u \tag{1}$$

where k^u is the keywords given by users, s^s is the source sentence, s^u is the expected outputs.

B. ENCODER

We use bi-directional RNN with Gated Recurrent Unit (GRU) [37] as encoders. Because we have two inputs, source sentence and keywords, we use two encoders with the same structure to encode the source sentence and keywords respectively. Keywords encoder and sentence encoder do not share weights.

$$\overrightarrow{h_i} = f(w_i, h_{i-1}) \tag{2}$$

$$\overleftarrow{h_i} = f(w_i, h_{i+1}) \tag{3}$$

$$o_i = [\overrightarrow{h_i}; \overleftarrow{h_i}] \tag{4}$$

where f is the standard GRU, which is a common way to capture long-term dependency. And, the bidirectional encoder is capable of combining the context of both sides. In (4), we concatenate the vectors in two directions. The encoders

[1] N is sentence length, k is the beam size and C is the constraint count.

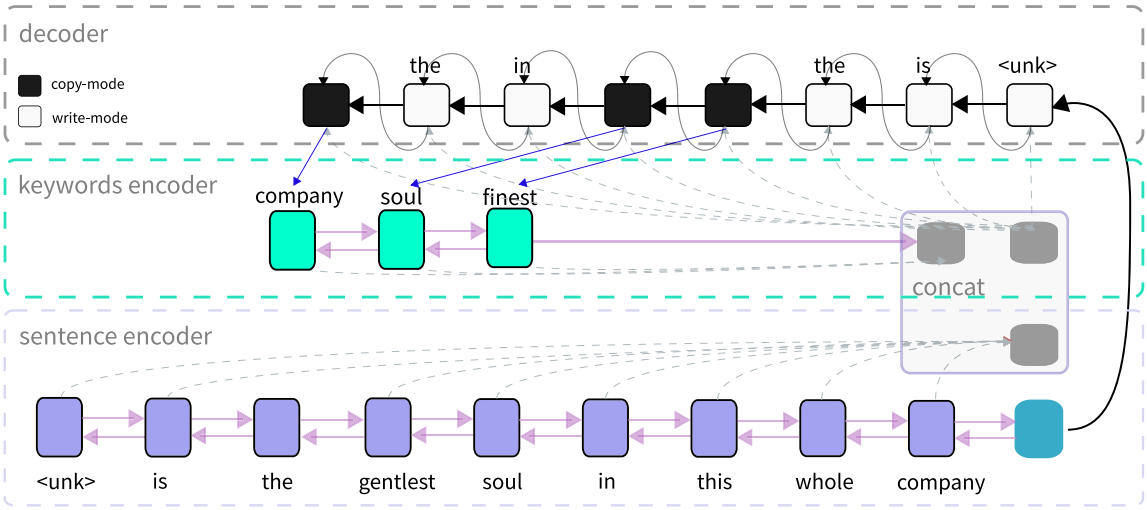


FIGURE 3. Overview of Keywords Controlled Network (KCN). The figure shows the paraphrase mode of KCN, which takes one source sentence and one target keywords. The autoencoder mode is similar that takes keywords extracted from the source sentence. Generated paraphrase conforms to the given keywords. The result paraphrase: ⟨*unk*⟩ ***is the finest soul in the company,*** where ***finest soul company*** are directly copied from keywords.

essentially compress the sequences into fixed-length vectors. For this task, the last hidden state of sentence encoder, h^s_{-1}, is used as the initial hidden state of the decoder. The outputs of sentence and keywords encoder at all time step, o^s_i and o^k_i, are saved for calculating attentions.

C. DECODER

Many models incorporate copy mechanism into decoder aiming to enhance the performance of information extraction [38], paraphrase [11], question answering [29] and summarization [12], [39].

In KCN, the decoder considers the encoding results of both sentence and keywords. There are two competitive decoder modes: copy-mode and writing-mode. We force the copy-mode decoder to copy from the keywords sequence and the write-mode generate words from the whole dictionary. To leverage information of source sentence and keywords, we not only initialize the first hidden state of the decoder by o^s_{-1}, like [15], but also calculate attentions via $\{o^s_0, \ldots, o^s_l\}$ and $\{o^k_0, \ldots, o^k_l\}$.

We calculate attentions for keywords sequence and source sentence via two attention functions with same structure [40]. The inputs are hidden state in the decoder and outputs of encoder at all time-steps. And the outputs, the attentive context vector c^k and c^s are then combined and predict target words. The attentive context vectors c are computed by (5).

$$\begin{aligned} c &= \sum_{i=1}^{n} \alpha_i \cdot o_i \\ \alpha &= softmax(\beta) \\ \beta &= w \cdot [e; h^D_{i-1}] \end{aligned} \tag{5}$$

where w is an unshared weight matrix, e is the embedded input token of decoder and h^D_{i-1} is the last hidden state of the decoder.

Then, we concatenate three vectors and map them into a joint representation.

$$c = g(w^{cat} \cdot [e; c^k; c^s] + b^{cat}) \tag{6}$$

$$o^D_t, h^D_t = f(h^D_{t-1}, c) \tag{7}$$

where f is GRU, $g(\cdot)$ is an activation function, w^{cat} is the weight matrix and b^{cat} is the bias.

We use selu [41] for activation:

$$g(x) = \lambda \begin{cases} x, & \text{if } x > 0 \\ \alpha e^x - \alpha, & \text{if } x \leq 0 \end{cases} \tag{8}$$

where $\lambda \approx 1.0507$ and $\alpha \approx 1.6733$ (these parameters are parts of selu, ensuring it work as expected). In addition to the non-linearity introduced as other activation functions do, this activation function adaptively converges its outputs to zero mean and one variance, which are beneficial to the training. Thus, we no longer need to turn to batch normalization [42] or layer normalization [43].

For the copy-mode, the decoder predicts the position of the target word in the keywords sequence; for the write-mode, the decoder predicts words from the whole dictionary. The generated outputs, either from the source sentence or from the dictionary, are fed into the decoder in the next time step.

$$\begin{aligned} p_{copy}(y_t) &= softmax(w^c \cdot o^D_t + b^c) \\ p_{write}(y_t) &= softmax(w^w \cdot o^D_t + b^w) \end{aligned} \tag{9}$$

where w^c, w^w are the weight matrixes. b^c and b^w are the biases. The dimensions of the output probability vectors are different: while the dimension of p_{write} is the size of the whole dictionary, that of p_{copy} is the length of the keyword sequence, in which words at the positions are different as the inputs change.

Finally, we train a binary gate to switch between the writing mode and copy mode:

$$p_{gate}(z_t) = sigmoid(w^g \cdot o_t^D + b^g) \tag{10}$$

where we use weight w^g and bias b^g to map o_t^D to a scalar. In the training stage, when the target word in the source sentence, we turn p_{gate} on by setting the ground truth $p^*_{gate} = 1$, and 0 otherwise. In the inference stage, the gate is turned on when the predicted $p_{gate} > 0.5$. When target words appear both in the dictionary and the keyword sequence, we force the model to use copy-mode at the training stage.

D. TRAINING STRATEGY

The KCN need keywords as one of the inputs. However, the existing paraphrase datasets only contain the source sentence s^s and the target sentence s^t. To train the network, We first use keywords extraction approach to extend the sentence pairs dataset $\langle s^s, s^t\rangle$ to quadruples dataset, $\langle s^s, k^s, s^t, k^t\rangle$, where k^s and k^t are the keywords extracted from source sentence s^s and target sentence s^t respectively.

1) KEYWORDS EXTRACTION

While the source sentence determines the meaning of the generated sentence, the keywords control the superficial form. Then how to simulate keywords inputs in the training stage? Because the original dataset $\langle s^s, s^t\rangle$ does not contain keywords, we extract $\langle k^s, k^t\rangle$ from $\langle s^s, s^t\rangle$ by the combination of *PositionRank* and *RandomSample*.

PositionRank [44] is an unsupervised graph model, leveraging position information to extract keywords. We use it as a function:

$$PR(s) = PositionRank(s) \tag{11}$$

where $PR(s)$ denotes that the keywords extracted from sentence s by *PositionRank*.

But the original *PositionRank* is designed for abstract-keywords extraction in academic papers, and thus not very good at extracting sentence level keywords. In training, we find that most of the words in s^s and s^t are overlapped while only a few words are different. Thus, $PR(s^s)$ and $PR(s^t)$ are always the same, which cannot lead to diverse paraphrase generation. To solve this problem, we use *RandomSample* to compensate for the insufficiency of *PositionRank*. *RandomSample* is to sample from the difference set of the two inputs. Unlike conventional keywords extraction, outputs of *RandomSample* may contain stop words, which is crucial in paraphrase generation task. We denote it as *RS*:

$$RS(s^a, s^b) = RandomSample(set(s^a) - set(s^b)) \tag{12}$$

where *RS* takes two sentences as inputs and the minus operator is the set subtraction.

We combine the results of two methods:

$$\begin{aligned} k^s &= \{PR(s^s), RS(s^s, s^t)\} \\ k^t &= \{PR(s^t), RS(s^t, s^s)\} \end{aligned} \tag{13}$$

where we reorder the results by the positions they appeared.

2) LOSS

The target is to optimize a multi-task joint probability of word copying, word generation, and gate prediction.

$$p(y_t, z_t) = p(y_t|z_t) \cdot p(z_t) \tag{14}$$

where y_t is the target word, either predicted by copy-mode or by write-mode, z_t is the mode-gate.

To incorporate copy mechanism, we set:

$$\begin{aligned} p(y_t|z_t = 1) &= p_{copy}(y_t) \\ p(y_t|z_t = 0) &= p_{write}(y_t) \end{aligned} \tag{15}$$

Thus,

$$\begin{aligned} &\log p(y_t, z_t) \\ &\quad = \log p(y_t|z_t) + \log p(z_t) \\ &\quad = z_t^* \cdot \log p_{copy}(y_t) + (1 - z_t^*) \cdot \log p_{write}(y_t) \\ &\qquad + z_t^* \cdot \log p_{gate} + (1 - z_t^*) \cdot \log(1 - p_{gate}) \end{aligned} \tag{16}$$

where z_t^* denotes whether the target word is in keywords sequence or not.

In every time step, we take one token from $\langle s, k, t\rangle$ triplet. So, in every mini-batch:

$$loss_{tri}(s, k, t) = -\frac{1}{N \times T} \sum_{i=1}^{N} \sum_{t=1}^{T} \log p(y_t^i, z_t^i) \tag{17}$$

where N is the batch size, T is the maximum timestep, s is the source sentences batch, k is the keywords batch, t is the target sentences batch.

KCN should output different paraphrases according to different keywords, while holding the same meaning of source sentence. Thus, we introduce autoencoder mode and paraphrase mode in training stage. At each train step, KCN takes either $\langle s^s, k^s, s^s\rangle$ or $\langle s^s, k^t, s^t\rangle$ randomly. That is, in autoencoder-mode, KCN takes $\langle s^s, k^s\rangle$ to reconstruct s^s. In paraphrase-mode, KCN takes $\langle s^s, k^t\rangle$ to generate s^t. Thus, the target of KCN in the training stage is:

$$KCN(s^s, k^s) = s^s \tag{18}$$

$$KCN(s^s, k^t) = s^t \tag{19}$$

where (18) is the autoencoder-mode target, (19) is the paraphrase-mode target.

The final loss is

$$loss = \begin{cases} loss_{tri}(s^s, k^s, s^s), & \text{if } r \le threshold \\ loss_{tri}(s^s, k^t, s^t), & \text{otherwise} \end{cases} \tag{20}$$

where r is a random number, *threshold* is the hyperparameter which determines the probability of autoencoder mode. The loss calculated are back-propagated to all layers of the model, whose gradients are used to optimize the parameters.

IV. EXPERIMENT

We design two complementary experiments–automatic evaluation and human evaluation.

Automatic evaluations include: (a) calculating the BLEU, METEOR and TER scores. (b) calculating the keywords coverage. The three scores stem from machine translation, measuring how much the generated sentences and target sentences are alike. The coverage of keywords reflects how KCN follows the keywords hint at the macro level. However, the paraphrase can be facially different in many ways, thus, some of the metrics in machine translation, such as BLEU, is not good enough to evaluate the generated paraphrases. In addition, although many works have been done toward automatically paraphrase identification, these models suffer from the problem of generalization and interpretability. Thus, we also use fine-grained human evaluations. They include: (a) grammar correctness and paraphrase identification (b) user-controlled paraphrase generation.

A. DATASETS

We use three prevalent large-scale datasets, PARANMT50M, MSCOCO[2] [45] and Quora.[3]

PARANMT50M is a large-scale paraphrase corpus generated by back-translation [25]. The sentences in the dataset is closer to what we use in real world. However, the property of back-translation indicates that there are limited kinds of paraphrase and little information loss. In other words, most of the sentence pairs in PARANMT50M are word-phrase level paraphrases. Thus, it is hopeless to learn some difficult types of paraphrase in this dataset.

MSCOCO is a widely used dataset in computer vision, especially for image caption and visual question answering. For image caption, five annotators describe the image with a sentence in their own ways, which can be seen as paraphrases. This dataset has been used for many paraphrase generation models [9], [10]. We follow their works, using common machine translation metrics to evaluate our model.

Quora Duplicate Question is used for sentence representation tasks and paraphrase generation in [11]. The dataset contains paraphrase sentence pairs which are annotated by human. Note that most of the sentences are questions.

To sum up, sentences in PARANMT50M lack syntactical and pragmatical diversities while those in MSCOCO and Quora Duplicate Questions lack some real-world patterns.

PARANMT50M is more generalized than other datasets for KCN. Firstly, MSCOCO and Quora are full of monotonous sentence patterns, such as *a cat and a dog* in MSCOCO and *What is Deep Learning*, which hinder syntactical paraphrase learning. Secondly, there are more detailed information mismatch in these two data set because of **pragmatic paraphrase**.[4] For example, In MSCOCO, the annotators can realize that other sentences refer to the same image, no matter if subtle information lost. Because of the asymmetric information of paraphrase pairs, the support of users become more important. By means of the users' hints, the KCN is able to complete the paraphrase with more details, such as the properties of entities. And the structure of paraphrase can also be modified by exchanging the order of keywords.

[2]https://github.com/iamaaditya/neural-paraphrase-generation/

[3]https://www.kaggle.com/c/quora-question-pairs

[4]Linguists call this phenomenon pragmatic paraphrase because the meanings of the sentences all refer to the same fact.

TABLE 1. Quantitative evaluation on MSCOCO dataset.

Model	BLEU	METEOR	TER
Source2Target	7.6	16.6	85.2
Seq2Seq	33.4	25.2	53.8
Residual LSTM	37.0	27.0	51.6
VAE-SVG	41.7	31.0	40.8
DGEN	42.6	31.3	-
KCN	**48.5**	**39.1**	**32.7**

B. EXPERIMENTAL SETUP

We used the PositionRank implemented by [46]. The hyperparameters were tuned on the validation set. In the training stage, the length of keywords was limited to $[2, length(target)/2]$, which should not lose too much valuable information while keeping the basic style. The teacher-forcing ratio in the decoder was 0.5. All of the word embeddings were shared by the whole network, include two encoders and one decoder. We set the dimension of embedding as 300, GRU as 300 per direction, batch size as 64 and learning rate as 0.001. Thus, when we concatenate two directions of two encoders, the vector dimension was 1200. Then the vector was constrained to 300 by an affine transformation. We used Adam [47] as optimizer.

The *threshold* in (20) was equal to 0.3 which controls autoencoder-mode and paraphrase-mode. In the experiment, if *threshold* is too high, the model will be hard to follow the keywords other than those in source sentence and generates $\langle blank \rangle$ repetitively. If *threshold* is too low, the model would ignore the meaning of the source sentence and fully depend on the keywords.

C. AUTOMATIC EVALUATION

We design two automatic experiments. 1) How similar are the outputs of KCN and the references? 2) How many keywords are covered in the outputs?

How similar are the outputs of KCN and the references?

Residual LSTM [9], VAE-SVG [10], DGEN [31] have evaluated their models by BLEU, METEOR and TER.[5] on MSCOCO dataset[6] As there are no public test set in PARANMT50M and Quora, we do not calculate scores of these datasets. The result of automatic evaluation on MSCOCO is in Table 1, where Source2Target calculates the

[5]Higher BLEU, METEOR are better and lower TER is better. We use the open source tool to calculate these metrics. https://github.com/jhclark/multeval

[6]We directly use the scores published on their papers.

TABLE 2. Examples of the result train on PARANMT50M, MSCOCO and Quora. KCN uses keywords line to generate the paraphrase line. The last line of PARANMT50M and MSCOCO shows some failed examples cause by some equivocal keywords. s is the source sentence, r is reference sentence, k is the keywords and p is the generated paraphrases.

	PARANMT50M	MSCOCO	Quora
s	⟨unk⟩ is the gentlest soul in this whole company.	a cell phone next to a book and on a white pillow	How do I see a private Instagram account?
r	⟨unk⟩ is the finest soul in the company.	a book and cell phone laying on a pillow	Can I see a private Instagram?
k	finest soul company	book cell phone pillow	Can private
p	⟨unk⟩ is the finest soul in the company	a book and a cell phone on a pillow	Can I see my private Instagram account?
s	They 're all humans.	a person using a computer that is hooked up to a tv	How do I hire an ethical hacker?
r	It's all people.	a person with using a laptop and a television screen	Where can I find/employ a real hacker?
k	people	person laptop television screen	a real
p	They 're all people.	a person using a laptop on a television screen	How do I hire a real hacker?
s	i need more than 10.	a man is snuffing a small pan pizza	What are a list of amazing books?
r	you have to **give** me more than $ 10.	a young man **looks** down at two pizzas	What is a list of good books?
k	to **give** $	young **looks** pizzas	list good
p	i need to **give up** more than 10.	a young man **looks like** two pizzas	What are a list of good books?

score directly using $\langle s^s, s^t \rangle$ in the dataset. Note that the target sentence is also a valid paraphrase of the source sentence, so it should have had a high score in automatic evaluation. However, its BLEU score is the lowest, 7.6. Besides, we can see that KCN is the best model and outperforms others by a huge margin. Note that DGEN uses external knowledge (i.e. PPDB dictionary [48]), and KCN uses keywords hints extracted from the target sentence, that might explain the superiority of the methods. Here the experiment result bears out our hypothesis: KCN has the capability of capturing the guidance of keywords and generating corresponding paraphrases. However, a notable drawback is presented in the last line of Table 2. A slight alteration of the generated sentence would change the meaning completely. This is hard to evaluate by the metrics aforementioned.

How many keywords are covered in the outputs?

KCN does not promise to cover every keyword in the generated paraphrases. We calculate the *keywords coverage* and the *length ratio*. Keywords coverage means how many keywords are covered in the outputs. And the length ratio is calculated by the length of keywords divided that of the outputs. The results for three datasets are in Table 3.

We can see that the coverage of keywords in MSCOCO is the highest, which coincides with its limited patterns in syntax. Quora is harder to gain equally high keywords coverage, but it uses relative fewer keywords. PARANMT50M covers least keywords, due to its complex language diversity. The sentences in PARANMT50M are often much longer than other datasets, and characterized by complex syntax.

It would be better to try other extraction methods for comparison, however, we find that most of the methods cannot work with short text keywords extraction, and their time efficiencies are very low. To further evaluate the paraphrase quality and the influence of keywords selection and, we design two fine-grained human evaluations.

D. HUMAN EVALUATION AND ANALYSIS

Machine translation metrics are prevalent in paraphrase generation is because traditional machine translation metrics perform well on MSRP paraphrase identification task [49], with about 70% precision. However, its assumption that the sentences are correct is not proper in the paraphrase generation area. In Table 1 Source2Target, we calculated the BLEU, METEOR, and TER using the sentence pairs in the test set directly but the scores were 50% lower than others, while the pairs are paraphrases by definition. Moreover, in pragmatic paraphrase, crucial information loss is hard to evaluate by machine. For example, in Table 2 the first line, paraphrasing *in this whole company* to *in the company* should not be view as big information loss because it mentioned *the finest* before. But if the *in the company* is removed, it will be much harder to say the sentence pairs are paraphrase. Thus, human evaluation is of paramount significance. We firstly employ human evaluators to judge grammar correctness and paraphrase identification. Then, we perform an assistant experiment to judge if the model can generate different paraphrases with different keywords hints.

TABLE 3. Coverage and length ratio.

Dataset	Coverage	Length Ratio
PARANMT50M	0.56	0.54
MSCOCO	0.90	0.60
Quora	0.77	0.38

TABLE 4. Human evaluation on paraphrase identification and correctness. Paraphrases are judged only when they are correct.

Dataset	Correct	Paraphrase
PARANMT50M	0.70	0.72
MSCOCO	0.66	0.90
Quora	0.81	0.66

Human evaluation is costly but highly flexible. The evaluators are asked two questions: 1) Are the generated paraphrases grammatically correct and semantically the same? 2) Can KCN generate different paraphrases depending on different keywords hints?

Are the generated paraphrases grammatically correct and semantically the same?

We randomly select 100 pairs from the test set, run KCN model and gain $\langle s, r, p \rangle$ triplets. s is the source sentence, r is the reference sentence in the test set, p is the generated paraphrase. The evaluators are asked to judge the result from two aspects: grammar correction and paraphrase identification.

TABLE 5. The results of the assistant experiment. Given the source sentence, KCN generates different paraphrases depending on different keywords hints. Replacing and reordering the keywords enable KCN to generate reasonable results correspondingly.

source	keywords	outputs
a plane is taking off from the runway	a plane from runway	a plane taking off from a runway
	airplane flies runway into sky	an airplane flies over the runway into the sky
	airplane runway sky	an airplane taking off a runway with a
	airplane sky	an airplane is taking off the sky
white refrigerator and a white and black microwave	refrigerator microwave	a white refrigerator and a microwave
	microwave refrigerator	a microwave and a refrigerator are on
	microwave white refrigerator	a microwave and a white refrigerator
	microwave small refrigerator	a microwave is shown with a small white refrigerator
	refrigerator green microwave	a refrigerator with a green microwave

The results of this experiment are in Table 4. Note that the paraphrase score of MSCOCO is much higher than that of PARANMT50M and Quora. As we discussed above, MSCOCO is full of pragmatic paraphrase which enables slight information mismatches. Thus, here the evaluators take loose standard to identify paraphrase in MSCOCO. They are asked to imagine the image depicted by the source sentence, then judge whether the generated paraphrases refer to the same thing or not. The results show that KCN learns to ignore the inessential information which is not included in the keywords and to reorder the information according to keywords hint, such as the first two lines in Table 2. In PARANMT50M, KCN are more likely to learn word-phrase level substitution rather than syntactical changes. This is likely stems from the inner properties in the training set: PARANMT50M is constructed by back-translation containing limited syntactical diversities. Thus, the experiments show that KCN is capable of learning the paraphrase properties in each dataset and generating paraphrases with high quality.

Can KCN generate different paraphrases depending on different keywords hints?

We also perform an assistant experiment. Given a source sentence, a real-world user extracts the keywords from it and modifies the keywords to different forms. The results are shown in Table 5. As we can see, KCN can not only correctly complete the phrase related to new keywords, but also generate proper paraphrase according to reordered keywords. In the first example, the user simply substitutes *taking off* to *flies*, then the KCN autocompletes the remained preposition *over* and *into*. In second example, *white refrigerator and a white and black microwave*, the user reorders the keywords, *microwave white refrigerator*. We can observe that the model generates *a microwave and a white refrigerator* corresponding to the new keywords order. Moreover, if we change some inessential keywords, such as adjective word *black and white* to *green*, the new keywords will appear in the proper position, which means that KCN is capable of generating more than paraphrase. Thus, the experiment results exactly match our hypothesis: KCN tends to accept the hints from keywords, and to generate corresponding outputs. This shows its potentials to be used as an interactive writing assistant for real-world users.

V. DISCUSSION

A. RELATION TO CONSTRAINED DECODING

The appearances of the keywords in the paraphrases generated by KCN are soft-constrained. In other words, KCN does not ensure that all keywords are contained in the outputs. There are also many hard-constrained decoding methods in machine translation [35], [36], image caption [34]. They all explored complicated beam-search techniques. The drawbacks of them lie in two aspects: 1. These ad-hoc decoding tricks are time-consuming. While their dependency on beam-search results in $O(Nk)$ or $O(NkC)$ complexity, that of our model is $O(N)$ which does not use any additional search in decoding, where N is the length of sentences, k is beam size and C is the number of constraints. 2. Since the semantics of the word constraints are all processed in the decoder part, the outputs might not be natural due to lack of the participation of the encoder. Different from theirs, KCN encodes the semantics of the keywords and fuse them with the representation of the input sentence. By this way, the model can adjust the outputs flexibly or even withdraw some of the improper keywords. Moreover, it decodes with greedy search or vanilla beam search, which are the most popular and effective techniques in the current NLP community.

B. KEYWORDS IN KCN

There can be many variants in KCN, especially the keywords extraction and the decoder.

Note that the keywords extraction strategy is only adopted in the training stage for simulating users' behaviors. In practice, we use the combination of PositionRank and RandomSample for keywords extraction. PositionRank can extract keywords independently while RandomSample extracts words considering the references. However, there exist some sentences which break both methods. For example, when PositionRank extracts no keywords from the sentence, RandomSample still works and does not leave the keywords input empty. In contrast, if the model only depends on RandomSample, it might miss the informative words, and make the outputs confusing. Better keywords extraction, especially for paraphrase generation task, remains future works.

The KCN decoder only copies words from the keywords sequence. And it is also not hard to implement a copy decoder

which considers both keywords and source sentence. However, this variant degrades the learning process as it adds bias to KCN: most of the output words are directly copied but seldom will it generate from the whole dictionary.

VI. CONCLUSION AND FUTURE WORKS

In this paper, we propose a Keywords Controlled Network for paraphrase generation. The generation processes of extracting keywords, modifying keywords, and generating paraphrase conform to the real-world paraphrase scenario: TOEFL writing. The experiment shows that KCN is able to generate multiple paraphrases with high quality, which outperforms the state-of-the-art models. And the human evaluation shows that KCN is capable of generating paraphrases controlled by users. Thus, KCN can be used not only as a great data pre-processing module for NLP applications but also as a writing assistant serving the real world users.

In the future, following directions are to be explored:

- The coverage score of the KCN is low because the soft constraint remove the ''not proper'' keywords. In the future, we will study better ways to capture the meanings of the keywords given the source sentence.
- It is better of giving KCN some choices for each keyword, in order to maximize the fluency. In the current setting, when KCN cannot figure out how to fuse the keywords into parapharses, it simply reject them.
- Training an end-to-end joint model of keywords extraction and paraphrase generation is beneficial to both task. By this way, the keywords extracted will be more related to paraphrase task, containing syntactical and stylistic information.

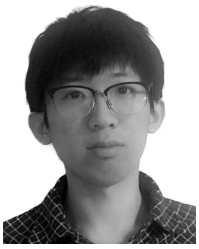
REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent, ''Representation learning: A review and new perspectives,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[2] N. Madnani and B. J. Dorr, ''Generating phrasal and sentential paraphrases: A survey of data-driven methods,'' *Comput. Linguistics*, vol. 36, no. 3, pp. 341–387, 2010.
[3] A. Fader, L. Zettlemoyer, and O. Etzioni, ''Open question answering over curated and extracted knowledge bases,'' in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 1156–1165.
[4] Y. Hou, Y. Liu, W. Che, and T. Liu, ''Sequence-to-sequence data augmentation for dialogue language understanding,'' Jul. 2018, *arXiv:1807.01554*. [Online]. Available: https://arxiv.org/abs/1807.01554
[5] I. A. Bolshakov and A. Gelbukh, ''Synonymous paraphrasing using WordNet and Internet,'' in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.* Berlin, Germany: Springer, 2004, pp. 312–323.
[6] K. R. McKeown, ''Paraphrasing questions using given and new information,'' *Comput. Linguistics*, vol. 9, no. 1, pp. 1–10, 1983.
[7] A. Fujita, ''Automatic generation of syntactically well-formed and semantically appropriate paraphrases,'' Ph.D. dissertation, Dept. Inf. Process., Nara Inst. Sci. Technol., Ikoma, Japan, 2005.
[8] M. Vila, M. A. Martí, and H. Rodríguez, ''Is this a paraphrase? What kind? Paraphrase boundaries and typology,'' *Open J. Mod. Linguistics*, vol. 4, no. 1, pp. 205–218, Mar. 2014.
[9] A. Prakash, S. A. Hasan, K. Lee, V. Datla, A. Qadir, J. Liu, and O. Farri, ''Neural paraphrase generation with stacked residual LSTM networks,'' Oct. 2016, *arXiv:1610.03098*. [Online]. Available: https://arxiv.org/abs/1610.03098
[10] A. Gupta, A. Agarwal, P. Singh, and P. Rai, ''A deep generative framework for paraphrase generation,'' Sep. 2017, *arXiv:1709.05074*. [Online]. Available: https://arxiv.org/abs/1709.05074
[11] Z. Li, X. Jiang, L. Shang, and H. Li, ''Paraphrase generation with deep reinforcement learning,'' Nov. 2017, *arXiv:1711.00279*. [Online]. Available: https://arxiv.org/abs/1711.00279
[12] Z. Cao, C. Luo, W. Li, and S. Li, ''Joint copying and restricted generation for paraphrase,'' in *Proc. 31st AAAI Conf. Artif. Intell.*, Feb. 2017, pp. 3152–3158.
[13] S. Ma, X. Sun, W. Li, S. Li, W. Li, and X. Ren, ''Query and output: Generating words by querying distributed word representations for paraphrase generation,'' in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2018, pp. 196–206.
[14] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, ''Adversarial example generation with syntactically controlled paraphrase networks,'' Apr. 2018, *arXiv:1804.06059*. [Online]. Available: https://arxiv.org/abs/1804.06059
[15] I. Sutskever, O. Vinyals, and Q. V. Le, ''Sequence to sequence learning with neural networks,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
[16] J. Zhang, X. Jin, J. Sun, J. Wang, and A. K. Sangaiah, ''Spatial and semantic convolutional features for robust visual object tracking,'' *Multimedia Tools and Applications*. New York, NY, USA: Springer, 2018.
[17] L. Xiang, X. Shen, J. Qin, and W. Hao, ''Discrete multi-graph hashing for large-scale visual search,'' *Neural Process. Lett.*, vol. 49, no. 3, pp. 1055–1069, Jul. 2018. doi: 10.1007/s11063-018-9892-7.
[18] D. Zeng, Y. Dai, F. Li, J. Wang, and A. K. Sangaiah, ''Aspect based sentiment analysis by a linguistically regularized CNN with gated mechanism,'' *J. Intell. Fuzzy Syst.*, vol. 36, no. 5, pp. 3971–3980, May 2019.
[19] D. Zeng, Y. Dai, F. Li, R. S. Sherratt, and J. Wang, ''Adversarial learning for distant supervised relation extraction,'' *Comput., Mater. Continua*, vol. 55, no. 1, pp. 121–136, 2018.
[20] W. Li, Z. Chen, X. Gao, W. Liu, and J. Wang, ''Multi-model framework for indoor localization under mobile edge computing environment,'' *IEEE Internet Things J.*, to be published.
[21] I. Androutsopoulos and P. Malakasiotis, ''A survey of paraphrasing and textual entailment methods,'' *J. Artif. Intell. Res.*, vol. 38, pp. 135–187, May 2010.
[22] W. Dolan and C. Brockett, ''Automatically constructing a corpus of sentential paraphrases,'' in *Proc. 3rd Int. Workshop Paraphrasing (IWP)*, Jan. 2005, pp. 9–16.
[23] A. Barrón-Cedeño, M. Vila, M. A. Martí, and P. Rosso, ''Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection,'' *Comput. Linguistics*, vol. 39, no. 4, pp. 917–947, Dec. 2013.
[24] J. Wieting and K. Gimpel, ''ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations,'' in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Jul. 2018, pp. 451–462.
[25] J. Wieting, J. Mallinson, and K. Gimpel, ''Learning paraphrastic sentence embeddings from back-translated bitext,'' Jun. 2017, *arXiv:1706.01847*. [Online]. Available: https://arxiv.org/abs/1706.01847
[26] P. W. Culicover, ''Paraphrase generation and information retrieval from stored text,'' *Mech. Transl. Comp. Linguistics*, vol. 11, nos. 1–2, pp. 78–88, Mar./Jun. 1968.
[27] B. Pang, K. Knight, and D. Marcu, ''Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences,'' in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, May 2003, pp. 102–109.
[28] C. Quirk, C. Brockett, and W. Dolan, ''Monolingual machine translation for paraphrase generation,'' in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Jul. 2004, pp. 142–149.
[29] J. Gu, Z. Lu, H. Li, and V. O. K. Li, ''Incorporating copying mechanism in sequence-to-sequence learning,'' in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Aug. 2016, pp. 1631–1640.
[30] O. Vinyals, M. Fortunato, and N. Jaitly, ''Pointer networks,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2692–2700.
[31] S. Huang, Y. Wu, F. Wei, and M. Zhou, ''Dictionary-guided editing networks for paraphrase generation,'' Jun. 2018, *arXiv:1806.08077*. [Online]. Available: https://arxiv.org/abs/1806.08077
[32] G. Foster and G. Lapalme, *Text Prediction for Translators*. Montreal, QC, Canada: Université de Montréal, 2002.
[33] S. Barrachina, O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. Lagarda, H. Ney, J. Tomás, E. Vidal, and J.-M. Vilar, ''Statistical approaches to computer-assisted translation,'' *Comput. Linguistics*, vol. 35, no. 1, pp. 3–28, 2009.
[34] P. Anderson, B. Fernando, M. Johnson, and S. Gould, ''Guided open vocabulary image captioning with constrained beam search,'' in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Sep. 2017, pp. 936–945.

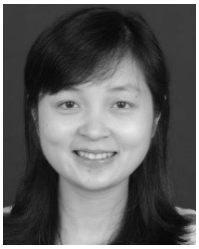
[35] C. Hokamp and Q. Liu, "Lexically constrained decoding for sequence generation using grid beam search," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Jul. 2017, pp. 1535–1546.
[36] M. Post and D. Vilar, "Fast lexically constrained decoding with dynamic beam allocation for neural machine translation," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2018, pp. 1314–1324.
[37] K. Cho, B. Van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," Sep. 2014, *arXiv:1409.1259*. [Online]. Available: https://arxiv.org/abs/1409.1259
[38] X. Zeng, D. Zeng, S. He, K. Liu, and J. Zhao, "Extracting relational facts by an end-to-end neural model with copy mechanism," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Jul. 2018, pp. 506–514.
[39] W. Kryściński, R. Paulus, C. Xiong, and R. Socher, "Improving abstraction in text summarization," Aug. 2018, *arXiv:1808.07913*. [Online]. Available: https://arxiv.org/abs/1808.07913
[40] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," Sep. 2014, *arXiv:1409.0473*. [Online]. Available: https://arxiv.org/abs/1409.0473
[41] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 971–980.
[42] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," Feb. 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167
[43] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," Jul. 2016, *arXiv:1607.06450*. [Online]. Available: https://arxiv.org/abs/1607.06450
[44] C. Florescu and C. Caragea, "PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, Jul. 2017, pp. 1105–1115.
[45] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.
[46] F. Boudin, "pke: An open source python-based keyphrase extraction toolkit," in *Proc.26th Int. Conf. Comput. Linguistics, Syst. Demonstrations*, Osaka, Japan, Dec. 2016, pp. 69–73. [Online]. Available: https://www.aclweb.org/anthology/C16-2015
[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," Dec. 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980
[48] E. Pavlick, J. Bos, M. Nissim, C. Beller, B. Van Durme, and C. Callison-Burch, "Adding semantics to data-driven paraphrasing," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process.*, vol. 1, Jul. 2015, pp. 1512–1522.
[49] N. Madnani, J. Tetreault, and M. Chodorow, "Re-examining machine translation metrics for paraphrase identification," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.* Montreal, QC, Canada: Association for Computational Linguistics, 2012, pp. 182–190.

DAOJIAN ZENG was born in Deyang, Suchuan, China, in 1985. He received the B.S. and M.S. degrees from the School of Electronic and Information Engineering, Soochow University, in 2011, and the Ph.D. degree from the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (IACAS), in 2015.

Since 2015, he has been a lecturer with the School of Computer and Communication Engineering, Changsha University of Science and Technology. His research interests include natural language processing and information extraction. Since 2011, he has published several papers in the leading journals and top conferences, such as ACL, EMNLP, COLING, and IJCAI. He was a recipient of the Best Paper Award in COLING 2014 and CCL 2017. He has served on several program committees of the major international conferences in the field of natural language processing, and also served as a Reviewer for several journals.



HAORAN ZHANG was born in Tianjin, China, in 1996. He received the B.S. degree from the Changsha University of Science and Technology, Changsha, Hunan, China, in 2018. He is currently pursuing the M.S. degree in information management with the University of Illinois Urbana–Champaign, in 2019. His research interests include natural language processing, relation extraction, and weak supervised learning.



LINGYUN XIANG received the B.E. degree in computer science and technology and the Ph.D. degree in computer application from Hunan University, Hunan, China, in 2005 and 2011, respectively. She is currently a Lecturer with the School of Computer and Communication Engineering, Changsha University of Science and Technology. Her research interests include information security, steganography, steganalysis, machine learning, and pattern recognition.

JIN WANG received the B.S. and M.S. degrees from the Nanjing University of Posts and Telecommunications, China, in 2002 and 2005, respectively, and the Ph.D. degree from Kyung Hee University, South Korea, in 2010. He is currently a Professor with the School of Computer and Communication Engineering, Changsha University of Science and Technology. He has published more than 200 international journal and conference papers. His current research interests include wireless communication and networking, performance evaluation, and optimization. He is a member ACM.

GUOLIANG JI was born in Huanggang, Hubei, China, in 1987. He received the B.S. degree in information and computing science from Three Gorges University, Yichang, Hubei, in 2010, and the M.S. and Ph.D. degrees in computer science and technology from the Chinese Academy of Sciences, Beijing, China, in 2017.

From 2017 to 2019, he was a Researcher with the Water Resources Technology Research Center, China Three Gorges Corporation. He has authored a one book and more than ten articles. His research interests include machine learning, deep learning, and knowledge graphs in water resources.