Received 11 February 2023, accepted 17 March 2023, date of publication 23 March 2023, date of current version 29 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3260849

RESEARCH ARTICLE

Paraphrase Generation Model Integrating Transformer Architecture, Part-of-Speech Features, and Pointer Generator Network

YU-CHIA TSAI AND FENG-CHENG LIN
Department of Information Engineering and Computer Science, Feng Chia University, Taichung 407, Taiwan

Corresponding author: Feng-Cheng Lin (fclin@fcu.edu.tw)

This work was supported by the National Science and Technology Council, Taiwan, under Grant NSTC111-2121-M-035-001 and Grant NSTC 112-2420-H-006 -002.

ABSTRACT In recent years, hardware advancements have enabled natural language processing tasks that were previously difficult to achieve due to their intense computing requirements. This study focuses on paraphrase generation, which entails rewriting a sentence using different words and sentence structures while preserving its original meaning. This increases sentence diversity, thereby improving the performance of downstream tasks, such as question–answering systems and machine translation. This study proposes a novel paraphrase generation model that combines the Transformer architecture with part-of-speech features, and this model is trained using a Chinese corpus. New features are incorporated to improve the performance of the Transformer architecture, and the pointer generation network is used when the training data contain low-frequency words. This allows the model to focus on input words with important information according to their attention distributions.

INDEX TERMS Multi-encoder, paraphrase generation, pointer generation network, transformer.

I. INTRODUCTION

With the rise of the attention mechanism and the Transformer architecture [1], natural language processing applications have become more widespread. Generative tasks related to natural language processing include paraphrasing. Paraphrasing involves changing the structure and words of a sentence without changing its meaning. Samples of paraphrased text are listed in Table 1. Several downstream applications are supported by paraphrasing. Nakov [2] uses the syntactic tree for paraphrase generation, which augments the existing training data set to enhance translation quality. Li et al. [3] proposed the use of paraphrasing to improve the efficiency of question and answering systems in reading comprehension. Question–answering systems are sensitive to minor changes in input. The stability of question–answering systems can be improved by expanding their corpora. Gao et al. [4] proposed improving the quality of dialogue system responses through data augmentation. Research on paraphrasing has become more diverse over time. Paraphrasing has utility in several applications.

The associate editor coordinating the review of this manuscript and approving it for publication was Yen-Lin Chen.

TABLE 1. Samples of paraphrased chinese text.

source	寶寶吐奶後怎麼處理
target	嬰兒吐奶了怎麼辦
source	家電要去哪買
target	家電哪裡有賣
source	學習英語下載什麼軟體好
target	學英語用什麼軟體比較好

Most research on paraphrasing utilizes English language corpora from sources including Twitter [5], Quora questions [6], and the Microsoft Research Paraphrase Corpus [7]. Few studies have used Chinese-language corpora. This study used a Chinese corpus and combined the characteristics of Chinese to improve the effectiveness of paraphrasing in Chinese. This study proposed a novel network architecture based on the Transformer neural network that uses a Chinese-language corpus. The model combines text and part-of-speech features because we believe that even the same word combination in a sentence will generate different part-of-speech modifications depending on the sentence structure. The aim of this study was to create a model that learns the information in the text and the difference caused by changes in parts of speech, thereby enabling the model to generate natural-sounding sentences. Furthermore, the Chinese language does not have the same rigid word boundaries as other languages, making it difficult to determine the structure of a sentence. To address this, we often utilize part of speech tagging as an indirect means of generating the complete sentence structure. The model also uses the Pointer Generator Network (PGN) [8] to handle out-of-vocabulary words. Each output must be generated from a predefined dictionary that may not contain low-frequency words. The PGN enables the proposed model to learn the information in the original sentence and retain the important information when generating a new sentence.

II. RELATED WORK

A. Seq2Seq

Prakash et al. [9] and Gupta et al. [10] have proposed applying the Seq2Seq model, which is based on Long Short-Term Memory (LSTM), for paraphrasing. Prakash et al. [9] noted that applying the residual network architecture [11] to the stacked LSTM network architecture was beneficial for training. Gupta et al. proposed combining the LSTM network architecture with Variational Auto-Encoder (VAE) [10], [12], which prevented excessive deviation from the original sentence in terms of semantic expression by adding constraints when generating sentences. Vaswani et al. [1] proposed a new Seq2Seq network architecture in 2017, which solved the translation problem by replacing the traditional recurrent neural network with the self-attention mechanism. The self-attention mechanism achieves parallelization by performing an inner product operation on the matrix Q, K, V, which cannot be achieved by the RNN algorithm. Additionally, positional encoding is utilized to preserve the position information of the entire sequence. This was the baseline model of our study. In subsequent experiments, the Transformer architecture and related models were used for performance comparison and analysis. Wang et al. proposed a new model architecture combining the Transformer architecture with the Multi-Encoder architecture and multiple features. The model proposed by Wang et al. [13] converted text into the derived feature data of FRAMES and ROLES through SLING [14], proposed by Ringgard et al., then fused the three feature values through the linear layer and subsequently generated the final result through the decoder.

B. POINTER GENERATOR NETWORK (PGN)

The PGN [15] was first proposed, it mainly used to solve the convex hull, Delaunay Triangulation, and Traveling Salesman Problem. Seq2Seq is unable to address the issue of the size of the output dictionary varying depending on the input. Therefore, See et al. [8] incorporated the attention mechanism into the model for determining the output according to the attention distribution of the input. This architecture is proposed as a solution to resolve the issues of repeated sentences and out-of-vocabulary words. PGN and a new loss function are also introduced to address these issues. Ravuru et al. [16] proposed combining VAE [12] and the PGN [8] and applying them to the task of paraphrasing. Zhang et al. [17] combined the Transformer architecture with the PGN. Zhang et al. suggested that writing product copy is similar to abstract extraction. Through this network architecture, a product copy containing important information can be generated.

C. METRICS

BLEU [18] is a metric for evaluating the quality of machine-translated text, and it is based on the vocabulary level. In this approach, the number of overlapping n-grams in the generated and reference texts is counted, and a penalty factor is added for shorter outputs. For example, when the input sentence is too short (which leads to an inflated score), the penalty factor is applied. METEOR [19] considers synonyms to expand WordNet, overcoming the problem of synonyms that BLEU cannot handle and solving the problem of similar results occurring under the same word stem. BERTscore [20] uses the BERT [21] pretraining model to convert each word of the input text and each word of the paraphrased text into a word vector; then, the cosine similarity is compared according to the word vector representation in the vector space. The higher is the cosine similarity of the vectors, the smaller is the angle between the word vectors as well as between the input text and the paraphrased text, and the closer is the meanings of the two sentences. Two aspects of paraphrasing should be evaluated. One is the preservation of the original semantics, and the other is the diversity of the output. In addition to proposing a framework for data collection, Chen et al. proposed the use of n-grams for calculating the difference between the input and paraphrased sentences to evaluate the diversity of paraphrasing [22]. Various evaluation metrics have been proposed, but they only evaluate the quality of generated sentences from a single aspect. Shen et al. proposed a new evaluation metric called ParaScore [23]. ParaScore combines the BERTscore and normalized edit distance (NED). NED can be used to judge the diversity of sentence generation by calculating how many times two sentences need to be edited. Using these two evaluation metrics, ParaScore [23] denotes the two important aspects of paraphrased sentence: whether the semantics are preserved and whether the generated sentences are diverse.

III. PROPOSED METHOD

A. DATA PREPROCESSING

In the data preprocessing stage, the text data of the training data set are converted into numerical data that can be used for model training. The corpora used in this paper are the Simplified Chinese text data set of LCQMC [24] and the Phoenix Paraphrasing dataset [25]. The Simplified Chinese text is first converted into Traditional Chinese in OpenCC [26]. The basic unit of text is the word in this study.

TABLE 2. Part-of-speech tagging example.

source	寶寶吐奶後怎麼處理
POS	寶寶(Na) 吐(VC) 奶(Na) 後(Ng) 怎麼(D) 處理(VC)

The CKIP tagger [27] was used for part-of-speech tagging and for generating data about the parts of speech. An example of part-of-speech tagging is shown in Table 2. According to the length of each tokenized word, the corresponding part-of-speech is set to the same length. Thus, the length of the text data is consistent with the length of the part-of-speech data. Specific nouns do not require special treatment, as the PGN allows the model to preserve their information during output generation. The length of the output is not fixed; instead, it continues until the model produces an End-Of-Sequence (EOS) character to signal the completion of the output. Table 3 shows the entire data preprocessing stage and transformation of features at each stage.

TABLE 3. Data preprocessing example description.

source	寶寶吐奶後怎麼處理
add token	<BOS>寶寶吐奶後怎麼處理<EOS>
transform	[0,42,42,164,13,77,34,30,448,10,1]
source	寶寶吐奶後怎麼處理
POS	Na VC Na Ng D VC
padding	Na Na VC Na Ng D D VC VC
add token	<BOS>Na Na VC Na Ng D D VC VC<EOS>
transform	[0,5,5,7,5,13,21,21,19,19,1]

B. MODEL

The proposed architecture is shown in Fig. 1. The architecture can be described as follows. The Transformer architecture

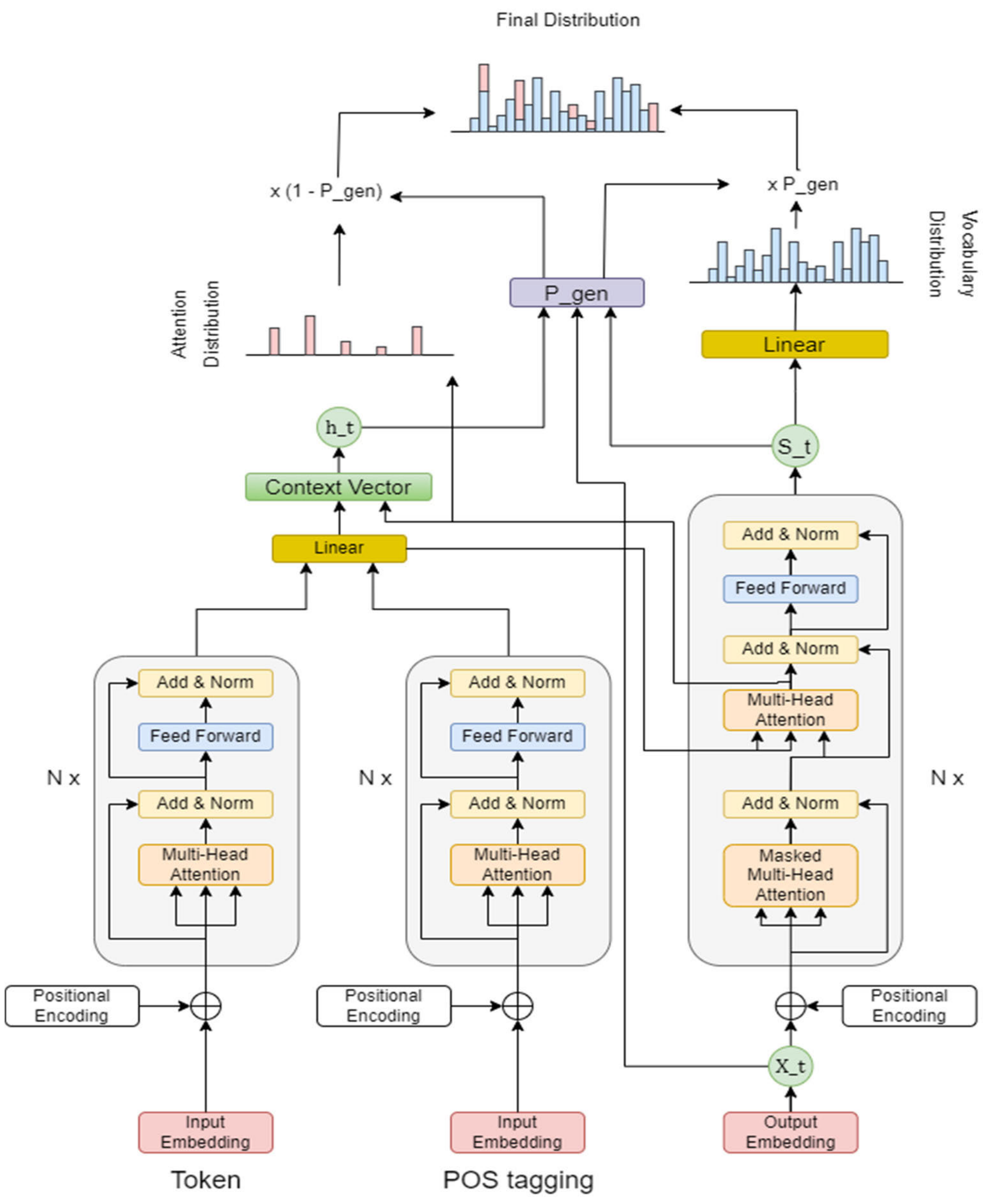

FIGURE 1. Proposed architecture.

has an encoder–decoder structure [1]. The model consists of three main components. One is an encoder for the token, another is an encoder for the part-of-speech, and the other is a decoder. Finally, the attention distribution of the source and the probability distribution of all words in the dictionary are weighted according to P_{gen}.

To better focus on similar concepts, we refer to the writing style of the main literature [1] and use a unique number to represent formulas that share the same concept. First, we convert the token into input embedding and add positional encoding. The formula for this step is as follows:

$$\begin{aligned} PE_{(pos,2i)} &= \sin(pos/10000^{2i/d_{model}}) \\ PE_{(pos,2i+1)} &= cos(pos/10000^{2i/d_{model}}) \end{aligned} \tag{1}$$

where pos is the position of the sequence and ith word embedding in a given dimension ($\boldsymbol{d}_{model}$ is 512 in this study, and i ranges from 0 to 511). An example is shown in Fig. 2.

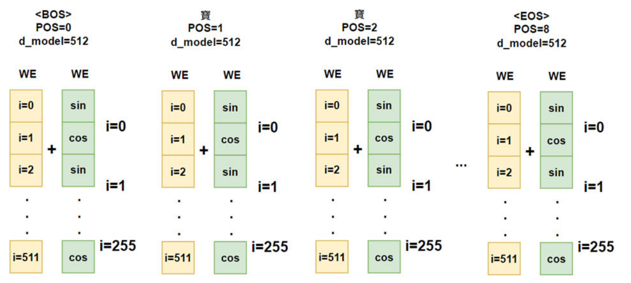

FIGURE 2. Example of positional encoding.

Next, according to W_Q, W_K *and* W_V, we convert sequence into matrices Q, K and V.

$$\begin{aligned} Q &= W_Q\,(WE) \\ K &= W_K\,(WE) \\ V &= W_V\,(WE) \end{aligned} \tag{2}$$

where matrices W_Q, W_k, *and* W_v are learnable parameters. Next, Q, K, and V are divided into h heads, and a single attention operation corresponding to Q, K, and V is performed in each head.

$$\begin{aligned} Attention\,(Q, K, V) &= softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \\ MultiHead(Q, K, V) &= Concat(head_1, \ldots, head_h)W^O \end{aligned} \tag{3}$$

$$\text{Where} \quad head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{4}$$

where d_k is K of dimension and matrices W_i^Q, W_i^K and W_i^V are learnable parameters. Next, the operation result of multi-head attention or fully connected feed-forward network (FFN) is passed through a residual operation, and a hidden state for the token is combined and generated using layer normalization.

$$Add\&Norm = LayerNorm(x + Sublayer(x)) \tag{5}$$

$$FFN\,(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{6}$$

where $Sublayer(x)$ is the result of multi-head attention or FFN, and matrices W_1 and W_2 and scalars b_1 and b_2 are learnable parameters. The encoder for part-of-speech does the same operation as the encoder for text. The results generated by the two encoders are concatenated and passed through a linear layer to generate a hidden state for the part-of-speech. A sentence will have different parts of speech depending on the sentence structure. This approach enables the model to learn to correctly understand the sentence structure based on the part-of-speech information.

$$HS_{fusion} = W_{fusion}\left(concat\left(HS_{toekn}\,HS_{postagging}\right)\right) + b \tag{7}$$

where matrix W_{fusion} and scalar b are learnable parameters. Due to the attention mechanism, the model receives all words as input in one step, which causes the decoder to simultaneously obtain all the information of the timestep. Therefore, the decoder uses mask tags to hide subsequent information, thereby preventing the model from receiving information after time t and affecting the decoding status. The masking process is illustrated in Fig. 3.

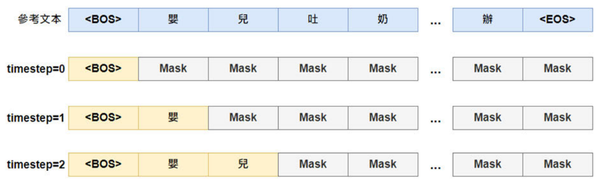

FIGURE 3. Example of mask used in decoder.

In the decoding stage, according to the attention distribution of the encoder at different timesteps, the attention scores of each head are summed, and the obtained value is the attention distribution of the decoder at the current timestep for the fusion of part-of-speech features and tokens.

$$a_t = \sum_{i}^{n} head_i \tag{8}$$

where $head_i$ represents different heads, t represents different timesteps, and a_t represents the attention distribution in different timesteps. The characteristics of the PGN can be used by the model to understand the structure of the sentence input according to the hidden state containing part-of-speech features, preserving important features when outputting the results. In addition to the probability of decoder vocabulary, attention distribution is used to strengthen the output. P_{gen} is based on the combination of the context vector, word embedding of the target sentence, and decoder output. According to Formula (7), the sum of HS and attention distribution are weighted to obtain the context vector h_t^*. The process of calculating the context vector value is shown in Fig. 4.

$$h_t^* = \Sigma_i a_i^t HS_{fusion} \tag{9}$$

where t represents a different timestep.

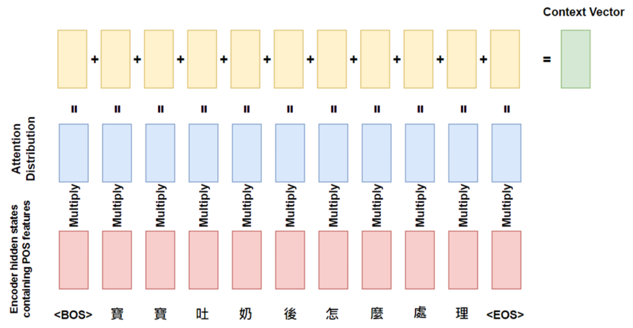

FIGURE 4. **Calculation of context vector value.**

Next, P_{gen} is calculated from h_t^*, the target sequence word embedding, and the output of the decoder.

$$P_{gen} = \sigma(w_{h^*}^T h_t^* + w_s^T S_t + w_x^T x_t + b_{ptr}) \tag{10}$$

where σ is the sigmoid function and matrices $w_{h^*}^T$, w_s^T, and w_x^T and scalar b_{ptr} are learnable parameters. The value of P_{gen} ranges between 0 and 1.

The result generated by the decoder is converted according to the size of the vocabulary, weighted with P_{gen}, and added to the probability of the input sequence. P_{gen} is a weight factor that has different weight assignments according to the timestep. Based on P_{gen}, the model decides whether to copy from the input sequence according to the attention distribution or generate a token from the dictionary.

$$P(w) = P_{gen} P_{vocab}(w) + (1 - P_{gen}) \Sigma_{i:w_i=w} a_i^t \tag{11}$$

$P_{vocab}(w)$ represents the probability of the next token in the dictionary, $\Sigma_{i:w_i=w} a_i^t$ implies that in the input sequence, if the same token appears in the same timestep, the model will add the attention distribution of the same token in a different position.

During training, the loss function used is negative log likelihood (NLL), and $loss_t$ is calculated according to the w_t^* probability generated in each timestep.

$$loss_t = -logp\left(w_t^*\right)$$
$$loss = \frac{1}{T} \Sigma_{t=0}^T loss_t \tag{12}$$

IV. EXPERIMENT

In this section, the data set is introduced, and the quality of the generated sentences is evaluated. Several evaluation methods are used to evaluate the sentences generated by different model architectures. The parameter configurations of the evaluation metrics are adjusted according to the human evaluator's results.

A. DATASET

We used the LCQMC [24] and Phoenix Paraphrasing dataset [25]. We first selected the positive data in the LCQMC [24] and filtered out uncodable sentences and the English data, because the CKIP tagger cannot produce detailed oral English parts, which may affect model performance. Furthermore, as part of our pre-processing steps, we exclude excessively long sentences from the dataset. We also filtered the Phoenix Paraphrasing dataset [24]. The data set details are shown in Tables 4 and 5.

TABLE 4. **LCQMC [24] details.**

type	size	avg sentences length
train	135,781	10.05
dev	4,245	13.26
test	2,998	9.74

TABLE 5. **Phoenix paraphrase dataset [25] details.**

type	size	avg sentences length
train	367,085	8.45
dev	8,133	8.47
test	10,000	8.38

B. AUTOMATIC EVALUATION

Python 3.7.7 and Pytorch 1.10.2 were used to implement the proposed architecture and subsequent evaluations.

- Transformer [1]: Baseline model
- Multi-Encoder Transformer [13]: Integrating Transformer architecture using CKIP tagger to generate part-of-speech features
- Transformer + Pointer Generator Network [17]: Transformer architecture combined with PGN
- Multi-Encoder Transformer + Pointer Generator Network: Our proposed architecture

The hyperparameter configuration of the encoder architecture for part-of-speech in each model architecture is consistent with the configuration of the encoder hyperparameters of the token. The configuration parameters used in the Transformer model are consistent with those described in the original paper [1].

We first evaluated the model using an additional corpus to train the Word2vec model and applied it to the word embedding layer of the model. The experimental results of the additional corpus are shown in Table 6. We used Wikipedia's corpus [28] for training, and after preprocessing the data, we continued to finetune the pretrained Word2vec model. Finally, we evaluated the model by applying the evaluation metrics on the test data set. The PINC score, representing the diversity of sentences, increased slightly, whereas the BERTscore, representing semantic preservation, did not have much loss. Therefore, the follow-up experiments are all analyzed with the model by using the additional corpus.

TABLE 6. Experimental results of using additional corpus to train word embedding.

Metrics	LCQMC[24]		Phoenix Paraphrase Dataset[25]	
	Multi-Encoder Transformer + PGN (original corpus)	Multi-Encoder Transformer + PGN (additional corpus)	Multi-Encoder Transformer + PGN (original corpus)	Multi-Encoder Transformer + PGN (additional corpus)
BLEU-1	**0.816**	0.816	**0.543**	0.539
BLEU-2	0.685	**0.686**	**0.351**	0.345
PINC	0.316	**0.321**	0.684	**0.689**
BERTscore	0.907	**0.908**	**0.834**	0.833

Table 7 shows the experimental results of four models (T represents Transformer, MET represents Multi-encoder Transformer, T + PGN represents Transformer + PGN, and MET + PGN represents Multi-encoder Transformer PGN) by using the LCQMC. We calculated the average score of each model. Based on the experimental results for BLEU-1, BLEU-2 [18], and PINC [22], we hypothesize that the relatively high scores are mostly a result of repeated input sentences, which in turn affects the values of other evaluation metrics. Therefore, we determined the distribution of each model PINC [22] in each score interval of the LCQMC.

TABLE 7. Evaluation of four models using LCQMC.

Metrics	LCQMC[24]			
	T	MET	T+PGN	MET+PGN
BLEU-1	0.815	0.810	**0.816**	**0.816**
BLEU-2	**0.690**	0.681	0.688	0.686
PINC	0.313	**0.327**	0.307	0.321
BERTscore	**0.910**	0.903	0.907	0.908
ParaScore.Free	**0.900**	0.897	0.897	0.898

Table 8 shows the experimental results obtained using the Phoenix Paraphrasing dataset. The scores for the Multi-encoder Transformer, Transformer + PGN, and our proposed architecture were slightly lower than those for BLEU-1 and BLEU-2. However, for the PINC scores, the performance of other model architectures was higher than that of the Transformer architecture. This indicates that by adding additional features and using P_{gen}, the diversity of sentences can be improved while preserving a certain degree of semantic similarity.

The hyperparameter configuration used for the ParaScore evaluation metric was the same as that used in the original paper. In the next section, the hyperparameter configuration for ParaScore is based on the results of human evaluation.

The distribution of each interval is shown in Fig. 5. The BERT and BLEU scores were 1 due to repeated sentences, in which the paraphrased and input texts were the same.

TABLE 8. Evaluation of four models using phoenix paraphrasing dataset.

Metrics	Phoenix Paraphrase Dataset[25]			
	T	MET	T+PGN	MET+PGN
BLEU-1	**0.546**	0.543	0.541	0.539
BLEU-2	**0.363**	0.350	0.347	0.345
PINC	0.673	0.686	0.687	**0.689**
BERTscore	**0.840**	0.834	0.833	0.833
ParaScore.Free	**0.847**	0.842	0.842	0.841

This in turn affected the overall average BERT and BLEU scores.

C. HUMAN EVALUATION

This section describes the evaluation of the models with the LCQMC and Phoenix Paraphrasing dataset. The evaluation involves selecting 30 sentences from each model for a total of 60 sentences for human evaluation.

The human evaluation results are shown in Table 9. The architecture proposed in this study provided higher performance than the other three models when using the Phoenix Paraphrasing dataset, and it was more in line with human cognition than simply using the Transformer architecture for the LCQMC. We hypothesize that this may be due to the distribution of the corpus. The corpus contains sentences with high similarity between the source and target, and the model tends to repeat input sentences. Features other than text are regarded as noise.

We adjusted the parameters for ParaScore according to the human evaluation. The original study used $\omega = 0.05$ and $\gamma = 0.35$. Under the configuration of ω and γ, the automatic evaluation metric has the highest degree of correlation with the results of human evaluation.

$$NED = \frac{dist(X,C)}{max(|X|,|C|)}$$

$$DS\,(X,C) = \begin{cases} \gamma & d > \gamma \\ d \cdot \dfrac{\gamma+1}{\gamma} - 1 & 0 \leq d \leq \gamma \end{cases}$$

$$ParaScore.Free = Sim(X,C) + \omega \cdot DS(X,C) \tag{13}$$

where X represents the input sentence; C represents the generated sentence; d is the value of NED, Sim(X,C) is the

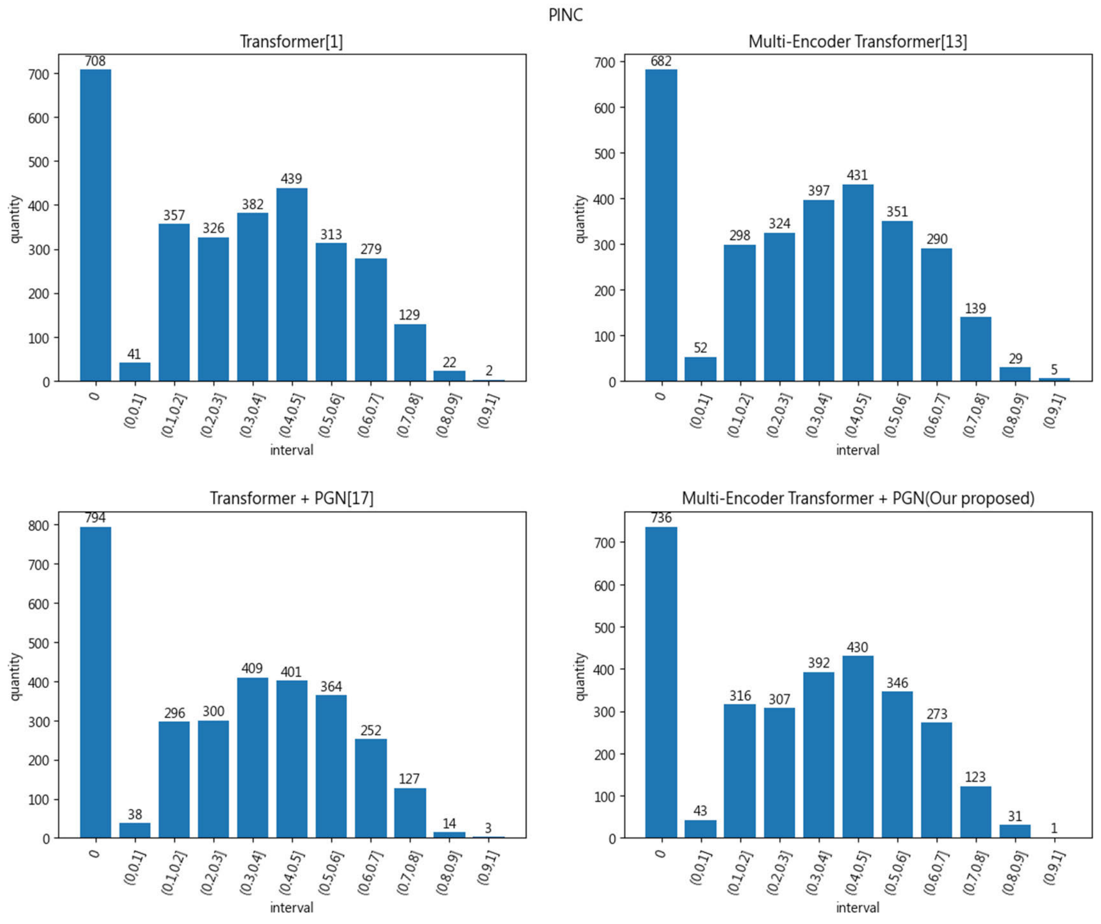

FIGURE 5. Statistical table of PINC intervals for each model on LCQMC.

TABLE 9. Results on human evaluation.

Model	LCQMC		Phoenix Paraphrase Dataset	
	score	Std	score	Std
Transformer[1]	**3.845**	**1.137**	3.671	1.042
Multi-Encoder Transformer[13]	3.553	1.230	3.279	1.117
Transformer + PGN[17]	3.505	1.291	3.604	1.007
Multi-Encoder Transformer + PGN	3.703	1.359	**3.779**	**0.942**

BERTscore used to indicate the semantic similarity of the two sentences; DS(X,C), which can be passed through NED, is used to represent the diversity of sentences; and ω and γ are the hyperparameters.

According to ParaScore, d can be replaced by other ways to express the difference between the two sentences.

Therefore, in addition to the correlation analysis for NED, we also replaced d with the PINC score. The PINC score was close to that of human evaluation. The best parameter configuration was $\omega = 0.05$ and $\gamma = 0.45$ for the LCQMC and $\omega = 0.05$ and $\gamma = 0.65$ for the Phoenix Paraphrasing dataset. The results of the correlation experiment are shown in Table 10. The P values of the experimental results are all less than 0.01.

TABLE 10. Correlation between ParaScore and human evaluation.

d	LCQMC		Phoenix Paraphrase Dataset	
	Pr.	Spr.	Pr.	Spr.
NED	0.478	0.417	0.354	0.319
PINC	**0.534**	**0.439**	**0.390**	**0.348**

Table 11 shows the values generated after all test data were reconfigured according to the new hyperparameter configuration. Several phenomena are revealed, one of which is because of the relationship between the size of the data set

TABLE 11. Results for parascore with adjusted hyperparameters.

model	LCQMC	Phoenix Paraphrase Dataset
Transformer[1]	**0.903**	**0.866**
Multi-Encoder Transformer[13]	0.900	0.860
Transformer + PGN[17]	0.901	0.860
Multi-Encoder Transformer + PGN	0.902	0.860

and the size of the human evaluation data. The volumes of the LCQMC and the data of human evaluation are close to the volume of the Phoenix Paraphrasing dataset. Therefore, after adjusting the parameters of ParaScore, the performance of each model is closer to the results of human evaluation. However, a gap still exists between the ParaScore for the Phoenix Paraphrasing dataset and the results of human evaluation. Another reason is that according to the experimental results, the correlation between the human evaluation results and ParaScore for the Phoenix Paraphrasing dataset is relatively weak compared with the correlation results for the LCQMC. A weak correlation indicates that the model is unable to generate the same trend as human evaluation.

In Table 12 and 13, examples of sentences generated by the architecture proposed in this study using each of the data sets are listed.

TABLE 12. Sentences generated using LCQMC.

Source	Generated Sentences
怎樣看到好友微信的朋友圈	微信怎麼看到朋友圈
怎麼領取支付寶紅包	支付寶紅包怎麼領取
適合在婚禮上唱的歌	適合婚禮上唱的流行歌
到醫院看腎虛，掛什麼科	腎虛檢查去醫院掛什麼科

TABLE 13. Sentences generated using phoenix paraphrasing dataset.

Source	Generated Sentences
日本哪個眼霜好	日本眼霜排行榜
割完雙眼皮多久可以化妝	割完雙眼皮幾天能化妝
孕婦吃燕窩有什麼危害	孕婦吃燕窩的壞處
胃下垂的主要表現	胃下垂的症狀有哪些

V. CONCLUSION AND FUTURE WORKS

Most of the research on paraphrasing has been conducted using English corpora. Relatively few studies have used Chinese corpora. In addition to using a Chinese corpus, this study used the new part-of-speech features derived from the CKIP tagger. By using the part-of-speech feature, the model was able to better learn the part-of-speech and the grammatical rules of the text in Chinese. In addition, we used the PGN to improve paraphrasing by assigning the attention distribution to the source sentence. The experimental results indicate that according to human evaluation using the Phoenix Paraphrasing dataset, the model exhibited higher performance than the other models. The correlation between human evaluation results and ParaScore indicated that the PINC evaluation metric was better than NED. After adjusting the evaluation metric parameters, ParaScore values for each model did not differ substantially, whereas PINC values, which represent sentence diversity, increased more than those for other model architectures. When using the LCQMC, according to the ParaScore and human evaluation results, the architecture proposed in this study was not the best, but according to the PINC results, the architecture proposed in this study was better than the Transformer architecture.

According to feedback from several human evaluators, most of the text generated by each model was similar. In addition, the conversion from Simplified Chinese to Traditional Chinese affects the meaning of the text. A complete and large-scale Traditional Chinese corpus should be developed for future research.

This study used the NLL loss function. Future studies can integrate the focus of various indicators into the loss function to enable the model to learn according to the degree of semantic preservation and sentence diversity during training and to improve the quality of sentence generation.

ACKNOWLEDGMENT

The authors thank all participants for their contributions and project funding.

REFERENCES

[1] A. Vaswani, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[2] P. Nakov, "Improved statistical machine translation using monolingual paraphrases," in *Proc. 18th Eur. Conf. Artif. Intell.*, Patras, Greece, vol. 178, Jun. 2008, p. 338.

[3] Y. Li, H. Li, and J. Liu, "Towards robust neural machine reading comprehension via question paraphrases," in *Proc. Int. Conf. Asian Lang. Process. (IALP)*, Nov. 2019, pp. 290–295, doi: 10.1109/IALP48816.2019.9037673.

[4] S. Gao, Y. Zhang, Z. Ou, and Z. Yu, "Paraphrase augmented task-oriented dialog generation," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 639–649, doi: 10.18653/v1/2020.acl-main.60.

[5] W. Lan, S. Qiu, H. He, and W. Xu, "A continuously growing dataset of sentential paraphrases," 2017, *arXiv:1708.00391*.

[6] S. Iyer, N. Dandekar, and K. Csernai. *First Quora Dataset Release: Question Pairs*. Quora. Accessed: Oct. 1, 2022. [Online]. Available: https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

[7] B. Dolan, C. Quirk, and C. Brockett, "Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources," in *Proc. 20th Int. Conf. Comput. Linguistics*, Geneva, Switzerland, Aug. 2004, pp. 350–356. Accessed: Oct. 1, 2022. [Online]. Available: https://aclanthology.org/C04-1051

[8] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," 2017, *arXiv:1704.04368*.

[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," 2016, *arXiv:1610.03098*.

[10] A. Gupta, A. Agarwal, P. Singh, and P. Rai, "A deep generative framework for paraphrase generation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018, pp. 1–8.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, Jun. 2016, pp. 770–778.

[12] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[13] S. Wang, R. Gupta, N. Chang, and J. Baldridge, "A task in a suit and a tie: Paraphrase generation with semantic augmentation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 1, 2019, pp. 7176–7183.

[14] M. Ringgaard, R. Gupta, and F. C. N. Pereira, "SLING: A framework for frame semantic parsing," 2017, *arXiv:1710.07032*.

[15] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.

[16] L. Ravuru, H. Choi, K. M. Siddarth, H. Lee, and I. Hwang, "Paraphrase generation based on VAE and pointer-generator networks," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2019, pp. 860–866.

[17] X. Guo, Q. Zeng, M. Jiang, Y. Xiao, B. Long, and L. Wu, "Automatic controllable product copywriting for e-commerce," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 12423–12431.

[18] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proc. 40th Annu. meeting Assoc. Comput. Linguistics*, 2002, pp. 311–318.

[19] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proc. ACL Workshop Intrinsic Extrinsic Eval. Measures Mach. Transl. Summarization*, 2005, pp. 65–72.

[20] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating text generation with BERT," 2019, *arXiv:1904.09675*.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[22] D. Chen and W. B. Dolan, "Collecting highly parallel data for paraphrase evaluation," in *Proc. 49th Annu. Meeting Assoc. Comput. linguistics, Human Lang. Technol.*, 2011, pp. 190–200.

[23] L. Shen, L. Liu, H. Jiang, and S. Shi, "On the evaluation metrics for paraphrase generation," 2022, *arXiv:2202.08479*.

[24] X. Liu, "LCQMC: A large-scale Chinese question matching corpus," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 1952–1962.

[25] *Baidu Research Open-Access Dataset—Introduction*. Accessed: Sep. 21, 2022. [Online]. Available: https://ai.baidu.com/broad/introduction?dataset=paraphrasing

[26] C. Kuo. *Open Chinese Convert Open Chinese Conversion*. Accessed: Sep. 21, 2022. [Online]. Available: https://github.com/BYVoid/OpenCC

[27] CKIP Lab. *CkipTagger*. Accessed: Sep. 21, 2022. [Online]. Available: https://github.com/ckiplab/ckiptagger

[28] *Wikipedia Database Downloads, Wikipedia, the Free Encyclopedia*. Accessed: Nov. 16, 2022. [Online]. Available: https://zh.wikipedia.org/w/index.php?title=Wikipedia:%E6%95%B0%E6%8D%AE%E5%BA%93%E4%B8%8B%E8%BD%BD&oldid=69463056

YU-CHIA TSAI was born in Taoyuan, Taiwan, in 1999. He received the M.S. degree in computer science and information engineering from Feng Chia University, Taichung, Taiwan, in 2023. His research interests include natural language processing, edge computing, and computer vision.

FENG-CHENG LIN received the Ph.D. degree in computer science and information engineering from National Chi Nan University, Taiwan, in 2008. Currently, he is an Associate Professor with the Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan. His research interests include cloud computing, deep learning, business intelligence, and multimedia applications.