Learning When and What to Quote: A Quotation Recommender System with Mutual Promotion of Recommendation and Generation

Lingzhi Wang[1,2], **Xingshan Zeng**[3], **Kam-Fai Wong**[1,2]
[1]The Chinese University of Hong Kong, Hong Kong, China
[2]MoE Key Laboratory of High Confidence Software Technologies, China
[1,2]{lzwang,kfwong}@se.cuhk.edu.hk [3]zxshamson@gmail.com

Abstract

This work extends the current quotation recommendation task to a more realistic quotation recommender system that learns to predict when to quote and what to quote jointly. The system consists of three modules (tasks), a prediction module to predict whether to quote given conversation contexts, a recommendation module to recommend suitable quotations and a generation module generating quotations or sentences in ordinary language to continue the conversation. We benchmark several competitive models for the two newly introduced tasks (i.e., when-to-quote and what-to-continue). For quotation recommendation, compared with previous work that is either generation-based or ranking-based recommendation, we propose a novel framework with mutual promotion of generation module and ranking-based recommendation module[1]. Experiments show that our framework achieves significantly better performance than baselines on two datasets. Further experiments and analyses validate the effectiveness of the proposed mechanisms and get a better understanding of the quotation recommendation task.

1 Introduction

The rise of social media platforms exposes people to more opportunities to share viewpoints (Lee and Ma, 2012; Bakshy et al., 2015). People get to know each other by what they post or say, and the art of chatting on the internet has become more and more important. Using quotations would be a good way to make one's expression more clear, beautiful, and persuasive (Booten and Hearst, 2016). However, for many individuals, thinking up a suitable quotation that fits the ongoing context is a daunting task. The issue becomes more pressing when quoting in online conversations where quick responses are usually needed on mobile devices.

[1]The code is available at https://github.com/Lingzhi-WANG/GenRecMutualPromo.

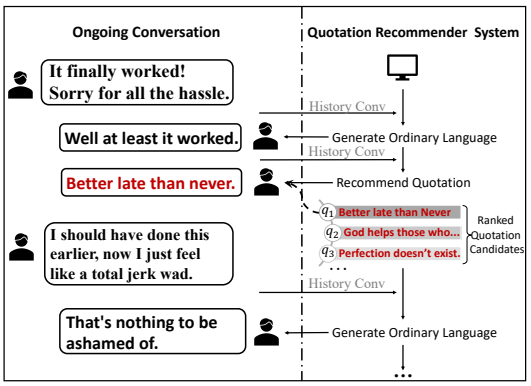


Figure 1: An example of interactions between user and quotation recommender system. Words in ordinary language are in **black** and quotations are in **red**.

To that end, extensive efforts have been made to quotation recommendation — aiming to recommend suitable quotations given conversation context. Nevertheless, previous work (Tan et al., 2015; Liu et al., 2019; Wang et al., 2020) mostly focuses on what to quote, i.e., ranking the quotation candidates, but ignores the problem of whether or when to quote, which should be an indispensable part of a real-world applicable system as people may not realize that quoting in good time can enhance their persuasiveness because of their insufficient knowledge of quotations. We extend the previous quotation recommendation task to a quotation recommender system that consists of three modules, a when-to-quote module to predict whether to recommend quotations, a recommendation module to recommend quotations given conversation context and a generation module to generate sentences in ordinary language or quotations to continue the conversation.

To better illustrate the system, Fig. 1 shows an interaction example between user and system. The system can generate sentences in ordinary language (e.g., "Well at least it worked") to continue the conversation or recommend proper quotations in more persuasive language, where the when-to-quote module decides to recommend or generate

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 3094–3105
December 7-11, 2022

ordinary sentences. As we are the first to formulate the quotation recommender system with two new modules (i.e., the when-to-quote prediction module and generation module), we provide benchmarks for the two newly added modules and propose a unified framework with novel quotation recommendation module.

For quotation recommendation, the previous work either employs generation-based (Liu et al., 2019; Wang et al., 2020) or ranking-based (Tan et al., 2015; Wang et al., 2021) models, where the quotations are regarded as word sequences and labels, respectively. Rather than applying either of them, we choose to fully explore the advantages of both types of models and propose a novel framework with a mutual promotion of recommendation and generation. Our basic recommendation module is a ranking-based framework, where a pretrained language model is adopted to obtain the context representation. We use the top quotations recommended by the basic recommendation module as pseudo references to enhance the training of generation module. The motivation of using pseudo references comes from the observation that multiple quotations might be acceptable for one context. Taking Fig. 1 as an example, quotations q_1 to q_3 all can continue the context well. And the multiple references are beneficial for model to learn diverse patterns of generation, fitted in different scenarios (Zheng et al., 2018).

Besides, the pseudo references-enhanced generation module is further adopted to facilitate the recommendation module by re-ranking the recommended quotation list. We expect that the semantic coherence between context and quotations can be emphasized by the cross-attention in the generation decoder. Specifically, we get the posterior probabilities of quotations by feeding them to decoder and re-rank the top quotations recommended by the basic recommendation module accordingly, which is denoted as generative ranking. Rather than employing searching algorithms like greedy search (Wilt et al., 2010) or beam search (Cohen and Beck, 2019) used in conventional generation due to unlimited search space, we choose to use generative ranking by calculating the language probability (Yang et al., 2018) since the search space is limited when generating quotations (the number of quotation candidates is usually fixed). Compared to the previous work that relies on beam-search for quotation recommendation (Liu et al., 2019; Wang et al., 2020), the generative ranking method does not require any post-processing to match the generated sentences to quotations and shows better performance.

For experiments, our recommendation module outperforms the previous work significantly on two datasets (Weibo and Reddit), and performs well on when-to-quote prediction and what-to-continue generation. Extensive experiments show that our mutual promotion mechanism is effective. More interesting experiments such as generative ranking vs. beam search are given to yield a better understanding of the quotation recommendation task.

The main contributions of this work are:

- We propose a novel quotation recommendation framework with a mutual promotion of recommendation and generation, which outperforms the previous work significantly.
- We extend the previous quotation recommendation task to a complete recommender system with two new targets, when to quote and what to continue, and provide corresponding benchmarks.
- We conduct extensive and interesting experiments to show the effectiveness of our framework.

2 Related Work

Quotation recommendation is in line with content-based recommendation (Liu et al., 2019) or cloze-style reading comprehension (Zheng et al., 2019), which learns to put suitable text fragments (e.g., words, phrases, sentences) in the given contexts. The current research on quotation recommendation can be divided into two categories as follows.

Quotation Recommendation for Formal Writing. Studies in this category explore various setting, such as quoting famous sayings in books (Tan et al., 2015, 2016), and using idioms in news articles (Liu et al., 2019; Zheng et al., 2019). Tan et al. (2015) propose a learning to rank framework, where particular features (e.g., frequency, vote, web-popularity, and quote-rank) are employed. Then Tan et al. (2016) first apply neural models, which avoids the time-consuming calculation and extraction of the features. As for recommending idioms for news articles, Liu et al. (2019) propose a neural machine translation framework, in which they suppose that the idioms are in pseudo target language and context is the source language to be translated. Zheng et al. (2019) formulate the idiom recommendation task as a cloze test task and contributes a new dataset and provides a benchmark to evaluate

the ability of understanding idioms.

Quotation Recommendation for Online Conversation. This line of work faces more challenges in modeling the complex interaction and noisy context of online conversations. Lee et al. (2016) combine a recurrent neural network and a convolutional neural network to learn semantic representation and structure representation. Wang et al. (2020) contribute two datasets for quotation recommendation on online conversations. They propose a seq-to-seq framework to do the quotation recommendation and employ a neural topic model to get the latent representation of the history and thus assist the recommendation. Wang et al. (2021) propose a ranking model where a transformation from queries to quotations is employed to enhance the quotation recommendation performance.

3 Our Model

3.1 Problem Formulation

Input to the System. The input mainly contains the observed conversation context C, and the quotation list Q. The conversation C is formalized as a sequence of turns (e.g., posts or comments) $\{t_1, t_2, ..., t_{n_c}\}$ where n_c represents the length of the conversation (i.e., number of turns). t_i ($1 \leq i \leq n_c$) denotes the i-th turn of the conversation and we use $\boldsymbol{w}_i$ to indicate the word tokens contained in it. The quotation list Q covers all quotations that have appeared in the training corpus. It is represented with $\{q_1, q_2, ..., q_{n_q}\}$, where n_q is the number of quotations and q_k is the k-th quotation in list Q.

Output of the System. Conditioned on the observed conversation C, the when-to-quote prediction module outputs a label $y^p \in \{\langle\text{quo}\rangle, \langle\text{gen}\rangle\}$, where $\langle\text{quo}\rangle$ indicates needing to quote and $\langle\text{gen}\rangle$ means no need to quote. Then the recommendation module outputs a label $y^q \in \{1, 2, ..., n_q\}$ to indicate which quotation to recommend. Finally, the generation module will generate an output sequence $y^g = \{y_1^g, \ldots, y_n^g\}$ based on the conversation context C and the prediction label y^p.

3.2 Quotation Recommendation Framework

BART-based Generation Module. The generation module of our designed model follows a general Transformer (Vaswani et al., 2017) sequence-to-sequence framework. To relieve the data burden and enhance context modeling, we choose to finetune a pre-trained BART (Lewis et al., 2020) model

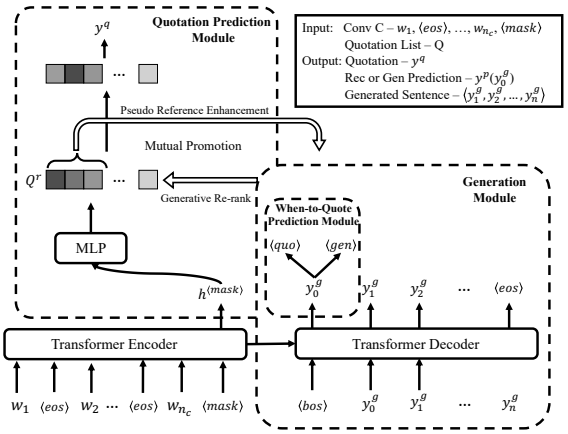


Figure 2: Our framework for quotation recommender system, which consists of three modules, a *recommendation module* to recommend quotations fitting the context, a *prediction module* to decide when to quote, and a *generation module* to generate sentences to continue the conversation with ordinary language or quotations.

for our generation module. BART was trained with several denoising objectives on large-scale unlabeled data, and has been shown to be effective in many generation tasks, like summarization, machine translation and persona-based response generation. During finetuning, we concatenate the utterances t_i ($1 \leq i \leq n_c$) in context C with appended $\langle\text{eos}\rangle$ tokens in their chronological order as the input, and maximize the probability of the ground-truth target sequence T (T can be a quotation or a general sentence in a real-world conversation). The whole process is summarized as:

$$\boldsymbol{H}^c = \text{Transformer\_Encoder}(\boldsymbol{w}^c) \quad (1)$$

$$y_k^g = \text{Transformer\_Decoder}(y_{<k}^g, \boldsymbol{H}^c) \quad (2)$$

$$\mathcal{L}_{gen}^{basic} = -\sum_{k=1}^{n} \log(p(y_k^g | y_{<k}^g, \boldsymbol{H}^c)) \quad (3)$$

where $\boldsymbol{w}^c = [\boldsymbol{w}_1; \langle\text{eos}\rangle; \boldsymbol{w}_2; ..; \boldsymbol{w}_{n_c}; \langle\text{mask}\rangle]$, and $y_{<k}^g$ represents the target tokens before y_k^g. It has been proved effective (Schick and Schütze, 2021) to simulate the operations conducted in the pre-training stage during finetuning, thus we add an $\langle\text{mask}\rangle$ token at the end of context, indicating that we need to produce corresponding context in the position of $\langle\text{mask}\rangle$ token.

When-to-Quote Prediction Module. To make the framework simplified, we embedded the prediction procedure into the process of generation. To that end, we treat the prediction labels as two special instruction tokens, $\langle\text{quo}\rangle$ and $\langle\text{gen}\rangle$, to indicate whether to recommend quotations or generate normal sentences when continuing the conversation. Specifically, to enable our generation

module generating prediction labels, we extend its original vocabulary $\mathcal{V}$ with $\langle$quo$\rangle$ and $\langle$gen$\rangle$ to be $\mathcal{V}' = \mathcal{V} \cup \{\langle\text{quo}\rangle, \langle\text{gen}\rangle\}$. Our generation module is supposed to first generate an instruction token and then generate a quotation or a normal sentence accordingly. Therefore, the objective in Eq. 3 should be reformulated as:

$$\mathcal{L}_{gen}^{pred} = -\sum_{k=0}^{n} \log(p(y_k^g | y_{<k}^g, \boldsymbol{H}^c)) \tag{4}$$

where $y_0^g \in \{\langle\text{quo}\rangle, \langle\text{gen}\rangle\}$ denotes the instruction token, and $y^g = \{y_1^g, y_2^g, ..., y_n^g\}$ is the target sequence to be generated.

What-to-Quote Recommendation Module. We adopt the representation for the $\langle$mask$\rangle$ token in each conversation to represent the target sentence. The motivation comes from the idea in masked language models (Devlin et al., 2019), where they use $\langle$mask$\rangle$ tokens to replace the original content and then use the representation in those positions to predict the original tokens. We denote the representation of the $\langle$mask$\rangle$ token as $\boldsymbol{h}^{\langle\text{mask}\rangle}$. Then it is fed into a two-layer MLP for recommendation:

$$\boldsymbol{r}^q = \boldsymbol{W}_2 \times \alpha(\boldsymbol{W}_1 \boldsymbol{h}^{\langle\text{mask}\rangle} + \boldsymbol{b}_1) + \boldsymbol{b}_2 \tag{5}$$

where $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are learnable parameters, and α is a non-linear activation function and we use *tanh* in our work. The output representation $\boldsymbol{r}^q$ will be an n_q-dimension vector and the candidate quotation list Q will be ranked according to the probability computed based $\boldsymbol{r}^q$:

$$p_r(\hat{q} = k) = \text{softmax}(\boldsymbol{r}^q)_k \tag{6}$$

We then denote the ranked quotation list as $Q^r = \{q_1^r, q_2^r, ..., q_{n_q}^r\}$ and the top m quotations as $Q_{1:m}^r$.

3.3 Mutual Promotion of Recommendation and Generation

Pseudo References Enhanced Generation. Our promotion for generation is motivated by the fact that there might be multiple quotations acceptable for one giving context. We take the instance shown in Figure 1 as an example. The ground truth quotation provided is q_1, while q_2 and q_3 are also suitable for that context from the perspective of semantic coherence. Therefore, we propose to use the top m_q predictions ($Q_{1:m_q}^r$) to serve as pseudo references. We assume that the pseudo references can help the training of the generation module and thus enhance the generation performance.

Specifically, for each input context C with quotation target, we first extract the content of top m_q quotations (i.e. $Q_{1:m_q}^r$) in recommendation module. After prepending with the instruction token $\langle$quo$\rangle$, they also serve as the references for the same context C to increase the training corpus. To distinguish them from ground truth targets, we add confidence weights to the losses computed by the pseudo instances and the weights are set as the recommendation probability computed with Eq. 6. Therefore, the total objective for the generation module is summarized as:

$$\mathcal{L}_{gen} = -\sum_{(C,T)\in\mathcal{D}} p(C,T) \sum_{k=0}^{n} \log(p(y_k^g | y_{<k}^g, \boldsymbol{H}^c)) \tag{7}$$

where $\mathcal{D}$ represents the total training corpus including pseudo instances and C and T are context and target, respectively. $p(C,T) = 1$ if it is not a pseudo instance; otherwise let T be the content of quotation q_k, then $p(C,T) = p_r(\hat{q} = k)$ computed with Eq. 6.

Generative Re-Ranking Enhanced Recommendation. The promotion for recommendation is based on a designed re-ranking mechanism, where the ranked quotation list Q^r will be re-ranked based on the generative probability calculated by our generation module. The idea comes from our assumption that a well-trained generation module can evaluate the semantic coherence of a given context and target. To that end, we choose to re-rank the top m_g quotations ($Q_{1:m_g}^r$) produced by our recommendation module. We first feed the generation module with the context C (as input) and the corresponding top m_g quotations (as target) and then derive the average log-probability for each quotation to compute generation-based quotation probability, which will be added to the original probability (i.e., $p_r(\hat{q} = k)$) for final re-ranking-based recommendation probability:

$$p_g(\hat{q} = k) = \frac{\exp(\frac{1}{n_k+1} \sum_{i=0}^{n_k} \log p(y_i^{g_k}))}{\sum_{j \in q_{1:m}^r} \exp(\frac{1}{n_j+1} \sum_{i=0}^{n_j} \log p(y_i^{g_j}))} \tag{8}$$

$$p(\hat{q} = k) = \lambda \cdot p_r(\hat{q} = k) + (1 - \lambda) p_g(\hat{q} = k) \tag{9}$$

where $\log p(y_i^{g_k})$ is short for $\log p(y_i^{g_k} | y_{<i}^{g_k}, \boldsymbol{H}^c)$, representing the log-probability of the i-th token in quotation q_k^r. λ is a hyper-parameter to control the effects of two probabilities and $p(\hat{q} = k)$ is the final probability for re-ranking. The quotation with highest probability will be recommended.

Joint Training of Recommendation and Generation Modules. Previous work shows that the

quotation recommendation can be regarded as either a ranking task or a generation task, as the recommended quotation is also used to continue the conversation. This kind of dual identity indicates that we can finetune our model with both recommendation and generation objectives to make the two modules promote each other. The total learning objective therefore is defined as:

$$\mathcal{L} = \mathcal{L}_{rec} + \gamma \cdot \mathcal{L}_{gen} \tag{10}$$

where γ is a hyper-parameter to control the tradeoff between the two losses and $\mathcal{L}_{rec}$ is the learning objective for recommendation module:

$$\mathcal{L}_{rec} = -\sum_{(C,q^c)\in\mathcal{D}} \log p_r(\hat{q} = q^c|C) \tag{11}$$

where q^c is the ground truth quotation for conversation C. Alg. 1 in Apendix A depicts the detailed process of our mutual promotion.

4 Experimental Setup

Datasets and Statistics. For the task of what to quote, we employ two datasets, Weibo and Reddit, released by Wang et al. (2020). We adopt the same data split as Wang et al. (2020) for a fair comparison. For when to quote, we augment the two datasets by splitting the history contexts in original datasets[2]. Specifically, we built two different rules for Weibo and Reddit to detect possible positions in context that might need prediction (please refer to Appendix B for details). The statistics of the datasets can be found in Appendix C (Table 8).

Evaluation Metrics. For recommendation, we adopt popular evaluation metrics, including MAP (Mean Average Precision), P@1 (Precison@1), P@3 (Precison@3), and nDCG@5 (normalized Discounted Cumulative Gain@5) for evaluation by following Wang et al. (2020). For generation, we adopt BLEU[3], Rouge-1 and Rouge-L to evaluate the generated sentences by following Wang et al. (2020). To evaluate the task of when to quote, we adopt Accuracy, F1 and Recall scores.

Parameter Setting. For Reddit, we use English BART-Base model released by Lewis et al. (2020) to initialize our model; while for Weibo, we use Chinese BART-Base model released by Shao et al. (2021). Both are with 768 hidden dimension, 12 attention heads and 6 layers of encoder and decoder. The middle dimension of our MLP recommendation layer is also 768. We use Adam optimizer (Kingma and Ba, 2015) with a learning rate of 1e-4 for optimization and the batch size is 64. Dropout strategy (Srivastava et al., 2014) with dropout rate of 0.1 and L_2 regularization with 0.0003 effect value, as well as early stopping (Caruana et al., 2001), are used to alleviate overfitting. We set γ in Eq. (10) as 0.1 to make the model focus more on the recommendation task. Top 5 (i.e., $m_q = 5$) quotations are used for pseudo references and Top 30 (i.e., $m_g = 30$) quotations are used for re-ranking. λ in Eq. (9) is set to 0.6.

Comparisons. For our main experiments on quotation recommendation, we compare our model with two simple baselines (RANDOM and FREQUENCY) and several previous proposed methods, including two generation-based models (NCIR (Liu et al., 2019) and CTIQ (Wang et al., 2020)), and three ranking-based models (LTR (Tan et al., 2015), BERT+MLP (Devlin et al., 2019) and TRANSQQ (Wang et al., 2021)). For generation, we also compare the two generation-based models, as well as TAKG (Wang et al., 2019), another generation model for a similar task. As for when-to-quote prediction, we compare some basic methods as no previous work explores this task. To save space, we introduce all of them in detail in Appendix D.

5 Experimental Results

In this section, we first compare the recommendation results of our model with the previous quotation recommendation models in §5.1. Then we report the results of when-to-quote prediction and generation in §5.2 and §5.3, respectively. Finally, further experiments are given in §5.4 and §5.5, together with a case study in §5.6, to provide insights on how our method works.

5.1 Main Recommendation Results

We compare our model with the state-of-the-art baselines and conduct an ablation study to show the effectiveness of the designed mechanisms.

Comparison with Baselines. We first report the main comparison results of quotation recommendation in Table 1 and draw the following observations.

• *Our model outperforms all the baselines significantly on both Weibo and Reddit.* It can be seen

[2]To ensure a same domain, we construct samples with original data rather than collecting new corpus.

[3]We utilize Sacrebleu package (https://github.com/mjpost/sacrebleu) to calculate the scores.

Models	Weibo				Reddit			
	MAP	P@1	P@3	NG@5	MAP	P@1	P@3	NG@5
RANDOM	0.6	0.1	0.1	0.3	0.5	0.1	0.1	0.1
FREQUENCY	7.1	2.6	7.1	6.6	4.7	1.0	3.0	2.9
Generative								
NCIR	26.5	22.6	27.8	26.7	12.2	7.3	12.3	11.4
CTIQ	30.3	27.2	33.2	31.6	21.9	17.5	25.8	23.8
Ranking								
LTR	9.3	3.6	8.5	8.1	7.1	1.7	6.4	6.2
BERT+MLP	31.4	27.9	34.0	32.3	26.4	18.0	30.2	28.5
TRANSQQ	34.9	30.3	36.1	34.9	31.8	23.3	35.0	32.1
Our Model	**39.9**	**35.8**	**41.2**	**40.1**	**35.7**	**28.4**	**38.8**	**36.2**

Table 1: Main comparison results (in %) on quotation recommendation. Our model outperforms baselines significantly on all metrics ($p<0.01$, paired t-test).

Models	Weibo				Reddit			
	MAP	P@1	P@3	NG@5	MAP	P@1	P@3	NG@5
Our Full Model	**39.9**	**35.8**	**41.2**	**40.1**	**35.7**	**28.4**	**38.8**	**36.2**
- PLM	37.1	32.3	38.2	37.4	33.3	26.2	35.9	33.6
- GenPromo	39.0	34.9	40.5	39.2	34.2	27.0	36.4	34.5
- RecPromo	39.1	35.0	40.3	39.0	35.2	28.1	37.6	35.5
- MutualPromo	38.1	34.3	39.1	38.1	34.1	27.4	35.9	34.2

Table 2: Ablation results (in %) of recommendation.

that our model shows great improvements on all evaluation metrics compared to baselines. The improvements may mainly come from two aspects, the utilization of pretrained language model and the proposed mutual promotion mechanism. We'll discuss this more in ablation study.

• *Neural ranking models perform better than generative models.* We can see that the neural ranking models (i.e., BERT+MLP and TRANSQQ) get better performance than the generative models (i.e., NCIR and CTIQ), with larger performance gap on Reddit. This is because the generative models need to predict quotations word by word, and such an autoregressive process might introduce error accumulation and lead to worse performance. Although a post-processing procedure can match generated sentences (which are not always the same as those in the quotation list) to the quotations, it can only partially reduce the errors. This can be verified by the larger performance gap on Reddit, since the average length of quotations on Reddit is longer than that on Weibo (see Table 8).

Ablation Study. To better show the effectiveness of our model, we conduct an ablation study to compare our full model with four variants (please refer to Appendix D for the details). We report the results in Table 2. Some observations can be drawn:

• *PLM contributes a lot but our model without PLM can still have better performance than SOTA.* We can see that PLM contributes more than 2 MAP scores on both Weibo and Reddit, which shows the effectiveness of pre-training on large-scale unlabeled data. Nevertheless, our model without PLM initialization can have a better performance than previous SOTA (e.g., our model w/o PLM gets 37.1 MAP while previous SOTA TRANSQQ gets 34.9 MAP on Weibo). This validates that our model's performance improvement is not only because of the adoption of the pretrained language model.

Models	Weibo			Reddit		
	F1	Rec	Acc	F1	Rec	Acc
RANDOM	52.2	49.6	49.8	50.5	49.8	49.6
ALLYES	71.3	**100**	55.4	68.1	**100**	51.6
BILSTM	76.8	80.4	70.1	77.9	79.3	78.0
Our Model	**82.4**	89.3	**81.0**	**87.3**	89.8	**87.1**
- PLM	80.1	84.5	78.8	80.4	83.9	79.8

Table 3: Results (in %) on when-to-quote prediction.

• *Mutual promotion of recommendation and generation modules is effective and removing either of them will cause performance degradation.* From Table 2, we can see that our model without mutual promotion shows 1.8 and 1.6 MAP degradation on Weibo and Reddit, respectively, which exhibits the effectiveness of the proposed mutual promotion mechanism. We also notice that removing either promotion (i.e., GenPromo and RecPromo) would cause performance degradation, and the degradation is less than removing both of them. This further validates that the two promotions are not contradictory and can improve each other.

5.2 When-to-Quote Prediction Results

Before recommending, our system is supposed to predict whether to quote according to the context. We report the prediction results in Table 3. Due to the lack of previous work on this task, we list two simple baselines (RANDOM and ALLYES) to reveal how challenging the task is and a basic model BILSTM (Zhou et al., 2016) to show the performance of conventional neural networks. Also compared are our full model and the variant without PLM.

The results in Table 3 show that the simple baselines (i.e., RANDOM and ALLYES) perform much worse than neural-based models, indicating the importance of capturing semantic information from context. We can also find that our model gets better performance on almost all metrics (especially when using PLM) and achieves more than 80 F1 scores,

	Models	Weibo			Reddit		
		BLEU	RG-1	RG-L	BLEU	RG-1	RG-L
Quotation	TAKG	24.0	26.8	26.7	6.7	14.4	15.7
	NCIR	22.6	25.3	25.2	4.1	10.9	9.9
	CTIQ	27.2	29.5	29.5	9.5	20.3	18.8
	Our Model	**34.8**	**38.6**	**38.7**	**19.6**	**29.8**	**29.7**
	- GenPromo	32.7	35.6	35.7	13.9	23.0	23.2
	- PLM	26.3	29.6	29.9	12.4	18.0	17.9
All	BILSTM	16.3	23.6	23.9	3.43	8.75	8.61
	Our Model	**22.6**	**30.8**	**31.3**	**7.45**	**19.9**	**20.4**
	- GenPromo	21.5	29.0	29.3	5.16	16.3	16.5
	- PLM	18.1	24.8	25.4	4.45	12.1	12.0

Table 4: Main comparison results (in %) on generation.

which validates that it is feasible and also reliable to apply such a model for when-to-quote prediction. Another observation is that the prediction results on Reddit are generally better than Weibo. This is because Chinese quotations on Weibo are phrases, which are more flexible when used in conversations, thus more difficult for model to predict.

5.3 What-to-Continue Generation Results

We report the generation results from two aspects, results of quotation generation (upper part of Table 4), and results of sentences regardless of quotation or ordinary language (lower part of Table 4).

From the quotation generation results in Table 4, we find that the our full model outperforms the previous generation-based recommendation models (i.e., NCIR and CTIQ), which shows our model's stronger ability for quotation generation. On the other hand, the overall results drop a lot regardless of the generated content, especially on Reddit. This is reasonable since ordinary utterances might contain a variety of different content that is difficult to generate, while quotations are relatively fixed sentence expressions that appeared several times in the training corpus. Nonetheless, our model still shows better performance than conventional generation models like BILSTM. The ablation results on both comparison setting ("Quotation" and "All") also show that the promotion of generation module and the pretrained language model are effective.

5.4 Effectiveness of Mutual Promotion

Recommended Based VS. Random Pseudo References. To examine whether pseudo references provided by recommendation module are effective, we compare them with random references (i.e., randomly select m_q quotations as pseudo references). From Table 5, we can find that recommended-based references show positive effects on the performance while the random references do not show an obvious promotion. We can also notice that the models trained with random references are not sensitive to the values of m_q (the number of pseudo references). Conversely, the number of references affects recommended-based promotion a lot, and 5 pseudo references show the best promotion.

References	Weibo				Reddit			
	2	5	10	20	2	5	10	20
Random	38.1	38.2	38.1	38.3	34.2	34.1	34.0	34.1
$Q^r_{1:m_q}$	38.3	39.0	38.8	38.5	34.9	35.5	35.2	34.3

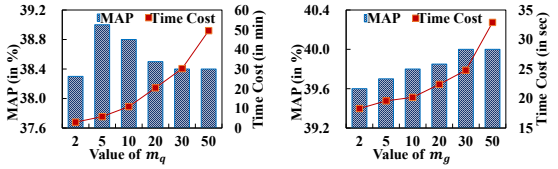
Table 5: MAP scores comparisons between random references and using $Q^r_{1:m_q}$ ranked by recommendation module as references with varying reference numbers (i.e.,$m_q = 2, 5, 10, 20$).

Models	Weibo			Reddit			Complexity
	P@1	P@3	NG@5	P@1	P@3	NG@5	
GenerativeRank	35.2	40.9	39.4	26.9	36.5	34.3	$\mathcal{O}(\|Q\|)$
BeamSearch	32.8	38.1	36.4	23.6	31.6	29.6	$\mathcal{O}(K^2T)$
BeamSearch†	33.1	38.4	36.8	24.1	32.7	30.7	$\mathcal{O}(K^2T + \|Q\|T^2)$

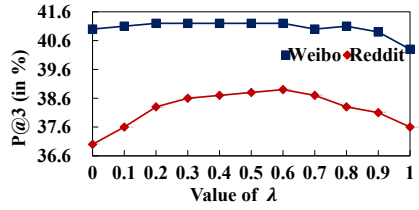
Table 6: Comparison results (in %) of generative ranking (i.e., our model with $m_g = |Q|$ and $\lambda = 0$) and beam search generation for quotation recommendation. † refers to adopting post-processing to match generated sentences to quotations with minimum edit distance. K denotes beam size and T is sequence length.

Generative Ranking VS. Beam Search. Previous methods (Liu et al., 2019; Wang et al., 2020) mainly use beam search to produce quotation recommendation. We argue that it is a suboptimal solution since the quotation numbers are fixed and the actual search space is limited. We propose generative ranking, which ranks the quotations by their posterior probability calculated by generation module. It can be viewed as a special case of our generative re-ranking enhanced recommendation, where we set $m_g = |Q|$ and $\lambda = 0$. We report the recommendation results of our generation module with different recommendation methods (i.e., generative ranking and beam search) in Table 6.

It can be found that our generative ranking shows better performance than beam search, even after post-processing. Nevertheless, it requires more computation cost than the naive beam search (generally $K \ll |Q|$ and $T \ll |Q|$), as it needs to pass all quotations through the model. This observation serves as one of the reasons why we propose generative re-ranking enhanced recommendation, i.e., only using top quotations provided by our recommendation module for generation module to re-rank, which saves computation cost.



(a) MAP and Time over m_q (b) MAP and Time over m_g



(c) P@3 over λ for Weibo and Reddit

Figure 3: Fig. 3(a) and 3(b): MAP scores and time cost over m_q and m_g. Fig. 3(c): P@3 scores over λ.

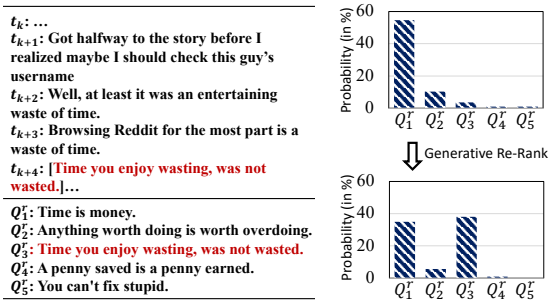
5.5 Analyses on Mutual Promotion

We explore how hyper-parameters m_q (number of quotations for pseudo references), m_g (number of quotations for generative re-ranking), λ (trade-off value for two probabilities in re-ranking) affect the recommendation performance in Fig. 3.

Effects of Pseudo References Number. Fig. 3(a) shows the MAP results and training time per epoch (in minutes) when using different numbers (i.e., m_q) of top quotations as pseudo references. Apparently, increasing m_q will result in longer training time. The best MAP score is achieved when m_q equals to 5, indicating that too many unrelated pseudo references is harmful for the final results.

Effects of Quotation Number for Re-ranking. We examine using how many quotations (m_g) for the re-ranking would achieve the best performance and the corresponding time cost (in seconds) in Fig. 3(b). As can be seen, the MAP scores keep increasing when more quotations are included to do the re-ranking until m_g equals 30 and keep unchanged when changed from 30 to 50. This indicates that the very top quotations provided by our recommendation module have already contained the most possible targets, and there is no need to re-rank the longer quotation list to save time cost.

Effects of Trade-off Value in Re-ranking. We also examine how the value of trade-off parameter λ influences the recommendation results in Fig. 3(c). As can be seen, the results on two datasets exhibit different trends. On Reddit, the best performance is achieved around the middle of the curve ($\lambda = 0.6$); while the performance on



(a) Context and Top 5 quotations (b) Probability Change

Figure 4: An example from Reddit with top 5 quotations recommended by basic recommendation module (Fig. 4(a)) and the change of probability distribution of the quotations (Fig. 4(b)).

Weibo remains unchanged when $\lambda \in [0, 0.6]$ and degrades if λ increases from 0.6. We attribute this to the fact that the quotations on Weibo are easier for generation module to predict (validated by much better BLEU scores achieved on Weibo from Table 4). Thus the generative re-ranking on Weibo is more reliable than on Reddit.

5.6 Case Study

We use one cherry-pick example to show the distribution differences of quotation probabilities before and after the generative re-ranking, respectively, in Fig. 4. We can see that among the top 5 quotations predicted by recommendation module, the ground truth quotation Q_3^r ranks the third place, and it gets the highest probability after the generative re-ranking. This might be because the generation module can capture the semantic coherence between the context and the quotation (e.g. "entertaining", "waste" in the context and "enjoy", "wasting" in the ground truth quotation) with cross attention, while the recommendation module treats quotations as discrete labels and ignore that kind of information.

6 Conclusion

This work explores a realistic recommender system that recommends quotations and provides when-to-quote prediction and what-to-continue generation. We provide the benchmarks for the two newly added tasks and propose a mutual promotion mechanism for quotation recommendation. Experiments show that our method can promote both generation and recommendation and contribute to the best quotation recommendation performance.

Limitations

A key limitation of this work is that our model cannot handle the cold-start problem because we adopt a fixed-size MLP layer in the basic recommendation module. Though we think the quotation candidate list is statically unchanged compared to other conventional recommendations (e.g., news, products recommendation), the cold start problem for quotation is still a good point to be explored in the future. Another limitation lies in the evaluation metrics. We only adopt traditional correctness evaluation metrics, which is not sufficient if the quotation recommendation is applied in personalized applications.

Acknowledgement

This research work is supported by Innovation & Technology Commission HKSAR, under ITF Project No. ITT/008/22LP.

References

Eytan Bakshy, Solomon Messing, and Lada A Adamic. 2015. Exposure to ideologically diverse news and opinion on facebook. *Science*, 348(6239):1130–1132.

Kyle Booten and Marti A. Hearst. 2016. Patterns of wisdom: Discourse-level style in multi-sentence quotations. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1139–1144.

Rich Caruana, Steve Lawrence, and C Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems*, pages 402–408.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *International Conference on Machine Learning*, pages 1290–1299. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Chei Sian Lee and Long Ma. 2012. News sharing in social media: The effect of gratifications and prior experience. *Computers in human behavior*, 28(2):331–339.

Hanbit Lee, Yeonchan Ahn, Haejun Lee, Seungdo Ha, and Sang-goo Lee. 2016. Quote recommendation in dialogue using deep neural network. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 957–960, New York, NY, USA. ACM.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yuanchao Liu, Bo Pang, and Bingquan Liu. 2019. Neural-based Chinese idiom recommendation for enhancing elegance in essay writing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Timo Schick and Hinrich Schütze. 2021. Few-shot text generation with natural language instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 390–402.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2015. Learning to recommend quotes for writing. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2016. A neural network approach to quote recommendation in writings. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Lingzhi Wang, Jing Li, Xingshan Zeng, Haisong Zhang, and Kam-Fai Wong. 2020. Continuity of topic, interaction, and query: Learning to quote in online conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6640–6650, Online. Association for Computational Linguistics.

Lingzhi Wang, Xingshan Zeng, and Kam-Fai Wong. 2021. Quotation recommendation and interpretation based on transformation from queries to quotations. In *Proceedings of the 59th ACL*, pages 754–758.

Yue Wang, Jing Li, Hou Pong Chan, Irwin King, Michael R. Lyu, and Shuming Shi. 2019. Topic-aware neural keyphrase generation for social media language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Christopher Makoto Wilt, Jordan Tyler Thayer, and Wheeler Ruml. 2010. A comparison of greedy search algorithms. In *third annual symposium on combinatorial search*.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. Breaking the beam search curse: A study of (re-) scoring methods and stopping criteria for neural machine translation. *arXiv preprint arXiv:1808.09582*.

Chujie Zheng, Minlie Huang, and Aixin Sun. 2019. Chid: A large-scale chinese idiom dataset for cloze test. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 778–787.

Renjie Zheng, Mingbo Ma, and Liang Huang. 2018. Multi-reference training with pseudo-references for neural translation and text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3188–3197.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 207–212.

Algorithm 1 Mutual Promotion Mechanism

Input: $\mathcal{D}_{train} = \{(C, T, q^c)\}$, $\mathcal{D}_{test} = \{(C)\}$
Output: $\mathcal{D}_{test} = \{(C, T_g \text{ or } \hat{q})\}$
  while not converge **do** ▷ Training the model
    for C, T, q^c in $\mathcal{D}_{train}$ **do**
      Compute generation loss with Eq. 4
      if T is a quote **then**
        Compute recommendation loss with Eq. 11
        Extract top m_q quotations $Q^r_{1:m_q}$ with Eq. 6
        for q in $Q^r_{1:m_q}$ **do** ▷ Promoting generation
          Construct pseudo reference T_q for C
          Compute generation loss with Eq. 7
        end for
      end if
      Optimize model with Eq. 10 by adding up the losses
    end for
  end while
  for C in $\mathcal{D}_{test}$ **do** ▷ Testing the model
    Predict y^g_0 ▷ $y^g_0 \in \{\langle\text{quo}\rangle, \langle\text{gen}\rangle\}$
    if $y^g_0 == \langle\text{gen}\rangle$ **then**
      Generate ordinary sentence T_g
    else ▷ Promoting recommendation
      Get top m_g quotations $Q^r_{1:m_g}$ with Eq. 6
      Re-rank $Q^r_{1:m_g}$ with Eq. 8 and 9
      Recommend $\hat{q}$ with highest probability
    end if
  end for

Appendix

A Mutual Promotion Algorithm

We use Alg. 1 to give a more clear look on how to apply our mutual promotion of generation and recommendation in our framework.

B Details on Constructing Corpus for When-to-Quote Prediction

As Chinese and English quotations are different from each other, we use different rules for Weibo and Reddit to construct corpus for the prediction.

B.1 Constructing Corpus for Weibo Data

The Chinese quotations in Weibo dataset are all Chengyu[4], which mostly consist of four characters and can be regarded as phrases. In our preliminary observation, Chengyu can be a noun, a verb and even an adjective and can be applied in any position of a sentence. Therefore, we considered any positions between two words or phrases (detected by a Chinese tokenizer) to be possible positions for when-to-quote prediction. We then constructed the context-generation pairs by splitting the original conversation at those positions. To alleviate the difficulty of generation and make it closer to quotation generation, we also made the generation limited to at most four words or phrases and remove those samples with too long content to be generated.

Dataset	Prediction Results			Not Sure
	F1 Score	Recall	Accuracy	Rate
Weibo	63.1	60.4	68.3	16.3
Reddit	62.2	63.6	60.7	14.0

Table 7: Human Results (in %) on when-to-quote prediction. "Not Sure Rate" indicates the proportion of cases the predictors are not sure about the answers.

B.2 Constructing Corpus for Reddit Data

The English quotations in Reddit dataset are full sentences obtained from Wikiquote[5]. In our preliminary observation, most of the quotations appeared in Reddit dataset are used after a complete sentence, to explain or summarize the previous statements (only 7.6% of them are used after words like "saying", "said", etc. to explicitly indicate that the following sentences might be quotations, and these can be regarded as a small amount of easy samples to be predicted). Therefore, we considered any positions between two sentences to be possible positions for when-to-quote prediction. We then constructed the context-generation pairs by splitting the original conversation at those positions. To alleviate the difficulty of generation, we made the generation include only one complete sentence and remove the rest content.

B.3 Human Evaluation on the Quality of the Constructed Samples

To examine whether the constructed corpus is of high quality, we conducted a human evaluation to check whether humans can predict well on the constructed corpus. We sampled both 50 samples from the original context-quotation samples and the newly constructed context-generation samples, respectively, to build a human evaluation test set. We then invited three crowd-workers to predict whether quotations can be used to continue the context. Each predictor gave a yes-or-no answer or mark as "not sure" (for those they thought are indistinguishable) for each sample. We then evaluate their results only on those are not marked as "not sure". The average results are displayed in Table 7.

As can be seen, humans scored higher than 60% in all metrics for both datasets, indicating that predicting with the corpus is possible from a human

[4] https://en.wikipedia.org/wiki/Chengyu

[5] https://en.wikiquote.org/wiki/Main_Page

	Weibo	Reddit
# of quotations	1,053	1,111
Average length of quotations	4.0	10.1
Original dataset		
# of conversations	19,081	44,541
Average turn # per conversation	2.51	4.25
Average length of turn	21.6	71.8
Added generation samples		
# of samples	18,585	44,085
Average turn # per sample	1.24	3.07
Average length of turn	32.3	51.1

Table 8: Statistics of Weibo and Reddit datasets.

perspective (a random prediction would score about 50%). On the other hand, about 15% (i.e., the not-sure rate) of the samples are difficult for human to distinguish, which is reasonable and thus worth trying from the perspective of machines.

C Statistics of the Datasets

We display the statistics of the two used datasets in Table 8, including those of our newly constructed samples. We can observe that quotations on Reddit have a longer average length, and the turn length is also much longer than Weibo, which might make it a more difficult dataset. Our newly added samples (served as negative samples for when-to-quote prediction) are similar in number to the original conversations. The average turn number becomes smaller, since we constructed them by splitting the original conversation context.

D Details on Baselines and Variants

D.1 Baselines for Comparisons

Baselines for recommendation:

(1) RANDOM: It ranks the quotations in quotation list randomly.

(2) FREQUENCY: It ranks quotations with their frequency, which means that quotations with higher appearance rate in training set will get a higher rank in the recommendation.

(3) NCIR (Liu et al., 2019): This work formulates quotation recommendation as a context-to-quote machine translation problem by using the encoder–decoder framework with attention mechanism for generation.

(4) CTIQ (Wang et al., 2020): The method employs an encoder-decoder framework enhanced by Neural Topic Model to continue the context with a quotation via language generation.

(5) LTR (Learning to Rank) (Tan et al., 2015): We first extracts the features (e.g., frequency, cosine similarity between quotes and contexts using TF-IDF, LDA, Word2Vec, etc.) mentioned in Tan et al. (2015) and then use the learning to rank tool RankLib[6] to do the recommendation.

(6) BERT+MLP (Devlin et al., 2019): The conversation contexts are directly fed to the pre-trained BERT model, followed by an MLP to predict quotation labels, which ignores the semantic information contained in the quotations.

(7) TRANSQQ (Wang et al., 2021): It introduces a transformation matrix that maps the query representations to quotation and recommends based on a linear projection towards the combination of quotation and conversation representations.

Additional baseline for generation:

(8) TAKG (Wang et al., 2019): A Seq2Seq framework incorporating latent topics for decoding, originally proposed for keyphrase generation.

Baselines for when-to-quote prediction:

(1) RANDOM: It randomly gives a yes-or-no answer for each sample.

(2) ALLYES: It always predicts needing to quote, regardless of the content.

(3) BILSTM: It uses simple BiLSTM to encode the context content, followed by an MLP to predict whether to quote.

D.2 Variants on Ablation Study

We have four variants on ablation study:

(1) "- PLM": We remove the BART initialization and train the model from scratch.

(2) "- GenPromo": We remove the promotion for generation module, i.e., do not use the pseudo references predicted by recommendation module to help the training.

(3) "- RecPromo": We remove the promotion for recommendation module, i.e., do not re-rank the top m_g recommendation results with generation module.

(4) "- MutualPromo": We remove the promotions for both generation and recommendation.

[6] https://github.com/danyaljj/rankLibl