Quotation Recommendation and Interpretation Based on Transformation from Queries to Quotations

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
[3]Huawei Noah's Ark Lab, China
[1,2]{lzwang,kfwong}@se.cuhk.edu.hk [3]zeng.xingshan@huawei.com

Abstract

To help individuals express themselves better, quotation recommendation is receiving growing attention. Nevertheless, most prior efforts focus on modeling quotations and queries separately and ignore the relationship between the quotations and the queries. In this work, we introduce a transformation matrix that directly maps the query representations to quotation representations. To better learn the mapping relationship, we employ a mapping loss that minimizes the distance of two semantic spaces (one for quotation and another for mapped-query). Furthermore, we explore using the words in history queries to interpret the figurative language of quotations, where quotation-aware attention is applied on top of history queries to highlight the indicator words. Experiments on two datasets in English and Chinese show that our model outperforms previous state-of-the-art models.

1 Introduction

Quotations are essential for successful persuasion and explanation in interpersonal communication. However, it is a daunting task for many individuals to write down a suitable quotation in a short time. This results in a pressing need to develop a quotation recommendation tool to meet such a demand.

To that end, extensive efforts have been made to **quotation recommendation**, which aims to recommend an ongoing conversation with a quotation whose sense continues with the existing context (Wang et al., 2020). As quotations are concise phrases or sentences to spread wisdom, which are always in figurative language and difficult to understand, they are assumed written in a different pseudo-language (Liu et al., 2019a). Intuitively, we

The code is available at https://github.com/Lingzhi-WANG/Quotation-Recommendation

[t_1]: Save your money. Scuf is the biggest ripoff in gaming.
[t_2]: What would you suggest instead?
[t_3]: Just use a normal controller.
[t_4]: Ooooooh, I get it now...you're just dumb.
[t_5]: The dumb ones are the people spending over $100 for a controller. [*A fool and his money are soon parted.*]
[h_1]: Anyone that spends that much money just to get different writing on a box..... [*A fool ... parted.*]
[h_2]: And that's probably why you'll never have a billion dollars. [*A fool ... parted.*]
[h_3]: Seriously. Why do people not do market research before buying something!?! [*A fool ... parted.*]

Figure 1: A Reddit conversation snippet (upper part) with three history queries (lower part). Quotations to be recommended are in square brackets. Indicative words are on wavy-underline.

can infer the meanings of quotations by their neighborhood contexts, especially by the *query* turn (the last turn of conversation that needs recommendation).

To illustrate our motivation, Figure 1 shows a Reddit conversation with some history queries associated with quotation Q, "*A fool and his money are soon parted*". From the queries (t_5 and h_1 to h_3), we can infer the meaning of quotation Q is "A foolish person spends money carelessly and won't have a lot of money." based on the contexts. From h_3, we can also know the implication behind the words, which is "Do a marketing research before buying". Humans can establish such a relationship between quotations and queries and then decide what to quote in their writings, so can machines (neural network). Therefore, we introduce a transformation matrix, in which machines can learn the direct mapping from queries to quotations. The matrix is worked on the outputs of two encoders, conversation encoder and quotation encoder, encoding conversation context and quotations respectively.

Furthermore, we can use the words in the queries to interpret quotations. h_1 to h_3 in Figure 1 are

754

denoted as *history queries*, and the words on wavy-underline are denoted as indicators to quotations. It can be seen that we can interpret quotations by highlighting the words in the queries. Therefore, we compute quotation-aware attention over all the history queries (after the same transformation as we mentioned before) and then display indicators we learned, which also reflects the effectiveness of the transformation.

In summary, we introduce a transformation between the query semantic space and quotation semantic space. To minimize the distance of their semantic space after transformation mapping, an auxiliary mapping loss is employed. Besides, we propose a way to interpret quotations with indicative words in the corresponding queries.

The remainder of this paper is organized as follows. The related work is surveyed in Section 2. Section 3 presents the proposed approach. And Section 4 and 5 present the experimental setup and results respectively. Finally, conclusions are drawn in Section 6.

2 Related Work

Quotation Recommendation. In previous works on quotation recommendation, some efforts are made for online conversations (Wang et al., 2020; Lee et al., 2016) and some for normal writing (Liu et al., 2019a; Tan et al., 2015, 2016). Our work focuses on the former. For methodology, the methods they applied can be divided into generation-based framework (Wang et al., 2020; Liu et al., 2019a) and ranking framework (Lee et al., 2016; Tan et al., 2015, 2016). Different from previous works which mainly focus on separate modeling of quotation and query and pay little attention to the relationship between them, our model directly learns the relationship between quotations and query turns based on a mapping mechanism. The relationship mapping is jointly trained with the quotation recommendation task, which improves the performance of our model.

3 Our model

This section describes our quotation recommendation model, whose overall structure is shown in Figure 2. The input of the model mainly contains the observed conversation 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
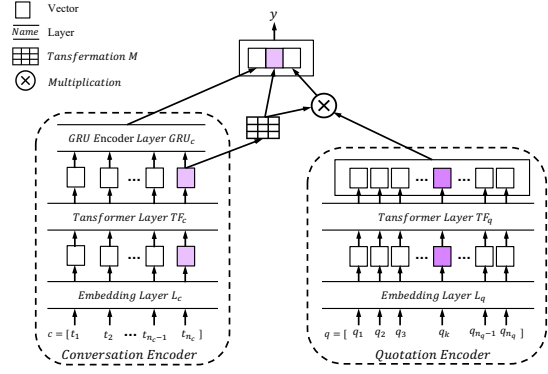

Figure 2: Our model for quotation recommendation.

(number of turns) and t_{n_c} is the query turn. t_i represents the i-th turn of the conversation and contains words $\boldsymbol{w}_i$. The quotation list q is $\{q_1, q_2, ..., q_{n_q}\}$, where n_q is the number of quotations and q_k is the k-th quotation in list q, containing words $\boldsymbol{w}'_k$. Our model will output a label $y \in \{1, 2, ..., n_q\}$, to indicate which quotation to recommend.

3.1 Conversation Modeling

Our model encodes the observed conversation c with a hierarchical structure, which is divided into three parts. The first part is an embedding layer mapping the words $\boldsymbol{w}_i$ in each turn t_i into vectors. We then apply transformer (Vaswani et al., 2017) to learn the representation for each turn. Similar to BERT (Devlin et al., 2018), we only use the encoder of transformer, which is stacked of several self-attention and feed-forward layers. We add a token [CLS] at the beginning of each turn. The hidden representation of [CLS] after transformer encoder is defined as the turn representation $\boldsymbol{r}_i^t$ of turn t_i. The procedures for the first two parts are summarized as follows:

$$\boldsymbol{h}_i^T = \text{FFN}(\text{Self\_Attention}(\text{Embed}([w_0; \boldsymbol{w}_i])))$$
(1)

where w_0 represents the [CLS] token, and $[;]$ indicates concatenation. Therefore $\boldsymbol{r}_i^t = \boldsymbol{h}_{i,0}^T$.

Next, we use a Bi-GRU (Cho et al., 2014) layer to model the whole conversation structure. With the turn representations $\{\boldsymbol{r}_1^t, \boldsymbol{r}_2^t, ..., \boldsymbol{r}_{n_c}^t\}$ ($\boldsymbol{r}_{n_c}^t$ is the representation for the query turn) of conversation c derived from previous procedure, the hidden states are updated as follows:

$$\overrightarrow{\boldsymbol{h}}_i^G = \overrightarrow{\text{GRU}}(\overrightarrow{\boldsymbol{h}}_{i-1}^G, \boldsymbol{r}_i^t), \quad \overleftarrow{\boldsymbol{h}}_i^G = \overleftarrow{\text{GRU}}(\overrightarrow{\boldsymbol{h}}_{i+1}^G, \boldsymbol{r}_i^t)$$
(2)

Finally, we define the conversation representation as the concatenation of the final hidden states from two directions: $\boldsymbol{h}^c = [\overrightarrow{\boldsymbol{h}}_{n_c}^G; \overleftarrow{\boldsymbol{h}}_1^G]$.

3.2 Quotation Modeling

For each quotation q_k in list q, we extract quotation representation $\boldsymbol{r}_k^q$ with similar operation as turn representations (see Eq. 1). As Liu et al. (2019b) points out, the language used in quotations is usually different from our daily conversations, which results in two different semantic spaces. Therefore, we do not share the parameters of the embedding layer and transformer layers for quotations and conversation turns. We concatenate all the quotation representations and get a combined quotation matrix $\boldsymbol{Q}$, which includes n_q rows and each row represents one quotation.

3.3 Recommendation Based on Transformation

To perform a reasonable recommendation, we consider the observed conversation c, the query turn t_{n_c} as well as the quotation list q. Since they are in different semantic spaces (Section 3.2), we first map the query turns into the space of quotations with a transformation matrix $\boldsymbol{M}$. We assume with such transformation, the space gap can be resolved. Thus, we can calculate the distance between queries and quotations. We use $\boldsymbol{z}^c$ to represents the distances between $\boldsymbol{r}_{n_c}$ and the quotations, and it is defined with the following equation:

$$\boldsymbol{z}^c = \boldsymbol{Q} \times (\boldsymbol{M}\boldsymbol{r}_{n_c}) \qquad (3)$$

Finally, the output layer is defined as:

$$\boldsymbol{y} = \boldsymbol{W}[\boldsymbol{z}^c; \boldsymbol{h}^c; \boldsymbol{M}\boldsymbol{r}_{n_c}] + \boldsymbol{b} \qquad (4)$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are learnable parameters. We recommend the quotations with the top n highest probabilities, which are derived with a softmax function:

$$p(\hat{q} = i) = \frac{\exp(y_i)}{\sum_{k=1}^{n_q} \exp(y_k)} \qquad (5)$$

3.4 Training Procedure

We define our training objective as two parts. The first part is called recommendation loss, which is the cross entropy over the whole training corpus $\mathbb{C}$:

$$\mathcal{L}_{rec} = -\sum_{c \in \mathbb{C}} \log p(\hat{q} = q^c | c, q) \qquad (6)$$

where q^c is the ground-truth quotation for conversation c in training corpus. The second part is to help on the learning of transformation matrix $\boldsymbol{M}$, where we minimize the distance between the transformed query turn representation and the corresponding ground-truth quotation:

$$\mathcal{L}_{map} = \sum_{c \in \mathbb{C}} ||\boldsymbol{M}\boldsymbol{r}_{n_c} - \boldsymbol{r}_{q^c}^q||_2^2 \qquad (7)$$

To train our model, the final objective is to minimize $\mathcal{L}$, the combination of the two losses:

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda \cdot \mathcal{L}_{map} \qquad (8)$$

where λ are the coefficient determining the contribution of the latter loss.

4 Experimental Setup

Datasets. We conduct experiments based on datasets from two different platforms, Weibo and Reddit, released by Wang et al. (2020). To make our experimental results comparable to Wang et al. (2020), we utilize their preprocessed data directly.

Parameter Setting. We first initialize the embedding layer with 200-dimensional Glove embedding (Pennington et al., 2014) for Reddit and Chinese words embedding (Song et al., 2018) for Weibo. For transformer layers, we choose the number of layers and heads as (2, 3) for Reddit and (4, 4) for Weibo. For the hidden dimension of transformer layers and BiGRU layers (each direction), we set it to 200. We employ Adam optimizer (Kingma and Ba, 2015) with initial learning rate with 1e-4 and early stop adoption (Caruana et al., 2001) in training. The batch size is set to 32. Dropout strategy (Srivastava et al., 2014) and L_2 regularization are used to alleviate overfitting. And the tradeoff parameter λ is chosen from {1e-4, 1e-3}. All the hyper-parameters above are tuned on the validation set by grid search.

Evaluation and Comparisons. Our model returns a quotation list arranged in descending order of likelihood of recommendation for each conversation. Therefore, we adopt MAP (Mean Average Precision), P@1 (Precison@1), P@3 (Precison@3), and nDCG@5 (normalized Discounted Cumulative Gain@5) for evaluation.

For comparison, we compare with previous works that focus on quotation recommendation. Below shows the details:
1) LTR (Learning to Rank). We first collect features (e.g., frequency, Word2Vec, etc.) mentioned in Tan et al. (2015) and then use the learning to rank tool RankLib [1] to do the recommendation.

---

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

Models	Weibo				Reddit			
	MAP	P@1	P@3	NG@5	MAP	P@1	P@3	NG@5
Baselines								
LTR	9.3	3.6	8.5	8.1	7.1	1.7	6.4	6.2
CNN-LSTM	11.3	7.3	11.0	10.8	5.2	4.1	7.0	6.9
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
BERT	31.4	27.9	34.0	32.3	26.4	18.0	30.2	28.5
OUR MODEL	**34.9**	**30.3**	**36.1**	**34.9**	**31.8**	**23.3**	**35.0**	**32.1**

Table 1: Main comparison results on Weibo and Reddit datasets (in %). NG@5 refers to NDCG@5. The best results in each column are in **bold**. Our model yields significantly better scores than all other comparisons for all metrics ($p < 0.01$, paired t-test).

2) <u>CNN-LSTM</u>. We implement the model proposed in Lee et al. (2016), which adopts CNN to learn the semantic representation of each turn and then uses LSTM to encode the conversation.

3) <u>NCIR</u>. It formulates quotation recommendation as a context-to-quote machine translation problem by using the encoder–decoder framework with attention mechanism (Liu et al., 2019b).

4) <u>CTIQ</u>. The SOTA model (Wang et al., 2020), which employs an encoder-decoder framework enhanced by Neural Topic Model to continue the context with a quotation via language generation.

5) <u>BERT</u>. We encode the conversation by BiLSTM on the BERT representations for the turns, followed by a prediction layer.

5 Experimental Results

5.1 Quotation Recommendation

Table 1 displays the recommendation results comparing our model with the baselines on Weibo and Reddit datasets. Our model achieves the best performance, exceeding the baselines by a large margin, especially on Reddit dataset. The fact that better performance comes from BERT and our model indicates the importance of learning efficient content representations. Our model further considers the mapping between different semantic spaces, resulting in the best performance.

Ablation Study. We conduct an ablation study to examine the contributions of different modules in our model. We replace the transformer layers with Bi-GRU (W/O Transformer) to examine the effects of different turn encoders. We also compare the models by removing transformation matrix M (W/O M) or mapping loss $\mathcal{L}_{map}$ (W/O $\mathcal{L}_{map}$). The results are shown in Table 2. As can be seen, each module in our model plays a role in improving per-

Models	Weibo			Reddit		
	MAP	P@1	NDCG@5	MAP	P@1	NDCG@5
W/O Transformer	29.9	25.9	29.8	25.8	17.4	25.7
W/O M	31.7	27.4	31.8	29.5	21.6	29.5
W/O $\mathcal{L}_{map}$	32.6	28.4	32.4	30.4	22.6	30.6
Full Model	34.9	30.3	34.9	31.8	23.3	32.1

Table 2: Comparison results of different variants of our model on Weibo and Reddit datasets (in %).

Figure 3: Upper part: example queries associated with the quotation "A fool and his money are soon parted.". Lower part: top 8 indicative words with the highest weighted summed self-attention scores. Darker colors represent higher weights.

formance. The largest improvement comes from applying transformers as our encoders. The performance drop due to removing transformation and mapping loss justifies our assumption of different semantic spaces between quotations and queries.

5.2 Quotation Interpretation

We also explore how to interpret the figurative language of quotations with our model. We first extract the queries that are related to one certain quotation as history queries, then compute quotation-aware attention over all history queries. Specifically, for quotation q_k, with its relative history queries $\{h_1, h_2, ..., h_{m_k}\}$ from the corpus (m_k is the history number), we can compute their quotation-aware attention (query-level) with their representations derived from our model :

$$a_{k,i} = \frac{\exp(\boldsymbol{r}_k^q \cdot \boldsymbol{r}_{h_i})}{\sum_{j=1}^{m_k} \exp(\boldsymbol{r}_k^q \cdot \boldsymbol{r}_{h_j})} \qquad (9)$$

On the other hand, we can extract the scores for the words in each history query with their self-attention weights (word-level) in transformer. Finally, the indicative words of one quotation are those with the highest scores after the multiplication of query-level and word-level attention scores.

Figure 3 shows an interpretation example. We display three example queries mentioned in Figure 1, with both their query-level attention (green) and

word-level attention (red). We can find that words like "spends", "money" and "dollars" are assigned higher scores since they are more related to the quotation topics . We also present the most indicative words derived from all history queries (the lower part of Figure 3). We can easily infer the meaning of the quotation with the help of indicative words like "idiots" and "buy".

6 Conclusion

In this paper, we propose a transformation from queries to quotations to enhance a quotation recommendation model for conversations. Experiments on Weibo and Reddit datasets show the effectiveness of our model with transformation. We further explore using indicative words in history queries to interpret quotations, which shows rationality of our method.

Acknowledgements

References

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.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

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.

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

Yuanchao Liu, Bo Pang, and Bingquan Liu. 2019b. Neural-based Chinese idiom recommendation for enhancing elegance in essay writing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5522–5526, Florence, Italy. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 175–180.

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, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, 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.