Personalized Response Generation with Tensor Factorization

Zhenghui Wang[†], **Lingxiao Luo**[‡], **Diyi Yang**[†]
[†]Georgia Institute of Technology, [‡]Tsinghua University
luolx17@mails.tsinghua.edu.cn
{zhwang,dyang888}@gatech.edu

Abstract

Personalized response generation is essential for more human-like conversations. However, how to model user personalization information with no explicit user persona descriptions or demographics still remains under-investigated. To tackle the data sparsity problem and the huge number of users, we utilize tensor factorization to model users' personalization information with their posting histories. Specifically, we introduce the personalized response embedding for all question-user pairs and form them into a three-mode tensor, decomposed by Tucker decomposition. The personalized response embedding is fed to either the decoder of an LSTM-based Seq2Seq model or a transformer language model to help generate more personalized responses. To evaluate how personalized the generated responses are, we further propose a novel ranking-based metric called Per-Hits@k which measures how likely are the generated responses come from the corresponding users. Results on a large-scale English conversation dataset show that our proposed tensor factorization based models generate more personalized and higher quality responses compared to baselines. We have publicly released our code at https://github.com/GT-SALT/personalized_response_generation.

1 Introduction

Building human-like conversational systems has received much attention in artificial intelligence communities, and personalized response generation is one essential step towards this goal, as more personalized responses are often associated with increased user engagement (Shum et al., 2018; Huang et al., 2020). To this end, we focus on the task of personalized response generation in this work, and argue that incorporating personalization into text generation can benefit many downstream applications such as social chit-chat chatbots (Zhang et al., 2018) and auto-complete responses like Smart Replies (Kannan et al., 2016).

Prior text generation work on modeling personalization mainly relied on explicitly given persona or demographic information. For instance, (Zhang et al., 2018; Wolf et al., 2019; Xu et al., 2020) utilized a set of persona sentences to profile users, and other line of research leveraged demographics to model user personalization (Zheng et al., 2019, 2020). Despite its effectiveness, such approaches are limited when it comes to real world scenarios. First, explicit persona or demographic information is often not available. Second, collecting such personalization information is usually costly and time-consuming, which also suffers from either artificially designed persona descriptions from third-party annotators or subjective and unreliable self-reports from users themselves (Stone et al., 1999). Although such explicit personalization information is often unavailable, content that users produce is generally ubiquitous and can indicate their preferences, personal information, styles, and knowledge in a relatively implicit but objective manner. Our work thus utilizes these posts and comments users made to learn latent representations of their personalization information.

Different generation models have been designed to learn user personalization information and further impose such representation on text generation. For instance, Li et al. 2016 proposed the Speaker model based on Seq2Seq framework by introducing trainable *speaker embedding* for each user and feeding it to decoder at each step of decoding. However, there are always a large number of distinct users and users often participate in only a few conversations; as a result, the speaker embedding may be under-fitted given the limited data points associated with a user. Another line of research uses generative memory network (Zhang et al., 2018),

Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021), pages 47–57

which first retrieves some most relevant responses to a user's input as the memory and then encodes them into an embedding. The difference between the embedding from memory network and speaker embedding is that the former encodes both information of question and user, while the latter represents only users. Nevertheless, the set of observable question-user pairs and their responses is still a small subset of the whole user and question sets, leading to the sparsity issue.

Matrix Factorization (MF) has been widely used to infer latent relationships between users and items in recommender systems, especially for data sparsity issues (kumar Bokde et al., 2015). Motivated by this, we propose to model latent interactions between questions and users by looking at who participated in which conversations, and infer user personalization information from data automatically, for personalized response generation tasks. Differently, as the score or rating used in recommender system usually denotes users' preferences towards items, such scalar is not enough to represent the semantic meaning of a response. Thus, we introduce a response vector to indicate the response content that a user will make for a given conversation, i.e., *personalized response embedding*, resulting in a tensor form representation for all question-user pairs. Decomposing this tensor (tensor factorization, TF) will lead to the factorized representations for each user, question, and dimension of the response embedding. We propose to augment response generation models with such TF-induced modules, which are model-agnostic and can be applied to many different generation models. Specifically, we introduce a TF module based framework on top of LSTM-based Seq2Seq model and transformer language model for personalized response generation, and further train them together in an end-to-end fashion. Evaluating response generation usually considers content relatedness and language quality to ensure that generated text is grammatically correct and fluent, using BLEU and Perplexity. However, evaluating personalization in personalized response generation is relatively challenging as there lacks effective metrics.

To this end, we propose a novel evaluation metric **Per-Hits@**k to model personalization , which for the response of a user first calculates its perplexity values via language models of all users, and then ranks the perplexity via this user's language model to examine whether it is ranked as top-k, based on a pre-trained GPT-2 language model (Radford et al., 2019) for each user. Our contributions are:

- propose a tensor factorization based framework to model personalization for response generation task;
- introduce a metric Per-Hits@k, to evaluate the personalization of the generated responses;
- experimental results on a large-scale personalized Reddit dataset show that our TF-based framework outperforms previous methods significantly in terms of both content generation quality and personalization.

2 Related Work

Personalized Response Generation Personalization has received much attention in the natural language processing community, such as personalized image captioning (Chunseong Park et al., 2017), personalized machine translation (Rabinovich et al., 2017), personalized response generation (Li et al., 2016), personalized intent classification and personalized slot tagging (Liu et al., 2016). Prior studies formulate the task of response generation as generating an output given an input text, mainly based on either the sequence-to-sequence (Seq2Seq) models (Vinyals and Le, 2015) or the pretrained models like GPT-2 (Radford et al., 2019) and BART (Lewis et al., 2019). When it comes to personalized response generation, Speaker model (Li et al., 2016) extended traditional response generation models by assigning each user with a trainable speaker ID embedding. Another line of research focuses on leveraging persona descriptions or demographic attributes (Zheng et al., 2020; Qian et al.; Wolf et al., 2019; Luo et al., 2019), building on recent personalized dialogue datasets such as PERSONA-CHAT (Zhang et al., 2018) and PersonalDialog (Zheng et al., 2019). For instance, Xu et al. (2020) utilized the predefined user persona description together with their semantically correlated content for generating personalized responses in dialogue systems.

Different learning paradigms have also been introduced for personalized response generation such as reinforcement learning (Mo et al., 2016; Yang et al., 2018; Xu et al., 2020) and transfer learning to benefit from a source domain with sufficient training data (Yang et al., 2017). However, most aforementioned approaches require explicit

persona or demographic information which is often unavailable in real world scenarios. To fill this gap, we propose to learn latent representation of personalized user information from users' posts and model personalization jointly together with traditional generation methods for personalized response generation.

Evaluation Metrics for Personalized Response Generation Current automatic evaluation metrics for response generation can be broadly categorized into three classes. (1) Content relatedness measures how related a generated response is with its corresponding ground-truth, with representative metrics such as BLEU (Papineni et al., 2002), NIST (Doddington, 2002), and METEOR (Lavie and Agarwal, 2007). Speaker sensitive responses evaluation model (SSREM) (Bak and Oh, 2020) enhances the relatedness score with a context-response classifier. (2) Language quality mainly refers to the fluency and diversity, where the former is measured via perplexity (Chen et al., 1998) and the latter is assessed via distinct diversity (Li et al., 2015; Yang et al., 2020) that indicates how diverse the generated responses are. (3) Style adherence aims to evaluate the adherence of the generated responses' language style to the user's own language style; example metrics include the average negative log-likelihood (NLL) of one poet's generated lyrics on it's poet specific language model (Vechtomova et al., 2018), stylistic alignment (Syed et al., 2020) that looks at the language style alignment at the surface, lexical and syntactic level, and Hits@1/N (Dinan et al., 2019) that measures how accurate the generated response can be classified to its corresponding user by a classifier. Our proposed Per-Hits@k metric thus belongs to the style adherence class, a more fine-grained metric compared to the average NLL metric (Vechtomova et al., 2018).

3 Preliminaries

3.1 Tucker Decomposition

To learn latent association between users, questions and responses for personalized response generation, we choose Tucker decomposition, one widely used tensor factorization algorithm. Tucker decomposition (Tucker, 1966) decomposes a given 3-mode tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ into a core tensor $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ and three factor matrices $\mathbf{A} \in \mathbb{R}^{I \times R_1}$, $\mathbf{B} \in \mathbb{R}^{J \times R_2}$, $\mathbf{C} \in \mathbb{R}^{K \times R_3}$:

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$$

Here, $\times_i$ denotes the mode-i product of a tensor by a matrix ($i \in \{1, 2, 3\}$). Any element $\mathcal{X}_{(i,j,k)}$ in $\mathcal{X}$ can be approximated by:

$$\sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \mathcal{G}_{(r_1,r_2,r_3)} \mathbf{A}_{(i,r_1)} \mathbf{B}_{(j,r_2)} \mathbf{C}_{(k,r_3)}$$

3.2 LSTM-based Seq2Seq Model

LSTM-based Seq2Seq model consists of an encoder LSTM, a decoder LSTM, and attention mechanism (Yao et al., 2015). Suppose the source text is $S = (x_1, x_2, \ldots, x_m)$ and the target text is $T = (x_{m+1}, x_{m+2}, \ldots, x_N)$, the encoder LSTM first encodes S into hidden vector $\boldsymbol{h}_m^e$ and cell vector $\boldsymbol{c}_m^e$, then the decoder LSTM has its initial hidden vector $\boldsymbol{h}_0^d$ and cell vector $\boldsymbol{c}_0^d$ as:

$$\boldsymbol{h}_0^d = \boldsymbol{h}_m^e$$
$$\boldsymbol{c}_0^d = \boldsymbol{c}_m^e$$

The hidden vector of decoder at time step t is:

$$\boldsymbol{h}_t^d = g(\boldsymbol{h}_{t-1}^d, \boldsymbol{c}_{t-1}^d, \boldsymbol{y}_t^*),$$

where g is the LSTM cell operation and $\boldsymbol{y}_t^*$ is the embedding of the input token at time step t.

Standard Seq2Seq models are not personalized, because there is no mechanism to incorporate user-specific information into their input. Speaker Model (Li et al., 2016) alleviates this by explicitly concatenating a trainable *speaker embedding* $\boldsymbol{v}_j$ to $\boldsymbol{y}_t^*$ for user j. Therefore, the hidden vector of decoder of Speaker model at time step t is:

$$\boldsymbol{h}_t^d = g(\boldsymbol{h}_{t-1}^d, \boldsymbol{c}_{t-1}^d, [\boldsymbol{y}_t^*; \boldsymbol{v}_j]),$$

3.3 Transformer Language Model

DialoGPT (Zhang et al., 2020) is a pre-trained conversational response generation model. Based on the architecture of GPT-2 (Radford et al., 2019), DialoGPT is trained on 147M Reddit discussions. For a question-user pair (i, j) with source input S and target response T, DialogGPT generates responses by modeling the conditional probability:

$$P(T \mid S) = \prod_{n=m+1}^{N} P(x_n \mid x_1, x_2, \ldots, x_{n-1})$$

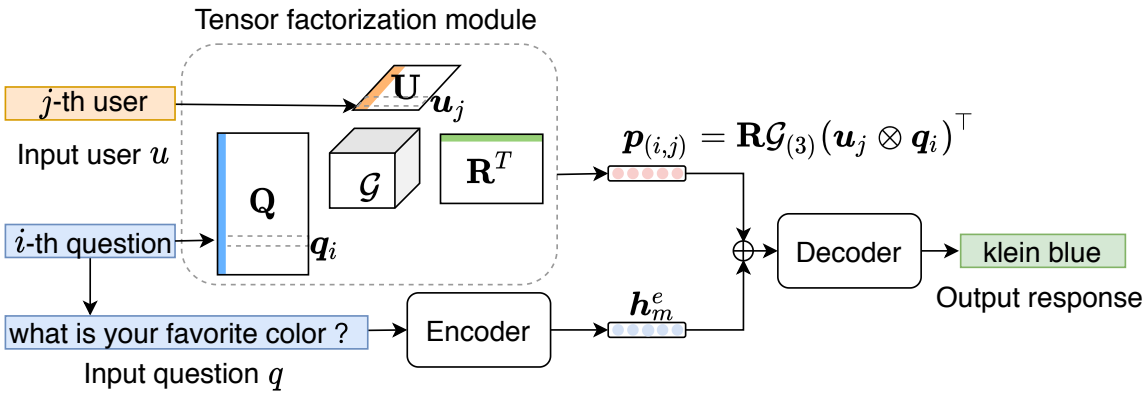

Figure 1: LSTM-based Seq2Seq model with our proposed tensor factorization module. The cell vector $\boldsymbol{c}_m^e$ from the encoder and the attention mechanism are omitted for brevity.

4 Method

We formulate the task of personalized response generation as follows: given a set of question-user pair $(q, u) \in S_\text{q} \times S_\text{u}$ where S_q and S_u refer to the question set and user set respectively, generate a response r for this question-user pair (q, u), i.e., posted by user u for question q. The overall model architecture is described in Figure 1.

4.1 Tensor Factorization Module

To enable personalized response generation, we first need to automatically infer personalized signals that users demonstrate in their participation such as questions that they might interact with, as such signatures are often not explicitly available. To this end, we introduce *personalized response embedding* $\boldsymbol{p}_{i,j}$, a K-dimensional vector, to represent the latent relationship between a question i and a user j. We then form a tensor using all $\boldsymbol{p}_{i,j}$ over all question-user pairs and factorize this tensor, to learn latent interactions between questions, users, and their responses.

Formally, for a dataset with $I = |S_\text{q}|$ questions and $J = |S_\text{u}|$ users, we have a tensor $\mathcal{P} \in \mathbb{R}^{I \times J \times K}$ where $\mathcal{P}_{(i,j,:)} = \boldsymbol{p}_{i,j}$ denotes each (i, j) pair. The notation $P_{(i,j,:)}$ refers to the mode-3 fiber (or tube) of the tensor $\mathcal{P}$. $\mathcal{P}$ can be further formulated via Tucker Decomposition as follows:

$$\mathcal{P} = \mathcal{G} \times_1 \mathbf{Q} \times_2 \mathbf{U} \times_3 \mathbf{R}$$

Here $\mathbf{Q} \in \mathbb{R}^{I \times R_1}$, $\mathbf{U} \in \mathbb{R}^{K \times R_2}$, $\mathbf{R} \in \mathbb{R}^{K \times R_3}$ are the factor matrices, and $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is a core tensor. Once these factor matrices and core tensor are determined, the personalized response embedding $\boldsymbol{p}_{i,j}$ for any question-user pair (i, j) can be calculated as:

$$\boldsymbol{p}_{i,j} = \mathcal{P}_{(i,j,:)} = \mathbf{R}\mathcal{G}_{(3)} \left(\boldsymbol{u}_j \otimes \boldsymbol{q}_i\right)^\top$$

where $\boldsymbol{q}_i$ and $\boldsymbol{u}_j$ denote i-th and j-th row vector of $\mathbf{Q}$ and $\mathbf{U}$ respectively. $\otimes$ is the Kronecker product of two matrices.

Next, we introduce different mechanisms to incorporate TF modules especially $\boldsymbol{p}_{i,j}$ into traditional LSTM-based models and Transformer Language Models. This is essential to train better TF modules since it is impossible to directly supervise $\boldsymbol{p}_{i,j}$ as no ground truth is available.

4.2 LSTM-based Model with TF Module

To utilize TF module for standard LSTM-based Seq2Seq models, we propose to incorporate $\boldsymbol{p}_{i,j}$ into the initial hidden vector and cell vector of the LSTM decoder to help generate more personalized response, as personalized response embedding $\boldsymbol{p}_{i,j}$ is expected to also encode the target response:

$$\begin{aligned} \boldsymbol{h}_0^d &= (1 - \lambda) \cdot \boldsymbol{h}_m^e + \lambda \cdot \boldsymbol{p}_{i,j} \\ \boldsymbol{c}_0^d &= (1 - \lambda) \cdot \boldsymbol{c}_m^e + \lambda \cdot \boldsymbol{p}_{i,j}, \end{aligned} \tag{1}$$

Here λ is a coefficient to balance the information from the LSTM encoder and the personalized response embedding. Note that our TF module is agnostic to encoder-decoder frameworks, and can be applied to any Seq2Seq model similarly, including but not limited to Seq2Seq, Speaker model (Li et al., 2016), Seq2Seq with memory network (Zhang et al., 2018), and Speaker model with memory network. Figure 1 describes how the TF module is integrated with an LSTM-based Seq2Seq model. The TF module is randomly initialized and trained together with the Seq2Seq model. This allows TF module to access the supervision from the

output response, thus learn the latent interaction between users and questions and produce personalized response embedding for the decoder.

4.3 Transformer with TF Module

Recent success of DialoGPT (Zhang et al., 2020) on conversational response generation shows the potential of (pre-trained) transformer language model for the task of response generation. Thus we propose to incorporate TF module with transformer language model, (DialoGPT in specific) for personalized response generation. Since DialoGPT is a language model rather than a Seq2Seq model, it does not have a encoder-decoder architecture but only one transformer model. Thus we cannot utilize $\boldsymbol{p}_{i,j}$ as the initial hidden vector for decoder like that in Eq. 1. Instead, we propose to add personalized response embedding $\boldsymbol{p}_{i,j}$ with the input token embedding, token type embedding and positional embedding together as the input embedding to DialoGPT model. As shown in Figure 2, the personalized response embedding $\boldsymbol{p}_{i,j}$ is added to token "<EOS>", "klein" and "bleu" in the input to decode the j-th user's response for the i-th question. The TF module that produces $\boldsymbol{p}_{i,j}$ is also trained together with the DialoGPT model in an end-to-end fashion.

5 Experiments

5.1 Dataset

To study the task of personalized response generation with no explicit personalization information, we used a personalized Reddit dataset **PERCHAT**, consisting of 200,156 responses that users posted to different questions, from r/AskReddit[1] (Wu et al., 2021). Building upon Wu et al. (2021), we used active users who joined more than average discussions, and popular questions that received more comments. This led to 4724 users under 39,187 questions. These users and questions were sampled because they were active users who joined more discussions or popular questions that received more comments. We filtered all forms of url links, emails and digits into unique tokens "url", "email" and "digit". Replicated words and punctuation were processed to their standard forms. We sampled 3 responses for each user for users in the validation and test set, and the rest are used for training. The proportion of split size of train, validation, test is 171812 : 14172 : 14172.

[1] https://www.reddit.com/r/AskReddit/

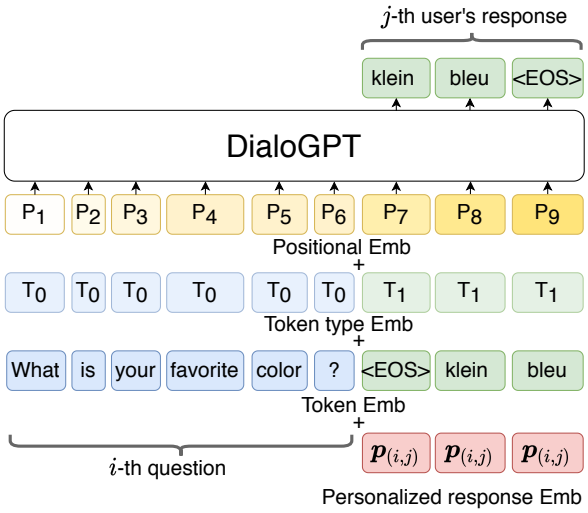

Figure 2: Input representation for DialoGPT model with TF module. TF module's personalized response embedding $\boldsymbol{p}_{i,j}$ is added with response token's word embedding, token type embedding and positional embedding.

5.2 Baselines and Our Models

We introduced several baselines for comparison with our proposed models. We introduced several baselines for comparison with our proposed models. (1) **DialoGPT**: A response generation model based on DialoGPT-medium provided in Zhang et al. 2019; (2) **Seq2Seq**: A standard Seq2Seq model with attention mechanisms with no personalization information; (3) **Speaker** model: Our implementation of the speaker model (Li et al., 2016). Following (Kottur et al., 2017), the Speaker embeddings were not initialized randomly but set as the average sentence embeddings from a user's all historical responses via sentence-BERT (Reimers and Gurevych, 2020); the dimension was reduced to 30 by principal component analysis. (4) **Memory** network: Our implementation of the generative memory network (Zhang et al., 2018) based on our Seq2Seq model with attention. We retrieved top-10 most relevant responses from a user for each question as the memory in the memory network; (5) **Memory+Speaker**: The generative memory network (Zhang et al., 2018), together with the use of the speaker embedding (Li et al., 2016).

Our models were based on the aforementioned baseline models by further incorporating our proposed TF module, i.e., the personalized response embedding from the TF module. **DialoGPT+TF** is a DialoGPT model with personalized response embedding added to each time step at the decoding stage shown in Figure 2. **Seq2Seq+TF**, **Speaker+TF**, **Memory+TF**, **Mem-**

ory+Speaker+TF are constructed on top of our baseline models with personalized response embedding added to the decoder as Eq. 1.

5.3 Evaluation Metrics

We evaluated different models with F1, BLEU, Distinct-N, perplexity (PPL), and our proposed Per-Hits@k. Here, F1 (Dinan et al., 2019) refers to the harmonic mean of precision and recall computed based on the tokens between generated and ground truth response. BLEU (Papineni et al., 2002) was first proposed for machine translation but is also widely used for evaluating response generation. Distinct-N (Li et al., 2015) aims to evaluate lexical diversity and we tested distinct unigrams (Distinct-1) and bigrams (Distinct-2). We used perplexity to evaluate the fluency of the generation model.

Per-Hits@k for Personalization Evaluation To evaluate the personalization in generated responses for a user, one needs to have a good understanding of that particular user who might sometimes have a very long posting history (500 responses per user on average in our dataset), making it hard for annotators to evaluate how personalized the generated response is for a user. Besides, not every response from a user will reveal their personalization information. Thus, we propose an automatic evaluation metric to evaluate the personalization degree of different generation models called Per-Hits@k. Suppose we have N users and there are M_i responses generated for user i to be evaluated. We firstly train a user-specific language model LM_i for each user i on all their responses in training set. We then test the j-th response's perplexity of user i on all users' language models, and denote its perplexity on user-n's language model as $\mathrm{ppl}^n_{i,j}$. We rank the perplexity of user i's j-th response over N user language models (the lower the perplexity, the higher rank), and denote the ranking of the perplexity on user i's language model LM_i with $\mathrm{rank}(\mathrm{ppl}^i_{i,j})$. We define the value of Per-Hits@k in Per-Hits@k metric as:

$$\text{Per-Hits@}k = \frac{1}{\sum_{i=1}^{N} M_i} \sum_{i=1}^{N} \sum_{j=1}^{M_i} \mathbf{1}_{x \le k}(\mathrm{rank}(\mathrm{ppl}^i_{i,j}))$$

This measures how likely the generated response will be ranked as top-k with its corresponding user language model among N users. In our implementation, we fine-tuned GPT-2 (small) (Radford et al., 2019) for each user i to instantiate this user i's language model LM_i. To ensure the quality of LM_i, we only consider a subset of users ($N = 500$) and choose these users who have the most responses.

5.4 Implementation Details

We implemented our models with PyTorch (Paszke et al., 2019). For TF module, the core tensor is of size $50 \times 50 \times 50$, dimension of personalized response embedding is 512 for all Seq2Seq-based models with TF module (denote as Seq2Seq-based+TF), while it is 1024 for the DialoGPT+TF model. For any Seq2Seq-based+TF model, both encoder and decoder have 2 LSTM layers with hidden size of 512, while DialoGPT+TF model is based on the pre-trained medium DialoGPT model with hidden size of 1024. Any word appears more than three times were included in the vocabulary of Seq2Seq-based+TF models, and the size of the vocabulary is 30K. DialoGPT+TF model uses the pre-trained Byte-Pair-Encoding (BPE) tokenizer of size 50,257. The λ coefficient in Eq. 1 is set to 0.2. Adam (Kingma and Ba, 2014) is used as the optimizer and the learning rate was set to 1e-3 for TF-Speaker model and 1e-5 for TF-DialoGPT by grid search. Top-k ($k = 2$) sampling (Fan et al., 2018) was used without any re-scoring techniques to generate response at test stage. We selected models with the highest average Per-Hits@k ($k = 1, 2, 3, 4, 5$) on validation set.

5.5 Results

As shown in Table 1, we reported F1, BLEU, Distinct-N and Per-Hits@k on test data. Distinct-N and Per-Hits@k on ground truth test data and Per-Hits@k on random ranking were also reported. Overall, we found that TF based models significantly improved the personalization metric Per-Hits@k compared to all baselines, with comparable and even better performances in terms of other metrics. Specifically, our proposed Seq2Seq+TF model had an average hist@k score 4 times higher than the Seq2Seq baseline and the Memory+Speaker+TF model had the highest personalization score. This demonstrates that our proposed TF module can model users' personalization well using users' posting history. Furthermore, **1)** Per-Hits@k on ground truth data was far below its upper bound 100% but still much higher than Per-Hits@k of generation models, showing the effectiveness of our Per-Hits@k metric to evaluate user personalization. For example, a Per-Hits@1 score of 9.47% indicated that 9.47% of the ground truth responses were ranked as top-1 by its users' language model over

Method	F1 %	BLEU %	Distinct-N % D-1	Distinct-N % D-2	PPL	Per-Hits@k % 1	2	3	4	5	Avg.
Random ranking	-	-	-	-	-	0.20	0.40	0.60	0.80	1.00	0.60
Ground truth	-	-	7.25	45.51	-	9.47	15.73	19.93	23.00	25.40	18.71
DialoGPT	13.64	0.86	3.24	18.68	27.20	0.40	0.67	1.00	1.27	1.60	0.99
Seq2Seq	14.42	1.22	0.66	4.01	92.24	0.60	0.87	1.00	1.20	1.53	1.04
Speaker	15.34	1.41	2.79	14.27	98.75	1.00	1.93	2.93	3.40	3.80	2.61
Memory	14.42	1.28	3.34	16.36	108.27	1.27	2.20	2.73	3.20	3.67	2.61
Memory+Speaker	14.60	1.11	3.52	17.64	110.31	1.53	2.73	3.47	4.33	5.00	3.41
DialoGPT+TF	13.61	0.80	**3.61**	**20.40**	**27.01**	0.53	1.00	1.27	1.67	1.73	1.24*
Seq2Seq+TF	**15.40***	1.58*	3.20*	15.95*	105.21	2.07*	3.20*	4.53*	5.40*	5.80*	4.20*
Speaker+TF	15.33	**1.59**	3.35*	17.19*	107.88	2.07*	3.33*	3.80	4.67	5.40*	3.85*
Memory+TF	14.99*	1.38*	3.52	16.69	107.46	2.40*	3.33*	4.00*	5.00*	5.60*	4.07*
Memory+Speaker+TF	14.99*	1.40*	3.34	16.29	107.55	**2.60***	**4.00***	**4.93***	**6.00***	**6.53***	**4.81***

Table 1: Performance comparison with baselines. A Wilcoxon signed-rank test was performed for Per-Hits@k and paired t-test was performed for other metrics, the significant ones ($p < 0.05$) over its baseline are marked as *.

Method	F1 %	BLEU %	Distinct-N % D-1	Distinct-N % D-2	PPL	Per-Hits@k % 1	2	3	4	5	Avg.
Ground truth	-	-	26.51	73.31	-	100	100	100	100	100	100
DialoGPT	15.67	0.11	28.33	65.18	**20.96**	1.41	2.11	2.82	2.82	2.82	2.39
Seq2Seq	16.05	0.47	21.69	52.29	60.10	2.11	2.11	2.82	4.93	4.93	3.38
Speaker	19.69	4.40	19.42	48.47	55.81	3.52	6.34	9.15	9.86	9.86	7.75
Memory	19.02	4.33	23.08	54.56	60.44	4.23	7.04	8.45	9.15	9.86	7.75
Memory+Speaker	19.89	3.18	22.98	58.51	59.03	4.93	7.75	11.97	14.79	16.20	11.13
DialoGPT+TF	15.11	0.17	**30.29**	**65.19**	21.30	2.11	2.82	2.82	3.52	3.52	2.96*
Seq2Seq+TF	**22.98***	**5.77**	20.43	49.75	57.38	9.86*	13.38*	**16.90***	**16.90***	17.61*	14.93*
Speaker+TF	20.70	4.16	22.72*	53.02*	56.50	10.56*	13.38*	14.08	15.49	15.49	13.80*
Memory+TF	21.31*	3.10	23.45	55.10	57.65	**11.27***	12.68	13.38	15.49	16.20	13.80*
Memory+Speaker+TF	20.79	2.31	23.67	58.58	57.64	10.56*	**14.08***	15.49*	**16.90***	**18.31***	**15.07***

Table 2: Performance comparison with baselines on top-1 focused test set. A Wilcoxon signed-rank test was performed for Per-Hits@k and paired t-test was performed for other metrics, the significant ones ($p < 0.05$) over its baseline are marked as *.

the 500 users. One explanation why Per-Hits@1 on ground truth data was far below 100% might be that these responses from a user do not necessarily always reveal their persona. **2)** Although both Seq2Seq and DialoGPT did not model user personalization explicitly, they had higher than random Per-Hist@k. **3)** Compared to Seq2Seq, both Speaker and Memory model had about double Per-Hits@k and some degree of improvements over BLEU, F1, and Distinct-N. Combining the Memory and Speaker models led to further improvement on Per-Hits@k. Seq2Seq model with personalized response embedding form TF module (Seq2Seq+TF) achieved higher Per-Hits@k than all baselines, and our Memory+Speaker+TF model showed the highest Per-Hits@k score, demonstrating the effectiveness of our proposed TF module in capturing user personalization by learning the latent interactions between questions, users, and their responses. **4)** Compared to Seq2Seq model, DialoGPT performed worse on content relatedness measures like BLEU and F1 and personalization measure Per-Hits@k. But our TF module still improved the personalization on top of DialoGPT model, as well as the diversity measure Distinct-N. Note that the perplexity could not be compared between DialoGPT and LSTM-based models since they have different vocabulary sets. **5)** Memory+Speaker model had better Per-Hits@k but lower BLEU than Seq2Seq model, while our TF module improved Memory+Speaker model's BLEU and Per-Hits@k at the same time. Due to the open-ended nature of these discussions, we observed relatively low BLEUs across different models, in line with prior work on personalized generation (Zheng et al., 2020; Li et al., 2016).

Since we have relatively high Per-Hits@k on the ground truth test set, we hypothesize that those top ranked responses in the ground truth test set by Per-Hits@k might be more likely to contain user personalization information. In other words, for certain question-user pairs, a user is more likely to respond with some personalized content that could be better recognized by their language model. We

Method	F1 %	BLEU %	Distinct-N % D-1	Distinct-N % D-2	PPL	Per-Hits@k % 1	2	3	4	5	Average
Random	15.45	1.36	**3.09**	14.90	**97.05**	**1.33**	1.73	1.87	2.53	2.93	2.08
History	15.34	1.41*	2.79	14.27	98.75	1.00	1.93	2.93*	3.40	3.80	2.61
TF-u	15.24	1.48*	2.48	12.52	101.77	1.07	1.60	2.40	2.73	2.93	2.15
History+TF-u	**15.60***	**1.59***	3.02	**15.42***	101.30	**1.33**	**2.67**	**3.67***	**4.20***	**4.93***	**3.36***

Table 3: Speaker model with different speaker embedding initialization methods. A Wilcoxon signed-rank test was performed for Per-Hits@k and paired t-test was performed for other metrics, the significant ones ($p < 0.05$) over Random are marked as *

top-m	Per-Hits@k from Seq2Seq+TF 1	2	3	4	5	Avg.
1	9.86	13.38	16.90	16.90	17.61	14.93
2	6.78	9.32	11.86	11.86	12.71	10.51
3	6.35	8.36	11.04	11.37	12.04	9.83
4	5.51	7.54	9.86	11.01	11.59	9.10
5	4.99	6.82	8.92	9.97	10.50	8.24
500	2.07	3.20	4.53	5.40	5.80	4.20

Table 4: Per-Hits@k on different top-m focused test sets.

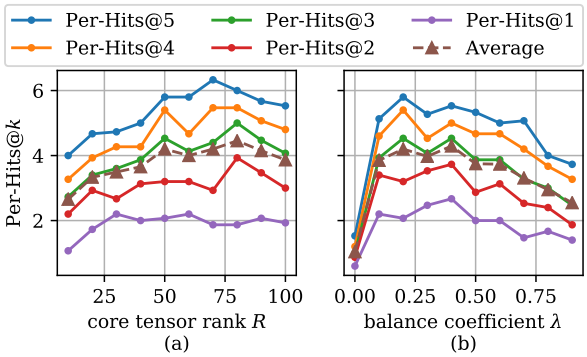

Figure 3: Per-Hits@k of Seq2Seq+TF model with (a): different Tucker's rank; (b): different balancer λ.

denote these question-user pairs that are ranked top-k by the Per-Hits@k from the test set as the top-m focused set. We evaluated Per-Hits@k of Seq2Seq+TF on different top-m ($m = 1, 2, 3, 4, 5$) test set in Table 4. Note that top-500 is the full test set we used for Per-Hits@k in Table 1. Per-Hits@k was higher on smaller top-m test set, showing the effectiveness of our Per-Hits@k measure, because Per-Hits@k of the same Seq2Seq+TF model was higher on the focused question-user subset when m is small, while lower on the larger and general test set. We then evaluated the baselines and our proposed models on top-1 focused test set in Table 2. Compared to the results on the full test set (Table 1), the gaps between our models and baselines on BLEU, F1, and Per-Hits@k are larger on this top-1 test set. This suggests that our TF module can help generate more personalized response for a user, especially in a context where a user is more likely to write personalized response.

5.6 Analysis and Ablation Studies

The Rank of Tucker Decomposition We first studied the influence of the rank of Tucker decomposition used in our TF module, i.e. the shape of the core tenser $\mathcal{G}$. We trained Seq2Seq+TF model with core tensor of shape $R \times R \times R, R \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. From Figure 3(a), we found that Per-Hits@k first increased along with the rank, indicating that TF module with higher rank might better model latent user-questioninteractions. When the rank reaches around 50, there seems to be limited averaged gains on Per-Hits@k. Thus, we chose core tensor of shape $50 \times 50 \times 50$ for our TF module.

The Balancer λ We then studied the influence of the λ coefficient in Eq 1 which is used to balance the question information from the encoder and personalized response embedding from the TF module. We varied Seq2Seq+TF model's λ from 0 to 1, as shown in Figure 3(b). Note that Seq2Seq+TF with $\lambda = 0$ is the Seq2Seq baseline. We observed that Per-Hits@k increased a lot when λ changed from 0 to 0.1, confirming the effectiveness of our proposed TF module in modeling user personalization. Moreover, TF module was not sensitive to the hyper-parameter λ as Per-Hits@k were stable for $\lambda \in [0.1, 0.4]$. Per-Hits@k decreased when λ was larger than 0.4, suggesting the importance to balance the encoder and TF module.

User Factor Matrix To examine whether the TF module has learned user personalization information in user factor matrix $\mathbf{U}$, we trained a Speaker model that initialized the speaker embeddings with user embeddings in $\mathbf{U}$ and other initialization methods. Specifically we studied the user factor matrix (**TF-u**) from the Seq2Seq+TF model in Table 1 and compared it with: 1) random speaker embeddings

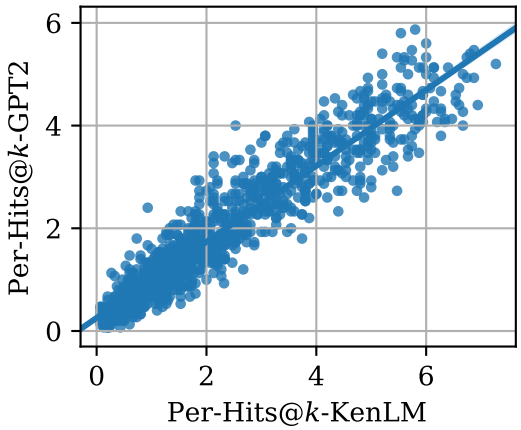

Figure 4: Per-Hits@k calculated by GPT2 and KenLM for different models. Pearson correlation r=0.941, with $p < .001$.

(**Random**) and 2) average sentence embeddings of each user's historical responses (**History**) which is used in our Speaker model baseline; 3) we further concatenated the history embeddings and our user embeddings in $\mathbf{U}$ to be the initial Speaker embeddings (**History+TF-u**). The results of the four variants of Speaker model are shown in Table 3. We found that both History and TF-u initialization improved Per-Hits@k over Random to some extent, suggesting that our TF module has learned some degree of user personalization in its user factor matrix $\mathbf{U}$. Although TF-u had smaller Per-Hits@k improvement over Random, History+TF-u has the best Per-Hits@k, indicating that the personalization information learned by TF module is different to that from users' posting history.

Robustness of Personalization Metric To test the robustness of our Per-Hits@k metric, we trained trigram language models with the **KenLM** toolkit (Heafield et al., 2013) for the user specific language models used in Per-Hits@k. While GPT-2 is a transformer-based language model pre-trained on large corpus and can be fine-tuned on each user's corpus, KenLM is impossible to follow this approach because it can only be trained in an end-to-end way, i.e. language models of KenLM is directly trained on each user's corpus. Thus we had two Per-Hits@k variants: Per-Hits@k-GPT2 (the one we used in previous sections) and Per-Hits@k-KenLM. We evaluated Per-Hits@k-GPT2 and Per-Hits@k-KenLM for all the models we trained with different settings and plot all (Per-Hits@k-KenLM, Per-Hits@k-GPT2) pairs for $k \in \{1, 2, 3, 4, 5\}$ in Figure 4. With a correlation of 0.941 between two variants, we conclude that Per-Hits@k is robust because it produces consistent and similar judgements regardless of which language model it uses.

6 Conclusion and Discussion

This work proposed a tensor factorization module to model user personalization from users' posting history for the task of personalized response generation, where explicit persona or demographic information is unavailable. To automatically evaluate the personalization of generated response, we proposed a new evaluation metric called Per-Hits@k. Extensive experiments on a large-scale dataset show that our proposed TF module outperforms previous methods significantly in terms of its content generation quality and also the personalization of generated responses. Our ablation studies further demonstrated the effectiveness and robustness of our TF based generation framework.

One limitation to note for our work is that our tensor factorization based framework to model personalization has only been tested on a corpus derived from Reddit (Wu et al., 2021). We acknowledge that potential user population bias might be introduced in this process. Another limitation of our results lies in dealing with new users, i.e., the cold start problem. Future research could further examine these issues, build upon our work to examine how different types of implicit information such as social knowledge and commonsense might be learned together with these user profiles in this tensor factorization manner, and model personalization in multi-turn dialogue systems.

Acknowledgment

We would like to thank the anonymous reviewers, and the members of Georgia Tech SALT group for their feedback. This work is supported in part by grants from Amazon, Salesforce, and the Institute for Data Engineering and Science (IDEaS).

References

JinYeong Bak and Alice Oh. 2020. Speaker sensitive response evaluation model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6376–6385, Online. Association for Computational Linguistics.

Dheeraj kumar Bokde, Sheetal Girase, and Debajyoti Mukhopadhyay. 2015. Role of matrix factorization model in collaborative filtering algorithm: A survey. *CoRR, abs/1503.07475*.

Stanley F Chen, Douglas Beeferman, and Roni Rosenfeld. 1998. Evaluation metrics for language models.

Cesc Chunseong Park, Byeongchang Kim, and Gunhee Kim. 2017. Attend to you: Personalized image captioning with context sequence memory networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 895–903.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2019. The second conversational intelligence challenge (convai2). *arXiv preprint arXiv:1902.00098*.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H Clark, and Philipp Koehn. 2013. Scalable modified kneser-ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, László Lukács, Marina Ganea, Peter Young, et al. 2016. Smart reply: Automated response suggestion for email. In *KDD*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Satwik Kottur, Xiaoyu Wang, and Vítor Carvalho. 2017. Exploring personalized neural conversational models. In *IJCAI*, pages 3728–3734.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003.

Xiaohu Liu, Ruhi Sarikaya, Liang Zhao, Yong Ni, and Yi-Cheng Pan. 2016. Personalized natural language understanding. In *INTERSPEECH*, pages 1146–1150.

Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. 2019. Learning personalized end-to-end goal-oriented dialog. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6794–6801.

Kaixiang Mo, Shuangyin Li, Yu Zhang, Jiajun Li, and Qiang Yang. 2016. Personalizing a dialogue system with transfer reinforcement learning. *arXiv preprint arXiv:1610.02891*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037.

Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. Assigning personality/profile to a chatting machine for coherent conversation generation.

Ella Rabinovich, Raj Nath Patel, Shachar Mirkin, Lucia Specia, and Shuly Wintner. 2017. Personalized machine translation: Preserving original author traits. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1074–1084.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. *arXiv preprint arXiv:2004.09813*.

Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19(1):10–26.

Arthur A Stone, Christine A Bachrach, Jared B Jobe, Howard S Kurtzman, and Virginia S Cain. 1999. *The science of self-report: Implications for research and practice*. Psychology Press.

Bakhtiyar Syed, Gaurav Verma, Balaji Vasan Srinivasan, Anandhavelu Natarajan, and Vasudeva Varma. 2020. Adapting language models for non-parallel author-stylized rewriting. In *AAAI*, pages 9008–9015.

Ledyard R Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311.

Olga Vechtomova, Hareesh Bahuleyan, Amirpasha Ghabussi, and Vineet John. 2018. Generating lyrics with variational autoencoder and multi-modal artist embeddings. *arXiv preprint arXiv:1812.08318*.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Yuwei Wu, Xuezhe Ma, and Diyi Yang. 2021. Personalized response generation via generative split memory network. In *North American Chapter of the Association for Computational Linguistics*.

Minghong Xu, Piji Li, Haoran Yang, Pengjie Ren, Zhaochun Ren, Zhumin Chen, and Jun Ma. 2020. A neural topical expansion framework for unstructured persona-oriented dialogue generation. *arXiv preprint arXiv:2002.02153*.

Jingfeng Yang, Diyi Yang, and Zhaoran Ma. 2020. Planning and generating natural and diverse disfluent texts as augmentation for disfluency detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1450–1460.

Min Yang, Qiang Qu, Kai Lei, Jia Zhu, Zhou Zhao, Xiaojun Chen, and Joshua Z Huang. 2018. Investigating deep reinforcement learning techniques in personalized dialogue generation. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 630–638. SIAM.

Min Yang, Zhou Zhao, Wei Zhao, Xiaojun Chen, Jia Zhu, Lianqiang Zhou, and Zigang Cao. 2017. Personalized response generation via domain adaptation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1021–1024.

Kaisheng Yao, Geoffrey Zweig, and Baolin Peng. 2015. Attention with intention for a neural network conversation model. *arXiv preprint arXiv:1510.08565*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL, system demonstration*.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2019. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *AAAI*, pages 9693–9700.