Learning to Abstract for Memory-augmented Conversational Response Generation

Zhiliang Tian,[1,3]* **Wei Bi,**[2] **Xiaopeng Li,**[1,3] **Nevin L. Zhang**[1,3]
[1]Department of Computer Science and Engineering,
The Hong Kong University of Science and Technology, Hong Kong
[2]Tencent AI Lab, Shenzhen, China
[3]HKUST-Xiaoi Joint Lab, Hong Kong
ztianac@cse.ust.hk victoriabi@tencent.com {xlibo,lzhang}@cse.ust.hk

Abstract

Neural generative models for open-domain chit-chat conversations have become an active area of research in recent years. A critical issue with most existing generative models is that the generated responses lack informativeness and diversity. A few researchers attempt to leverage the results of retrieval models to strengthen the generative models, but these models are limited by the quality of the retrieval results. In this work, we propose a memory-augmented generative model, which learns to abstract from the training corpus and saves the useful information to the memory to assist the response generation. Our model clusters query-response samples, extracts characteristics of each cluster, and learns to utilize these characteristics for response generation. Experimental results show that our model outperforms other competitive baselines.

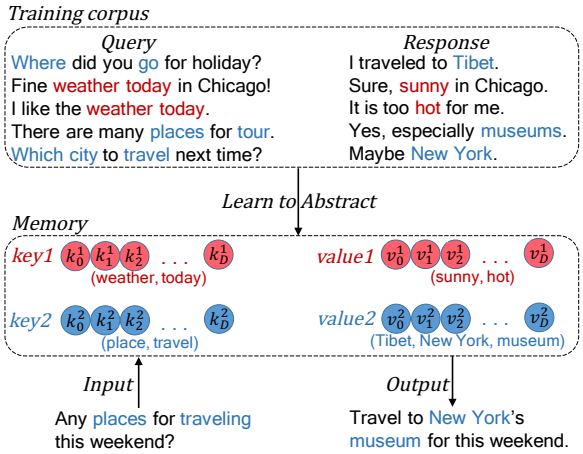

Figure 1: An example of abstracting training corpus and memorizing their characteristics in the form of key vectors and value vectors. Red and blue indicate two clusters. The input query matches the blue one and generates the response assisted by information collected from the last two training samples.

1 Introduction

Automatic human-computer dialogue / conversation is a core topic in natural language processing. There is a boom in research on open-domain chit-chat dialogue systems due to the availability of vast conversational data online. Most existing models of dialogue systems can be divided into retrieval-based models and generative models.

Given a query, retrieval-based (Ji *et al.*, 2014) models search for the most similar query stored in the training corpus and directly copy its corresponding response as the result. These models cannot create new replies customized for the given queries. Generative models (Shang *et al.*, 2015) learn a query-response mapping to generate responses by maximizing $P(r|q)$, where q is the input query and r is the response. The most popular generative model is the Sequence-to-Sequence (Seq2Seq) model (Sutskever *et al.*, 2014), which generates new utterances tailored for queries and achieves high coherence between queries and generated utterances. However, existing generative models often generate uninformative and universal responses (Li *et al.*, 2016a).

To address these issues, several researchers leverage retrieved results R to augment the information used in generative models. Such methods are called retrieval-augmented generative models and their objectives are to maximize $P(r|q, R)$, where R is one or a few (at most 3 in practice) retrieved results. Particularly, some researchers (Li *et al.*, 2017; Zhuang *et al.*, 2017; Song *et al.*, 2018) build the combination of retrieval and generative models, which retrieve one or a few responses r^+, and then feeds both the query q and r^+ into the generative model to maximize $P(r|q, R = r^+)$. It enriches generated responses by informatively retrieved responses but can only utilize a limited

*Work done while Zhiliang Tian was collaborating with Tencent AI Lab.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3816–3825
Florence, Italy, July 28 - August 2, 2019.

number of retrieved results due to their model architecture. Wu *et al.* (2018) edit the retrieved response r^+ with the Seq2Seq model based on the lexical differences between the input query q and its retrieved query q^+, whose objective is to maximize $P(r|q, R = \langle r^+, q^+ \rangle)$. It edits retrieved responses r^+ to make them relevant to queries, but their edited results rely heavily on the sentence pattern of r^+. Generally, the responses from such models are more informative and diverse than those from plain generative models, while maintain better relevance than the retrieved responses.

Although current retrieval-augmented generative models have achieved promising results, they still have following weaknesses: Firstly, they are limited by the quality of the retrieved results. Retrieval results are less coherent and relevant with query than generative models' (Song *et al.*, 2018). Irrelevant retrieved results would mislead the response generation. Secondly, these models can only utilize individual retrieved results, which makes the generation sensitive to those results, leading to a high variance in the performance. Moreover, the information from very few retrieved results may not be sufficient to enrich the response generation.

In this paper, we propose a memory-augmented generative model that memorizes and utilizes the common characteristics M of groups of query-response (q-r) pairs to enhance the response generation by maximizing $P(r|q, M)$. The advantage is that our model is less sensitive to the quality of individual q-r pairs and hence increases the robustness of response generation.

In particular, we divide the training corpus into multiple groups by clustering, extract common characteristics of each group, and learn to utilize the characteristics to assist generation. The idea is illustrated in Figure 1 (top), the training corpus is divided into two sets of closely related queries and their responses. We abstract query-response relationship hidden in those q-r pairs, save them to the memory (Figure 1 bottom), and use those relationships for response generation.

Our contributions can be summaried as:

1. We are the first to extract information from clusters of query-response pairs using a learnable memory, and to use the information to enhance the performance of conversation systems.

2. We propose a novel framework where the Seq2Seq, autoencoder and clustering model are jointly trained to abstract the training corpus and generate responses.

3. Our model outperforms state-of-the-art generative models and retrieval-augmented generative models in single-round conversation scenarios.

2 Related Work

Generative models build dialogue systems via end-to-end training. Ritter *et al.* (2011) first regard response generation as query-to-response translation. Following that, Shang *et al.* (2015) implement an end-to-end dialogue system borrowing the Seq2Seq model, while Li *et al.* (2016b) replace the maximum likelihood criterion with maximum mutual information (MMI) to deal with the universal response issue of the seq2seq.

The retrieval-based models are another branch in building dialogue systems. Ji *et al.* (2014) propose to apply information retrieval techniques to search for related queries and replies. Zhou *et al.* (2016) and Yan *et al.* (2016) improve it by neural networks.

Recently, several researchers (Song *et al.*, 2018; Li *et al.*, 2017; Zhuang *et al.*, 2017) propose to merge retrieval-based models and generative models. Cai *et al.* (2018) generate the response skeleton from the retrieved results and revise the skeletons to get the response. Guu *et al.* (2018) use the Seq2Seq model to edit a prototype from the retrieved results for text generation and Wu *et al.* (2018) leverage context lexical differences to edit prototypes for conversation.

There are some other directions to enhance generative models by adding additional information. Some of them introduce the knowledge base to conversation models, which provide task-specific knowledge (Madotto *et al.*, 2018; Wu *et al.*, 2019) or lexical knowledge to text generation (Young *et al.*, 2018; Parthasarathi and Pineau, 2018). Some other directions are to maintain sample-level temporary memory. Weston *et al.* (2014), Shang *et al.* (2015), Serban *et al.* (2016), Tian *et al.* (2017) and Le *et al.* (2018) memorize the previous utterances in a multi-round session. Unlike them, our model brings in the corpus-level memory and does not rely on any external resource.

3 Models

3.1 Model Architecture

Our model consists of two components: a memory module and a generative model. The memory module divides the training corpus into multiple groups of query-response pairs, and it extracts and memorizes the essential query-response correspondence information hidden in each group of pairs. The generative model generates responses for input queries and, while doing so, takes information stored in the memory module into consideration. It also learns the representations of queries and responses that are used in the memory module.

3.2 Query-Response Memory Module

Our memory module consists of K memory slots, and each memory slot is a pair containing a key cell and its corresponding value cell. Given a query, we search for the most similar key and output its corresponding value. Both the keys and the values are real-value vectors. They are called key embeddings and value embeddings respectively, and denoted as $\boldsymbol{k}_i$ and $\boldsymbol{v}_i$.

We group queries in the training corpus into K clusters. Each query is embedded as a vector. So, it makes sense to talk about the center $\boldsymbol{k}_i$ of each query cluster i. The center $\boldsymbol{k}_i$ is used as a key in our memory module. The corresponding value $\boldsymbol{v}_i$ is a vector that captures the common characteristics of the responses to queries in cluster i. We will say more about $\boldsymbol{k}_i$ and $\boldsymbol{v}_i$ later.

Read Operation. In our model, the input of Read Operation is the current query's representation $\boldsymbol{e_q}$ from the generative model. Given the $\boldsymbol{e_q}$, the Read Operation addresses the memory by the similarity between the current query $\boldsymbol{e_q}$ and every memorized key embedding $\boldsymbol{k}_i$, in which we apply a dot-product operation to measure the similarity.

We design two modes to fetch the value: **Soft Read** is a weighted summation over all K value embeddings in the whole memory according to the normalized similarity scores (Eq. 1). **Hard Read** is to fetch the value embedding whose key embedding is most similar to the current query $\boldsymbol{e_q}$ (Eq. 2). Finally, it returns the value as the output of the read operation.

$$\begin{aligned} \text{SoftRead}(\boldsymbol{e_q}) &= \sum_{i=1}^{K} \alpha_i \boldsymbol{v}_i, \\ \alpha_i &= softmax(\boldsymbol{k}_i \cdot \boldsymbol{e_q}). \end{aligned} \tag{1}$$

$$\text{HardRead}(\boldsymbol{e_q}) = \{\boldsymbol{v}_i | i = \underset{i=1}{\overset{K}{\text{argmax}}}(\boldsymbol{k}_i \cdot \boldsymbol{e_q})\}. \tag{2}$$

Write Operation. We collect the query representation $\boldsymbol{e_q}$'s and the response representation $\boldsymbol{e_r}$'s of all the training samples from the generative model, and then conduct K class K-Means clustering using $\boldsymbol{e_q}$'s. We let the center of the i-th cluster C_i be key embedding $\boldsymbol{k}_i$, and let the average of the representations of the responses to queries in C_i be value embedding $\boldsymbol{v}_i$ (Eq. 3).

$$\boldsymbol{v}_i = \frac{\sum_{j \in C_i} \boldsymbol{e_r}_j}{|C_i|}. \tag{3}$$

In this way, each key embedding $\boldsymbol{k}_i$ gathers similar queries together and obtains their representative information by fetching their cluster center. Each value embedding $\boldsymbol{v}_i$ retains the common characteristics of a group of responses $\boldsymbol{e_r}$ whose queries are similar. Hence, the pair $\langle \boldsymbol{k}_i, \boldsymbol{v}_i \rangle$ can be regarded as an abstraction of the query-response correspondence relationship hidden in the i-th cluster of queries and their responses. We can control the granularity of the abstraction by varying the number of clusters K.

In an extreme setting, if we set the memory size K equal to the training corpus size and use the hard read operation, our model nearly degenerates into a retrieval-augmented generative model. In this case, the generation relies on only one retrieved sample, the generated response becomes sensitive to the quality of that single sample and is restrained by the pattern of the single sample.

3.3 Memory-Augmented Response Generative Model

Our overall model consists of two branches (Figure 2). The top branch is a memory-augmented Seq2Seq (M-Seq2Seq) model that is used for response generation. The lower branch is a conditional autoencoder (CAE) that is used to learn response representations necessary for the memory writing.

The input to the M-Seq2Seq branch is a query q. It is first passed through an encoder to get a

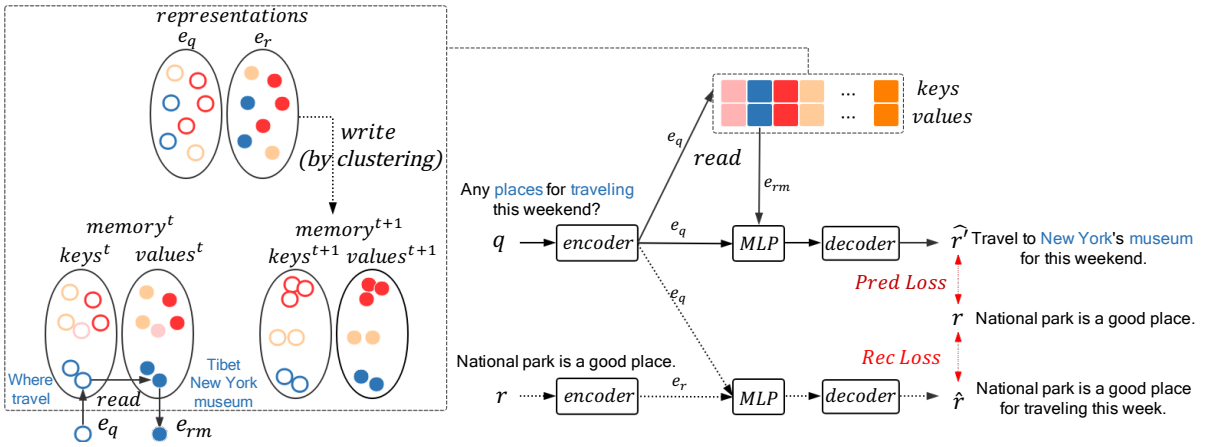

Figure 2: The architecture of our model. Solid arrows show both the training and generation (testing) processes; dashed arrows show the training process. The left part shows how to read memory at t-th step and write to update the memory from t-th to $(t+1)$-th step, where t indicates the step of updating the memory. The callout illustrates that a query matches the blue memory slot and reads its value "Tibet", "New York", and "museum" to promote its response generation. "Pred Loss" and "Rec Loss" mean the prediction and reconstruction loss respectively.

representation $\boldsymbol{e_q} = Encoder(q)$. The memory is then read using $\boldsymbol{e_q}$ as the key, and the output of the memory read is $\boldsymbol{e_{rm}}$. After that, $\boldsymbol{e_q}$ and $\boldsymbol{e_{rm}}$ are merged by an *MLP* (multi-layer perceptron), and then the merged results are fed into the decoder to decode the final response $\hat{r'}$, which is the generated response for the query q. The objective function for this branch is the first term of Eq. 8

During training, we feed $\langle q, r \rangle$ pairs to our model. The query q is fed to the M-Seq2Seq branch, while the corresponding response r is fed to the CAE branch. Similar to the previous case, r is first pushed through to get a representation $\boldsymbol{e_r} = Encoder(r)$. Then both $\boldsymbol{e_q}$ and $\boldsymbol{e_r}$ are fed to an *MLP* and then a decoder. The output is $\hat{r}$, a reconstructed version of r. The reconstruction loss is the second term of Eq. 8. We formalize the operations of M-Seq2Seq and CAE by Eq. 4 to Eq. 7.

Note that $\boldsymbol{e_q}$ is feed to the CAE for two reasons. First, it makes the embedding $\boldsymbol{e_r}$ of r dependent on the embedding $\boldsymbol{e_q}$ of q. Second, it makes the two branches work in a similar fashion so that the representaions learnt by CAE is adaptive to M-Seq2Seq. The CAE branch tries to reconstruct $\hat{r}$ from $\boldsymbol{e_r}$ and $\boldsymbol{e_q}$, while the M-Seq2Seq tries to generate an appropriate response $\hat{r'}$ from $\boldsymbol{e_q}$ and $\boldsymbol{e_{rm}}$. $\boldsymbol{e_{rm}}$ can be viewed as a rough estimation of $\boldsymbol{e_r}$ and hence is helpful in improving the quality of the generated response.

$$\boldsymbol{e_q} = Encode(q), \qquad \boldsymbol{e_r} = Encode(r), \tag{4}$$

$$\boldsymbol{e_{rm}} = Read(\boldsymbol{e_q}), \tag{5}$$

$$\boldsymbol{z} = MLP([\boldsymbol{e_r}, \boldsymbol{e_q}]), \quad \boldsymbol{z'} = MLP([\boldsymbol{e_{rm}}, \boldsymbol{e_q}]), \tag{6}$$

$$\hat{r} = Decode(\boldsymbol{z}), \qquad \hat{r'} = Decode(\boldsymbol{z'}). \tag{7}$$

The overall objective function contains two parts as shown in Eq. 8 : the prediction loss (first term) is derived from the general objective of the retrieval- or memory-augmented generative models $\max P(r|q, M)$, where we set $M = \boldsymbol{e_{rm}}$ for our model. The reconstruction loss (second term) is for learning the representations by reconstructing r, whose target is to improve the memory module so as to improve the $\boldsymbol{e_{rm}}$ for enhancing the generative model. In addition, λ is a factor to balance the losses.

$$\begin{aligned} L = & E_{q,r\sim\mathcal{D}} \log P(\hat{r}|q, \boldsymbol{e_{rm}}) \\ & + \lambda \cdot E_{q,r\sim\mathcal{D}} \log P(\hat{r}|q, r). \end{aligned} \tag{8}$$

3.4 Joint Training and Generation

To enable the memory module and the generative model to work together, we combine and jointly train them. We separate the memory writing and the generative model training into two phases, and then train the two phases alternatively. The two training phases switch once per epoch, which means we conduct the memory writing once the generative model finishes training current epoch.

In the generative model training phase, we train and update the model while keeping the memory module read-only. The generative model reads from the memory, trains to update itself, and collects representations $\boldsymbol{e_q}$'s and $\boldsymbol{e_r}$'s in preparation for memory writing. In the memory writing phase, parameters of the generative model are fixed. We conduct the clustering over all representations $\boldsymbol{e_q}$'s and $\boldsymbol{e_r}$'s collected from generative model training, and then write the results into the memory.

For response generation (testing) phase, we only rely on the M-Seq2Seq branch and the memory module in the read-only mode, since we cannot observe the r during generation. As indicated by the solid lines in Figure 2, it encodes q to acquire $\boldsymbol{e_q}$, reads out estimated response $\boldsymbol{e_{rm}}$ by the Read Operation, and goes through the *MLP* and decoder to generate $\hat{r}$.

4 Experimental Settings

4.1 Dataset

In our experiments, we validate the performance of our model on the context-independent (single-round) conversation task setting in which each sample is a query-response (q-r) pair. We utilize the benchmark dataset (Shang *et al.*, 2015), which collects about 4 millions q-r pairs from a popular Chinese social network website, Weibo.[1] For both testing set and validation set, we randomly select 900 queries, and then select randomly 5 responses under each query, thus both our testing set and validation set consist of 4.5k samples. Sentences are tokenized into word sequences with the Jieba word segmentation tool.[2] The vocabulary consists of the top 50k tokens (a mixture of Chinese words and characters), covering 99.98% words in this corpus, and all the out-of-vocabulary words are replaced with ⟨UNK⟩.

4.2 Implementation Details

We implement the query and response encoder with a one-layer bi-directional GRU, and the decoder with a one-layer GRU and attention mechanism (Bahdanau *et al.*, 2015). We apply the idea of variantial autoencoder (Kingma and Welling, 2014; Zhao *et al.*, 2017) into our model: before the MLP, we use the neural network to estimate the distribution of response vector, sample the vector by reparameterization, then feed it into MLP. Parameters of the query encoder and response encoder are not shared; the two MLP components in M-Seq2Seq branch and CAE branch also do not share parameters. The dimension of all hidden vectors and embeddings are 620 and the batch size is 64. We employ the Adam optimizer (Kingma and Ba, 2014) with the initial learning rate 0.0001 and gradient clipping 5. For generation, we apply a beam search with the size of 10. The memory size K is 1000, and the loss factor λ is 0.1. We implement our model on PyTorch. The implementation details can be found in our codes [3].

4.3 Baselines

We compare two versions of our proposed memory-augmented generative model (MemGM), i.e. MemGM with SoftRead (**MemGM-S**) and MemGM with HardRead (**MemGM-H**), with the following methods:

1. **Seq2Seq**. The standard Seq2Seq with the attention mechanism (Bahdanau *et al.*, 2015) in the decoder and the beam search during generation.

2. **MMI** (Li *et al.*, 2016b). We implement the MMI-bidi model that re-ranks the candidate responses by the maximum mutual information (MMI) criterion in the beam search to promote response diversity.

3. **CVAE**. The conditional variational autoencoders applied in conversation systems (Zhao *et al.*, 2017). We follow the their implementation and adapt it in our single-round conversation setting.

4. **EditRetrieve** (Wu *et al.*, 2018). The state-of-the-art retrieval-augmented generative model, which uses the information of the top-1 retrieved response to guide the response generation.

4.4 Evaluation Metric

Following previous work on response generation (Li *et al.*, 2016b; Yao *et al.*, 2017), we evaluate all competing methods by both automatic metrics and human evaluations. The automatic metrics are::

1. *Bleu 1-4*. *Bleu N* (Papineni *et al.*, 2002) measures the *N*-gram matching between generated responses with the ground-truth responses.

[1] www.weibo.com

[2] github.com/fxsjy/jieba

[3] github.com/tianzhiliang/MemoryAugDialog

	Automatic Metrics				Human Annotate	
	Bleu1,Bleu2,Bleu3,Bleu4	Sim-A, Sim-M	Dist1,Dist2	Entropy	Quality	Info
Seq2Seq	39.98 14.68 6.452 3.227	0.291 0.911	0.043 0.153	7.609	2.33	1.71
MMI	40.08 14.71 6.467 3.236	0.288 0.910	0.053 0.183	**7.724**	2.39	1.63
CVAE	39.85 14.80 6.318 3.012	0.294 0.919	0.044 0.156	7.569	2.42	1.71
EditRetrieve	37.67 10.73 3.437 1.111	0.294 0.932	0.057 0.187	7.586	2.41	1.72
MemGM-S	41.29 15.94 8.084 **4.911**	**0.303 0.936**	0.059 0.214	7.576	2.49	1.70
MemGM-H	**41.40 16.06 8.289** 4.872	0.300 0.935	**0.062 0.218**	7.684	**2.56**	**1.75**

Table 1: The overall performance for all competing methods on quality, relevance, diversity and informativeness.

2. *Sim-A, Sim-M*. They measure the relevance between the query and its response by their word embedding cosine similarity. *Sim-A* is the similarity between two sentence-level embeddings composed by averaging all word embeddings, while *Sim-M* is maximal word-word similarity among all the words of two sentences as (Liu *et al.*, 2016).

3. *Dist 1-2*. Distinct-1 and Distinct-2 (Li *et al.*, 2016b) are the metrics to evaluate the diversity of generated responses, which count the percentage of unique unigrams and bigrams among all test responses.

4. *Entropy*. It measures the informativeness of generated responses proposed by (Mou *et al.*, 2016), which is computed by averaging over all the character-level entropy within responses.

For human evaluations, we hire five annotators from a commercial annotation company to annotate 250 randomly selected test samples. Responses generated by different models are shuffled for each annotator. The annotators evaluate these samples on two aspects: the overall quality (*Quality*) and the informativeness (*Info*). We conduct a 5-scale rating on *Quality*: 1 point for a response irrelevant to the query, 3 points for a valid but meaningless response, 5 points for a coherent and appropriate response without typos. Points of 2 and 4 are for decision dilemmas. We also conduct a 3-scale rating on *Info*: 1 point for the universal response or the response containing no more than three unique words, 2 points for a normal response of a single clause or a single topic, and 3 points for an informative response including at least two clauses of different topics, which transfer the current conversation to another scenario (For example, the query is "How's the weather?"; and response "It's fine today, let's play basketball" transfers the weather topic to sports, which should be marked as 3 points).

5 Experimental Results and Analysis

5.1 Overall Performance

We report both the automatic metrics and human evaluation results of MemGM compared with other methods in Table 1. MMI scores higher than Seq2Seq on *Dist-1&2* owing to its re-ranking mechanism to promote the response diversity. CVAE has a similar performance to the Seq2Seq model. EditRetrieve outperforms Seq2Seq, MMI and CVAE on most metrics. But EditRetrieve underperforms on *Bleu* scores since retrieval models do not learn a query-to-response mapping and their ability of matching with the ground-truth is naturally lower.

MemGM gets the highest scores under most metrics, indicating that our model outperforms current methods on quality, relevance, diversity and informativeness. The improvement of MemGM-H's *Bleu-3&4* (+28.2% and +48.6% in comparison with Seq2Seq) indicates the memory module can extract and memorize trigram and 4-gram response patterns to enhance the generated responses.

For the two versions of our models, HardRead outperforms SoftRead on most metrics. This phenomenon indicates that fetching a single top memorized piece of information would be more helpful than fetching a mixture of multiple memory slots with multiple topics for generative models. Thus, in MemGM, HardRead is the proper mode for reading memory.

5.2 Impact of Memory Size

To investigate how the memory capacity influences the performance of MemGM, we carry out experiments on MemGM-H with a various memory size K and show the results in Table 2 (omitting *Bleu-3&4* and *Sim-M* due to limited space).

In Table 2, the extreme setting $K = |\mathcal{D}|$ works similarly to retrieval-augmented generative models since it saves all q-r pairs separately and utilize

them individually. The difference between them is that $K = |\mathcal{D}|$ reads the memory based on the simple similarity between two vectors e_q and k_i instead of searching the corpus via mature information retrieval technique, which is usually an ensemble of several text matching methods including similarity of query embeddings. That makes the query matching of $K = |\mathcal{D}|$ less accurate and unstable, thus its relevance and quality are weaker than EditRetrieval (Table 1) but does better on diversity. We treat $K = |\mathcal{D}|$ and other results in Table 2 as the comparison between individual and grouped q-r pairs on memory-augmented framework.

	Bleu1,Bleu2	Sim-A	Dist1,Dist2	Entropy
K=10	42.06 15.11	**0.301**	0.042 0.140	7.478
K=100	**42.75** 15.42	0.300	0.043 0.141	7.526
K=1k	41.40 **16.06**	0.300	0.062 0.218	7.684
K=10k	41.22 15.37	0.296	0.057 0.200	7.659
K=$\|\mathcal{D}\|$	34.89 9.764	0.265	**0.094 0.412**	**9.241**

Table 2: The performance of MemGM-H with different memory size K, where $K = |\mathcal{D}|$ means the extreme setting that each sample occupies a memory slot.

MemGM with a large memory size ($K \geq$10k) performs poorly on the response quality (*Bleu-1&2*) and the relevance (*Sim-A*) compared with the MemGMs with small memory. Too large of the memory size leads to too small of the sample size under each memory slot, which increases the instability and lower the quality of each memory slot. Especially, the performance of $K = |\mathcal{D}|$ illustrates the individual retrieval results are not reliable and usually lead to irrelevant results as the observation we will discuss in Sec 5.4.

However, large memory MemGMs ($K \geq$1k) performs well on diversity (*Dist-1&2*) and informativeness (*Entropy*), since the training corpus is partitioned into more memory slots and each slots contains more specific topics. And small memory MemGMs ($K \leq$10) result in low response diversity and informativeness. In conclusion, K=1k is the appropriate memory size to balance all aspects.

5.3 Contents of Memory

Our model is expected to cluster similar queries together to leverage the information of their responses. In addition, each memory slot should own a group of closely related queries. To verify the quality of the memory slots, we pick up the queries from the same memory slot and check the similarity between these queries.

	General	Topic Related	Entity Overlap	Total
$\|\boldsymbol{m}\|$	$\geq$ 1,000	350$\sim$1,000	$\leq$350	–
Cluster #	14	205	781	1,000
Query #	18,291	114,454	81,003	213,748
Query %	8.4%	54.4%	37.2%	100%

Table 3: The statistics on the size of memory slots ($|\boldsymbol{m}|$), cluster number (*Cluster #*), query number (*Query #*), and query proportion over all queries (*Query %*) for the three memory slot types.

	5 Queries under this Memory Slot
Case1: Topic Related Memory Slot	昨天在吉他店里，我们合作了一曲《猜谜到老》 (We play the "guess forever" in guitar store yesterday) 快来听我唱的"至少还有你"。 (Listen to the "at least I have you" sung by me!) 天空之城吉他独奏，最好听的一个版本 ("Castle in the Sky"guitar solo, the best version to hear) 艾薇儿出道十年12首风靡全球的单曲超赞 (12 world-renowned songs since Avril debuted) 夕阳醉了，太好听了。 ("the Setting Sun is Drunk". pleasant to hear.)
Case2: Entity Overlap Memory Slot	十年前的米兰，AC米兰 (**Milan** 10 years ago, AC Milan) 摩纳哥600万打包报价沙拉维+博阿滕—-米兰体育 (Monaco offers 600 millions$ for Shaarawy and Boateng—-**Milan** Sprots.) 全场比赛结束，乌迪内斯2 - 1 米兰 (The whole match was over, Udinese 2:1 **Milan**) 又是一件卡卡米兰时期的队服 (Another Kaka's team uniform when he was in **Milan**) PPTV这俩解说一直在黑米兰啊 (The two PPTV's commentators depreciated **Milan**)

Table 4: Five randomly selected queries under each example of memory slots.

We find that the status of memory slots are different under the different size of memory slot $|\boldsymbol{m}|$, where $|\boldsymbol{m}|$ means how many queries are memorized in this memory slot. We divide the memory slots into three types by their size $|\boldsymbol{m}|$ and show their statistics information in Table 3.

Topic-related Memory Slot (with size $350 < |\boldsymbol{m}| < 1000$ roughly) has a clear topic and the topics of its queries are highly related. For example, the queries under the memory slot shown in Case1 (Table 4) are related to the music topic. There are 205 such slots covering 54.4% queries, which can supply helpful information within the same topic for response generation.

Entity-overlap Memory Slot ($|\boldsymbol{m}| \leq 350$ roughly) has more specific topics and its queries usually share common entities. As shown in Case2 in Table 4, the cluster of queries talk about various football news related to "Milan". 781 out of 1000 slots are of this type and they cover 37.2% queries. In response generation, when the query has the same or similar entities with the memory

slot, it can read the information from that memory slot, which summarizes a group of utterances closely related to that entity.

General Memory Slot ($|\boldsymbol{m}| \geq 1000$) means the memory slot owning too many queries to have a clear topic, whose clustered queries have various topics and are not similar to each other. Fortunately, there are only 14 such slots influencing 8.4% queries.

In summary, most of the memory slots are of good quality and store useful information as we expect, which cover 91.6% of the queries.

5.4 Case Study

In this section, we first compare the cases from MemGM and EditRetrieve to analyze how MemGM exceeds the retrieval-augmented model. Then, we show two examples over all the methods to reveal the characteristics of different methods.

For the comparison between MemGM and EditRetrieve, we analyze the good/bad cases where MemGM-H outperforms/underperforms EditRetrieve and investigate the reasons.

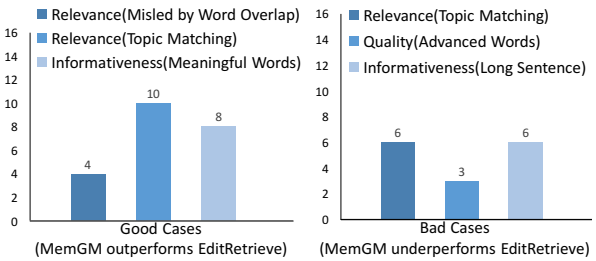

Figure 3: The reasons that MemGM outperforms/underperforms EditRetrieve on human annotated cases.

We collect the good/bad cases from human annotation results by this criterion: If more than four annotators marked the MemGM-H's *Quality* score higher/lower than EditRetrieve's by 2 points, this sample is the good/bad case. From all annotated samples, we obtain 22 good cases and 15 bad cases, and summarize the reasons in Figure 3.

There are 3 reasons for the good cases where MemGM outperforms EditRetrieve, shown in the left side of Figure 3. Firstly, we observe a phenomenon from EditRetrieve's results that the EditRetrieve's response has the word overlap with its query but is not related to the query at semantic level. The reason is that retrieval systems are highly reliant on the word matching, so they may retrieve fake results with high lexical similarity but indeed low relevance. Therefore, that phenomenon is due to "misled by word overlap" and it leads to EditRetrieve's irrelevant results on 4 cases where MemGM performs well. It indicates the retrieval quality limits the performance of retrieval-augmented models. Secondly, EditRetrieve's results mismatch the topics of given queries in 10 cases, where MemGM can generate relevant responses. Thirdly, MemGM outperforms EditRetrieve in 8 cases due to containing more meaningful words in MemGM's responses, where meaningful words means the notional words carrying the specific topic information.

To summarize the good cases, the major advantage of MemGM is the high relevance with queries compared with EditRetrieve. The performance on *Sim-A* and *Sim-M* (Table 1) verifies that MemGM exceeds EditRetrieve on relevance. Note that the retrieval-augmented models are sensitive to the quality of retrieved results; if retrieved query-response pairs are irrelevant to the query, utilizing such information would lead to topic drift and the generation of irrelevant responses.

There are 3 reasons for MemGM underperforming EditRetrieve (Figure 3 right). Firstly, MemGM also suffers from irrelevant responses due to mismatching the topic of queries. However, in terms of the relevance, the number of the MemGM's bad cases (6 cases) is much fewer than its good cases' (14 cases). Secondly, 3 cases of EditRetrieve are of higher quality owing to the suitable use of advanced and low-frequency words. Thirdly, 6 responses from EditRetrieve are longer and carry more information.

Two examples in Table 5 show the model performance. In the first case, EditRetrieve is misled by "ribs"; it treats "ribs of human" as "pork ribs" and responds an utterance related to "food" topic. MemGMs understand the topic and generate informative responses. In the second case, Seq2Seq, CVAE and MMI only respond by simply repeating the query. EditRetrieve can bring in a new word "rest" related to the query. MemGMs capture the "cold" from the given query and respond utterances related to both "cold" and "health".

6 Conclusion

In this paper, we propose a novel memory-augmented generative model for response generation. It leverages groups of query-response pairs to augment generative models instead of the individ-

Query	多亏我是个瘦子，伤心时我可以数数排骨 (Thanks to being a **thin** man, I can count my **ribs** during my sad time.)
Seq2Seq	我也是这么想的。(I do think so.)
MMI	我也曾伤心过(I have experienced the sad time)
CVAE	哈哈，我也是这样的。(Ha-ha, me to.)
EditRetrieve	我也想吃。。。(I want to **eat** it too...)
MemGM-S	伤心的时候会想起谁 (Who do you remember during sad time.)
MemGM-H	我可以说我是个胖子吗? (Can I say I am a **fat** man?)
Query	今天出来好冷，冻脸啊！大家要注意身体 (It's **cold** today when leaving, freezing my face.Attention to health!)
Seq2Seq	注意身体哦，注意身体 (Attention to health. attention to health.)
MMI	注意身体。。。。(Attention to health....)
CVAE	注意身体哦，注意身体 (Attention to health! attention to health.)
EditRetrieve	注意身体，好好休息。 (Attention to health, have a good **rest**.)
MemGM-S	好冷啊。注意保暖。 (It's too **cold**, pay attention to keep warm.)
MemGM-H	好冷啊！注意身体！ (It's too **cold**, attention to health.)

Table 5: The case study among all competing methods.

ual retrieval results. It clusters the training corpus into multiple groups, extracts and memorizes common characteristics of each group for generation. Compared with retrieval-augmented models, our model makes improvements on the quality, relevance and informativeness of response generation.

7 Acknowledgments

We would like to acknowledge Zhourong Chen and Xingjian Shi for their constructive advices and the anonymous reviewers for their insightful comments. Research on this article was supported by Hong Kong Research Grants Council under grants 16202118 and 16212516.

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

Deng Cai, Yan Wang, Victoria Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. Skeleton-to-response: Dialogue generation guided by retrieval memory. *arXiv preprint arXiv:1809.05296*, 2018.

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *Transactions of the Association of Computational Linguistics*, 6:437–450, 2018.

Zongcheng Ji, Zhengdong Lu, and Hang Li. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*, 2014.

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

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *stat*, 1050:10, 2014.

Hung Le, Truyen Tran, Thin Nguyen, and Svetha Venkatesh. Variational memory encoder-decoder. In *Advances in Neural Information Processing Systems*, pages 1515–1525, 2018.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *NAACL-HLT*, pages 110–119, 2016.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, 2016.

Feng-Lin Li, Minghui Qiu, Haiqing Chen, Xiongwei Wang, Xing Gao, Jun Huang, Juwei Ren, Zhongzhou Zhao, Weipeng Zhao, Lei Wang, et al. Alime assist: an intelligent assistant for creating an innovative e-commerce experience. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 2495–2498. ACM, 2017.

Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023*, 2016.

Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1468–1478, 2018.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. *arXiv preprint arXiv:1607.00970*, 2016.

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

Prasanna Parthasarathi and Joelle Pineau. Extending neural generative conversational model using external knowledge sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 690–695, 2018.

Alan Ritter, Colin Cherry, and William B Dolan. Data-driven response generation in social media. In *EMNLP*, pages 583–593, 2011.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, pages 3776–3783, 2016.

Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *ACL-IJCNLP*, pages 1577–1586, 2015.

Yiping Song, Cheng-Te Li, Jian-Yun Nie, Ming Zhang, Dongyan Zhao, and Rui Yan. An ensemble of retrieval-based and generation-based human-computer conversation systems. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4382–4388. AAAI Press, 2018.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

Zhiliang Tian, Rui Yan, Lili Mou, Yiping Song, Yansong Feng, and Dongyan Zhao. How to make context more useful? an empirical study on context-aware neural conversational models. *Annual Meeting of the Association for Computational Linguistics*, 2:231–236, 2017.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

Yu Wu, Furu Wei, Shaohan Huang, Zhoujun Li, and Ming Zhou. Response generation by context-aware prototype editing. *arXiv preprint arXiv:1806.07042*, 2018.

Chien-Sheng Wu, Richard Socher, and Caiming Xiong. Global-to-local memory pointer networks for task-oriented dialogue. *arXiv preprint arXiv:1901.04713*, 2019.

Rui Yan, Yiping Song, and Hua Wu. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *SIGIR*, pages 55–64, 2016.

Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan. Towards implicit content-introducing for generative short-text conversation systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2190–2199, 2017.

Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. Augmenting end-to-end dialogue systems with commonsense knowledge. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 654–664, 2017.

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. Multi-view response selection for human-computer conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 372–381, 2016.

Yimeng Zhuang, Xianliang Wang, Han Zhang, Jinghui Xie, and Xuan Zhu. An ensemble approach to conversation generation. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 51–62. Springer, 2017.