Empathetic and Emotionally Positive Conversation Systems with an Emotion-specific Query-Response Memory

Zhiliang Tian[1], **Yinliang Wang**[2], **Yiping Song**[3]*, **Chi Zhang**[2],
Dongkyu Lee[2], **Yingxiu Zhao**[2], **Dongsheng Li**[1], **Nevin L. Zhang**[2]
[1] College of Computer, National University of Defense Technology,
[2] Department of CSE, The Hong Kong University of Science and Technology,
[3] College of Science, National University of Defense Technology
{tianzhilianghit,yinliang.wang1024}@gmail.com, songyiping@nudt.edu.cn,
{czhangbt,dleear,yzhaocx,lzhang}@cse.ust.hk, lds1201@163.com

Abstract

Emotional conversation systems generate responses for the input queries considering the speaker's emotions in a conversation. Existing emotional conversation systems output emotional responses according to either a given emotion or the user's emotion reflected in the input queries. Following a given emotion may lead to an emotional drift between the given emotion and the conversation state, and following only the user's emotion may aggravate the user's negative feelings if users suffer from a negative mood. In this paper, we propose to generate empathetic responses catering to the user's emotions while leading the conversation to be emotionally positive. Particularly, by abstracting the conversation corpus, we extract and store the different responding strategies for different users' emotions and conversational topics into a memory. We encourage positive emotions in conversation via a sentiment evaluator. We model the memory outputs with a Gaussian mixture distribution and sample a final responding strategy from the distribution. The strategy acts as a condition to a transformer model to generate responses. The experiments verify our model surpasses the baseline methods in appropriateness, diversity, and generating emotionally positive responses.

1 Introduction

Most conversation models capture only the correlation between queries and responses and may overlook speaker's emotional states in the conversation. Emotional conversation models (Zhou et al., 2018; Song et al., 2019) consider speakers' emotions during the dialogues, where the speakers can be either users or chatbots. Those models make chatbots aware of the user's emotions and enable them to respond empathetically. There are two directions for emotional conversation models. (1) Controllable response generation enables chatbots to respond conditioned on a certain emotion (Shin et al., 2020; Liu et al., 2021) or style (Zhou and Wang, 2018; Dathathri et al., 2019). Those methods require an explicit emotion label as input and the label dominates the chatbot's response. (2) Empathetic response generation, detects the user's emotion from the query so that the chatbots can respond empathetically considering the user's emotion (Lin et al., 2019, 2020).

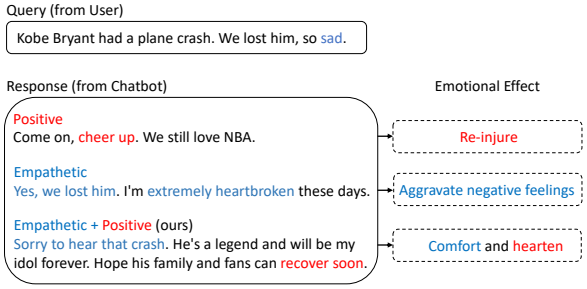

Figure 1: An example of the differences among controllable response generation (in red), empathetic response generation (in blue), and our chatbot (in a mixture of red and blue). When responding to the sad news, blunt cheering up might lead to reinjury and a plain show of empathy tends to aggravate negative feelings. A suitable response should be both empathetic and positive.

The above two directions cater to emotions in conversation but still possess some weaknesses. For the first direction, if the given emotion label mismatches with the user's emotion, controllable responses generation leads to an emotional drift (Deng et al., 2020) among dialogues, that is, the emotions of query and response are inconsistent and incoherent. As the example shown in Fig. 1, if the user tells a sad story and the chatbot aggressively encourages the emotion to be happy, the chatbot's responses may be crude resulting in reinjury to the user. As for empathetic response generation, without acquiring any additional inductive bias, the user's emotion does not necessarily hint how to respond empathetically (Shin et al., 2020), so considering the user's emotion usually pushes chatbots

*Corresponding Author

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6364–6376
December 7-11, 2022

to imitate the user's emotion. If the user is sad (Fig. 1), simply imitating the user's emotion may aggravate the user's negative feelings.

Existing research hardly explores to balance empathy and positive emotions in response generation. To achieve that goal, we should adopt different responding strategies to different situations and finally lead positive conversations. For example, to respond to a sad story, we should first show our empathy and then gently lead a positive emotion; for a happy utterance, we can directly congratulate the user and even share other good news.

In this paper, we propose a conversation model that generates empathetic responses and gently guides the mood of the conversation towards a positive direction. To customize responding strategies to different situations, we extract implicit responding strategies for specific emotions and topics by abstracting the training corpus. We store the extracted strategies into an emotion-specific query-response memory to assist the response generation. To lead positive emotions, we employ a sentiment evaluator to encourage positive responses. The above operations constrain the strategies and emotions in generation, thus they naturally decrease the diversity of the generated responses. To encourage diverse responses and fully utilize the memory outputs, inspired by conditional variational autoencoder (CVAE) (Sohn et al., 2015), we model the memory outputs with a Gaussian mixture distribution and mix the memory outputs (i.e. responding strategies) by sampling a strategy vector z from the distribution. Finally, the strategy vector z controls a conventional conversation model to generate responses. Our experiments verify that our model not only exceeds the comparing methods in quality and diversify but also encourages positive conversation. Our contributions are threefold: (1) We propose a conversation model to balance users' emotions and positive emotions. (2) We propose to abstract the corpus to build an emotion-specific query-response memory to carry responding strategies for different emotions and topics and propose a strategy mixer to fuse memory outputs. (3) Our model surpasses some latest emotional chatbots in appropriateness, diversity, and positivity of responses.

2 Related Work

2.1 Conversation Systems

Large scale corpora lead to a great success in data-driven conversation systems, including retrieval-based (Isbell et al., 2000; Ji et al., 2014) and generation-based methods (Shang et al., 2015). Generation-based methods achieve the end-to-end response generation. To prompt conversation systems, Serban et al. (2016) propose to consider the historical utterances. Li et al. (2016) diversify the generated response. Xing et al. (2017) strengthen the topic coherence among conversational utterances. Conversation systems also consider speakers' states, including speaker's personalities (Zhang et al., 2018) and roles (Hu et al., 2019).

2.2 Emotional Conversation Systems

Emotional conversation systems can be divided into two directions: catering to the user's emotions and expressing the chatbot's emotions. The first direction aims to capture user's emotions and make empathetic responses (suitable responses cottoning with the user's states) (Lin et al., 2019, 2020; Li et al., 2020; Lin et al., 2020; Gao et al., 2021; Li et al., 2022). Empathy refers to the capacity to respond with an appropriate emotion to another's mental states (Zhong et al., 2020). Rashkin et al. (2019) design a pipeline system, where a classifier predicts emotion words describing the user's query. Lin et al. (2019) propose an end-to-end neural conversation model that detects the emotion by an encoder and generates responses with decoders. Cao et al. (2020); Zheng et al. (2021) apply GPT (Generative pre-training) (Radford et al., 2018) to empathetic chatbots, and Zhong et al. (2020) employ BERT (Devlin et al., 2019) to emotional response generation. Liu et al. (2021) construct a dataset to simulate the dialogs between psychologist and help-seeker. Those methods mainly cater to users' emotions without planning emotions in future conversations, which is different from our model.

The second direction aims to enable the chatbot to respond conditioned on a given emotion, which is a sub-domain of controllable text generation (Lubis et al., 2018a,b; Dathathri et al., 2019; Colombo et al., 2019; Xu et al., 2021). Lubis et al. (2018a) achieves it with a HRED (hierarchical recurrent encoder-decoder) (Serban et al., 2016). Song et al. (2019) equip a Sequence-to-Sequence (Seq2Seq) (Sutskever et al., 2014) with lexicon-based attention and encourage the model to express emotion implicitly. Shin et al. (2020) leads the positive emotion in conversation via reinforcement learning. ECM (emotional chatting machine) (Zhou et al.,

2018) embeds the given emotion, models the emotion expression, and generates emotional words with external memory. Inspired by dual learning, Shen and Feng (2020) enable two chatbots to learn by chatting with each other under the specific emotion. Jiang et al. (2021) lead a happy conversation ending in multi-turn conversations. The conversation systems that explicitly consider the two directions mentioned above are underexplored, which is the goal of this paper.

2.3 Variational Autoencoder

Kingma and Welling (2013) propose variational autoencoder (VAE) to reconstruct a sample x through a latent variable $\boldsymbol{z}$. VAE estimates the intractable posterior $P(\boldsymbol{z}|x)$ with a recognition network. Sohn et al. (2015) extend VAE to conditional VAE (CVAE). CVAE estimates the latent $\boldsymbol{z}$ with a condition on the prior and posterior distribution. Zhao et al. (2017) apply CVAE to conversation to diversify the generated utterances, where queries act as the conditions and responses act as the x. Gao et al. (2019b) equip CVAE with interpretable latent variables. The differences between those methods and our models are: 1) the estimation of our posterior relies on the memory instead of the responses x in CVAE. 2) our posterior follows the Gaussian mixture to carry a mixture of potential responses.

3 Approach

3.1 Model Architecture

Our model understands the user's query, determines a responding strategy, and generates a response conditioned on the strategy. Our model has five modules (Fig. 2) as follows,

- *Encoder* represents the query q with a vector $\boldsymbol{e}$.
- *Emotion Detector (ED)* detects the emotion of the query q.
- *Responding Strategy Generator (RSG)*, the core module in this paper, generates a responding strategy vector $\boldsymbol{z}$ to guide the generation. *RSG* contains an emotion-specific query-response memory and a strategy mixer to determine a suitable responding strategy.
- *Conditional Conversation Model (CCM)* is a transformer-based conversation model generating a response for the given query q with the strategy vector $\boldsymbol{z}$.
- *Pre-trained Sentiment Evaluator (PSE)* evaluates the sentiment of generated responses and provides feedback to the above modules.

Our training procedure consists of two phases. The first phase (pre-training phase) pre-trains *ED*, *CCM*, and *PSE* separately. The second phase is to learn *Encoder* and *RSG* while freezing *PSE* and *CCM*. *Encoder* provides the query representation for *RSG*, *ED* detects query emotion for *RSG*, and then *RSG* generates a responding strategy vector to guide *CCM* to generate responses. *PSE* and *CCM* provide the feedback to supervise the training of *Encoder* and *RSG*.

3.2 Encoder

To represent the user's query, we employ a GRU-based encoder to embed the query into a vector. The *Encoder* consists of a word embedding layer, a GRU (Cho et al., 2014) layer, and a multilayer perceptron. The word embedding layer projects each query's word into a vector. Then, the GRU receives the sequence of word embeddings and outputs its last hidden state. The multilayer perceptron is composed of two fully connected layers with a non-linear function; the multilayer perceptron transfers the GRU's last hidden state to another vector, denoted as $\boldsymbol{e}$. $\boldsymbol{e}$ is the output of *Encoder* carrying the query's information. *Encoder* is involved in the two training phases. We introduce more details about the training in Sec. 3.7.

3.3 Emotion Detector (ED)

To explicitly represent the emotions of the user's query, we pre-train an emotion detector *ED* to classify the query's emotion. The output is an explicit emotion category (e.g. happy). As the usage of conventional text classification tasks in BERT (Devlin et al., 2019), we initialize *ED*'s parameters with a pre-trained BERT model, and then fine-tune *ED* over an emotion classification dataset, which has 7 emotion categories. This module is only trained in the first training phase and provides emotion categories to *RSG* in the second phase.

3.4 Responding Strategy Generator (RSG)

Responding strategy generator *(RSG)* aims to bridge the user's query with the responding strategies. Given the query vector $\boldsymbol{e}$ and query's emotion label, *RSG* generates a strategy vector $\boldsymbol{z}$ to guide the chatbot to respond suitable considering the user's emotion and lead to positive emotions.

The *RSG* is only trained in the second training phase and has two sub-modules: 1. *Emotion-specific Query-Response Memory* receives the query representation $\boldsymbol{e}$ from *Encoder* and outputs

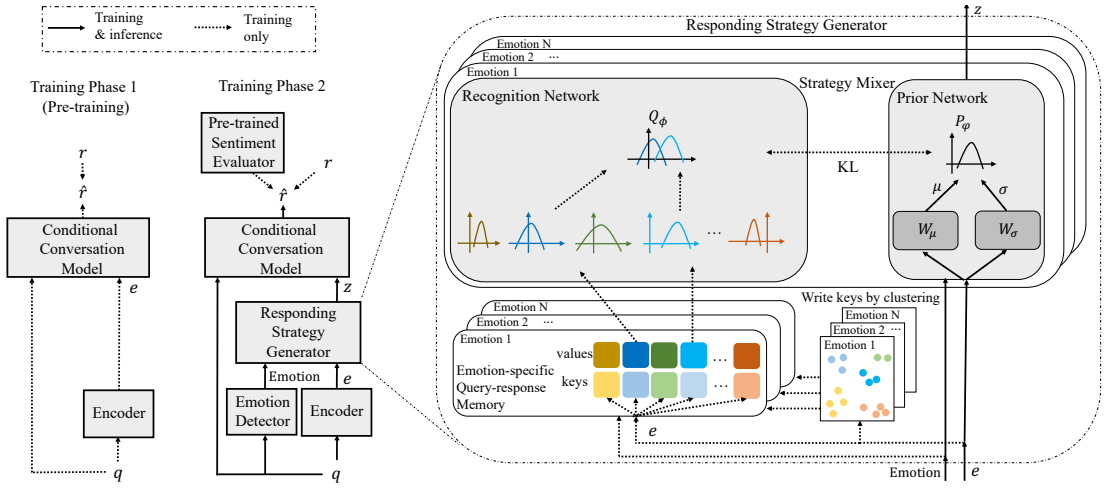

Figure 2: Overview of our model. The solid arrows indicate the operations for both training and inference while the dashed arrows mean the operations only for training. The left side shows the first and second training phases; the right side shows our core module, *Responding Strategy Generator*, which works in the second phase. Colors represent various groups of similar samples. q, r, $\hat{r}$ denotes query, ground-truth response, and generated response.

several vectors $\boldsymbol{v}$ carrying information about potential responses; 2. *Strategy Mixer* mixes the memory output $\boldsymbol{v}$ to obtain a strategy vector $\boldsymbol{z}$ for chatbots.

3.4.1 Emotion-specific Query-Response Memory

The module is designed to carry different implicit responding strategies for different user's emotions. This module consists of several key-value memories and each memory corresponds to a specific emotion. Each emotion-specific memory has K slots, which is a key-value vector pair $\langle \boldsymbol{k}_i, \boldsymbol{v}_i \rangle$. The key vector $\boldsymbol{k}_i$ represents a group of similar queries; the value $\boldsymbol{v}_i$ carries the information of a group of potential responses. The key-value slot memorizes a mapping from queries to the information of how to respond for the specific emotion and topic. The memory module is constructed by memory write and contributes to other modules by memory read.

- **Memory Read:** The input of memory read is the user's query representations $\boldsymbol{e}$ from *Encoder* and the detected query emotion from *ED*. We first locate the emotion-specific memory according to the detected emotion. Then, in the emotion-specific memory, the read operation search for memory slots by the similarity between query vector $\boldsymbol{e}$ and every memorized key vector $\boldsymbol{k}_i$, in which we measure the similarity via dot-product.

  Memory read fetches the value with two options: **Hard Read** is to fetch the value vector whose key vector is most similar to the query vector $\boldsymbol{e}$, to use the most suitable response information. **Soft Read** fetches the several most similar value vectors and their similarity scores, which means reading several potentially useful information. The output of memory read acts as the input of *Strategy Mixer*.

- **Memory Write:** It constructs the emotion-specific memory by clustering over all the training samples under the same emotion. We collect the query emotion detected by *ED* for all training samples and gather samples with the same emotion to the same subset. For each subset, we construct an emotion-specific memory: we collect the query vector $\boldsymbol{e}$ of all samples and conduct K class K-means clustering over the vectors $\boldsymbol{e}$. The center of i-th cluster serves as the key vector $\boldsymbol{k}_i$ for i-th memory slot.

  The K-means clustering (determining the keys $\boldsymbol{k}$) and the training of other modules (including *Encoder*, values $\boldsymbol{v}$, and *Strategy Mixer*) are conducted separately and iteratively. For each training epoch, we first train our model except the keys $\boldsymbol{k}$ via gradient descent by freezing $\boldsymbol{k}$. Once the training of this epoch finishes, we conduct K-means clustering. The cluster centers serve as the memory keys $\boldsymbol{k}$ for the next training epoch. In this way, memory keys and other modules are trained iteratively.

  The memory values $\boldsymbol{v}$ are the learnable parameters trained via gradient descent. After each clustering, we randomly initialize $\boldsymbol{v}$ and then train $\boldsymbol{v}$ together with other modules (*Encoder* and *Strategy Mixer*).

In this way, each key vector gathers similar user's queries and memorizes their representative information (i.e. cluster centers). Each value vector

$\boldsymbol{v}_i$ learns to extract the common responding characteristics for a group of users, the queries in a similar topic and a same emotion. Hence, the $\langle \boldsymbol{k}_i, \boldsymbol{v}_i \rangle$ pair memorizes the mapping from the user's query to the responding strategies for a specific emotion, thus our model can generate suitable responses catering to user's specific emotion.

3.4.2 Strategy Mixer

To fully utilize the memory outputs and encourage the diversity, inspired by CVAE, we propose to mix responding strategies (i.e. memory outputs) with the query and obtain the final strategy vector $\boldsymbol{z}$ by sampling. Like our memory module, each emotion has a strategy mixer. All the strategy mixers share the same structure and have their own parameters. For each sample, the model first locates the specific strategy mixer according to the emotion from *ED* and then uses the located mixer. In the following parts, we introduce the usage of one strategy mixer.

The only difference between our model and the vanilla CVAE is that 1. our posterior $Q_\phi(\boldsymbol{z}|M,q)$ accesses the memory M instead of r; 2. our memory may output one or multiple vectors instead of only one input r in CVAE. The reasonability of using memory M is that M carries the mapping from the user's queries to the potential responses and the memory output is the information about responses, thus the model can infer the response r by reading the memory. We use multiple vectors from memory M by constraining the vectors to follow a same class of distribution.

In Eq. 1, $Q_\phi(\boldsymbol{z}|M,q)$ is the approximate posterior to estimate the true posterior $P(\boldsymbol{z}|r,q)$ while $P_\varphi(\boldsymbol{z}|q)$ acts as the prior. The query q acts as the condition and the response r is the model output (see the deductions in Appendix D).

$$\begin{aligned}\mathcal{L}_{\text{SM}} &= \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log P_\theta(r|\boldsymbol{z},q)] \\ &- KL[Q_\phi(\boldsymbol{z}|M,q)||P_\varphi(\boldsymbol{z}|q)] \le \log P(r|q)\end{aligned} \tag{1}$$

Recognition network and prior network model the posterior $Q_\phi(\boldsymbol{z}|M,q)$ and prior $P_\varphi(\boldsymbol{z}|q)$, respectively. The decoder $P_\theta(r|\boldsymbol{z},q)$ is the *CCM* to generate the responses (introduced in Sec. 3.5). The actual input of strategy mixer is the query vector $\boldsymbol{e}$ and memory reading output while its output is the final strategy vector $\boldsymbol{z}$.

- **Recognition Network** models the **posterior** $Q_\phi(\boldsymbol{z}|M,q)$ that generates $\boldsymbol{z}$ by accessing the memory output and the query vector $\boldsymbol{e}$. Since vector $\boldsymbol{z}$ is hard to model from the memory directly, we construct a mixture of Gaussian over the memory. As each memory slot covers several similar samples, we assume each memory slot corresponds to a specific Gaussian distribution and all samples vectors in this slot follow that Gaussian distribution. Like the idea of Gaussian mixture, each query $\boldsymbol{e}$ may be similar to multiple slots and its corresponding vector $\boldsymbol{z}$ may come from a mixture of memory slots. Hence, $\boldsymbol{z}$'s posterior $P_\theta(\boldsymbol{z}|M,q)$ approximately follows a mixture of Gaussian. Particularly, there are K Gaussian distributions and each Gaussian corresponds to a memory slot. For the i-th slot, the value vector $\boldsymbol{v}_i$ acts as the mean of i-th Gaussian distributions. The variance of the Gaussian is a learnable scalar λ_i that times an identity vector $\boldsymbol{I}$ as used in (Yang et al., 2019). Each Gaussian is denoted as $\mathcal{N}(\boldsymbol{v}_i, \lambda_i \boldsymbol{I})$. The posterior $Q_\phi(\boldsymbol{z}|M,q)$ describes memory reading output with a mixture of c Gaussian distributions, which is weighted by the probability of the sample belonging to each Gaussian π_i. Notice that the memory has K slots in total, and memory reading only fetches c ($c \ll N$) at each time. The probability π_i is the normalized similarity between key $\boldsymbol{k}_i$ and $\boldsymbol{e}$.

$$\begin{aligned} Q_\phi(\boldsymbol{z}|M,q) &= \sum_i^c \pi_i \mathcal{N}(\boldsymbol{v}_i, \lambda_i \boldsymbol{I}), \\ \pi_i &= \textit{softmax}(\boldsymbol{k}_i \cdot \boldsymbol{e}) \end{aligned} \tag{2}$$

where c is the number of slots read from the memory. The hard read ($c = 1$) fetches a single slot (distribution); the soft read ($c > 1$) leads to a trade-off between quality and emotional effect.

- **Prior Network** models the prior $P_\varphi(\boldsymbol{z}|q)$ that generates $\boldsymbol{z}$ from query vector $\boldsymbol{e}$ without accessing memory M. As the usage in CVAE, we assume the prior follows Gaussian distribution. We employ two fully connected layers $\boldsymbol{W}_\mu, \boldsymbol{W}_\sigma$ to transfer the query vector $\boldsymbol{e}$ to the mean and variance of the prior distribution $P_\varphi(\boldsymbol{z}|q) = \mathcal{N}(\boldsymbol{W}_\mu \boldsymbol{e}, \boldsymbol{W}_\sigma \boldsymbol{e})$. The motivation for estimating $\boldsymbol{z}$ without accessing the memory is that generating $\boldsymbol{z}$ from the memory cannot be learned by gradient descent since the memory read is not differentiable.

$$\boldsymbol{z} = \sum_i^c \pi_i \boldsymbol{v} + \lambda_i \boldsymbol{I}\boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{1}),\ \text{posterior} \tag{3}$$

$$\boldsymbol{z} = \boldsymbol{W}_\mu \boldsymbol{e} + \boldsymbol{W}_\sigma \boldsymbol{e}\boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{1}),\ \text{prior} \tag{4}$$

We apply the reparameterization trick (Kingma and Welling, 2013) on the recognition (Eq. 3) and prior network (Eq. 4). We optimize the learnable parameters ϕ and φ. ϕ consists of $\boldsymbol{v}$, λ, and *Encoder*'s parameters; φ consists of the $\boldsymbol{W}_\mu$, $\boldsymbol{W}_\sigma$, and *Encoder*'s parameters.

3.5 Conditional Conversation Model (CCM)

Conditional conversation model (CCM) is to generate a response for a given query constrained by a specific condition. We first build a conventional conversation model with transformer (Vaswani et al., 2017) that transfers queries to responses. Based on the transformer, we incorporate a condition vector by appending the vector in front of the sequence of the transformer, the input word embeddings. *CCM* is trained only in the first training phase. In the second phase, *CCM*'s parameters are frozen while *CCM* and *PSE* act as the feedback to supervise *RSG*'s training.

3.6 Pre-trained Sentiment Evaluator (PSE)

Pre-trained sentiment evaluator *(PSE)* evaluates the sentiment of the responses generated by *CCM* and provides the feedback to *CCM*. *PSE* is a BERT-based sentiment classifier. Based on the initial parameters from a pre-trained BERT (Devlin et al., 2019), we fine-tune *PSE* over a sentiment classification dataset. *PSE* is trained in the first phase (pre-training phase) and its training does not involve other modules. After the training, we freeze its parameters and employ *PSE* to provide feedback in the second phase.

3.7 Model Training and Inference

Our training consists of two phases. The first phase, pre-training phase, aims to pre-train *ED*, *PSE*, and *CCM*. *ED* and *PSE* are trained alone on their own datasets. The left branch in Fig. 2 shows *CCM*'s pre-training. We pre-train *CCM* assisted by *Encoder*, where *Encoder*'s output vector $\boldsymbol{e}$ acts as *CCM*'s input condition. The output vectors cover a variety of conditions since the vectors come from various utterances.

The second training phase (middle branch in Fig. 2) trains *Encoder* and *RSG* while freezing *ED*, *PSE*, and *CCM*, since *ED*, *PSE*, and *CCM* is well trained in the first phase. *Encoder* encodes the query and feeds its output $\boldsymbol{e}$ to *RSG*. Then, *RSG* learns to generate the responding strategy vector $\boldsymbol{z}$. *CCM* takes $\boldsymbol{z}$, instead of $\boldsymbol{e}$ in the first phase [1], as its input condition. *CCM* generates the final response $\hat{r}$. *PSE* evaluates the sentiment of $\hat{r}$ and feedbacks to its other modules to optimize *Encoder* and *RSG*. *CCM* are not optimized by *PSE*'s feedback to avoid the complicated back-propagation through *CCM*'s every time steps. The loss function in this phase is a combination of the strategy mixer's loss $\mathcal{L}_{\text{SM}}$ (Eq. 1) and loss of sentiment score from *PSE* $\mathcal{L}_{sent}$. $\mathcal{L}_{\text{SM}}$ leads to appropriate and diverse responses since it imitates the emotion-specific query-responses from the training samples. $\mathcal{L}_{sent}$ encourages the positive conversation.

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{\text{SM}} + \alpha \mathcal{L}_{sent} \\ &= \mathcal{L}_{\text{SM}} - \alpha P(\mathit{PSE}(\hat{r}) = \text{positive}) \end{aligned} \tag{5}$$

where α denotes the weight balancing two losses and *PSE*'s sentiment score is identical to the probability of the generated response being positive.

The inference phase is the same as the second training phase except that the inference omits the recognition network. Solid arrows in Fig. 2 show the inference phase.

4 Experiments

4.1 Experimental Setting

Following (Shang et al., 2015), we use the conversation dataset from weibo.com with 1.25M samples. We show the details about hyper-parameters, codes, and datasets in Appendix. We conduct the experiments on the following methods.

- *Emotion Independent Models*. Seq2Seq (**S2S**) (Shang et al., 2015) and Transformer (**Trs**) (Vaswani et al., 2017) are conventional conversation models without considering the emotions.
- *Emotional Conversation Models*. To encourage the conversation to be positive, **S2S+ECM** follows Seq2Seq-based ECM (Zhou et al., 2018) that learns to generate a response with a given emotion label. We train the ECM on a corpus with emotion labels (positive and negative) and feed the positive label to the conversation model in inference. As transformer is more powerful than Seq2Seq, we implement a transformer-based ECM (Zhou et al., 2018) **Trs+ECM**. **Trs+Dual** (Shen and Feng, 2020) applies dual learning to controllable response generation. **Trs+RL** (Shin et al.,

[1] As the phase 1 is only the pre-training, $\boldsymbol{z}$ and $\boldsymbol{e}$ need not to have an identical distribution, but they are similar.

	Appropriateness			*Diversity*			*Emotion*		
	Bleu1 Bleu2 Bleu3 Bleu4	Nist3 Nist4	Qual	Dist1 Dist2 Dist3 Dist4	Ent	Div	Sent	Posi%	Emo
S2S	6.494 2.483 1.026 0.509	0.333 0.334	1.89	0.002 0.011 0.027 0.049	3.170	1.64	0.648	75.2%	1.70
S2S+ECM	4.971 2.264 1.305 0.902	0.322 0.324	1.76	0.009 0.051 0.110 0.170	4.354	1.37	0.722	78.4%	1.76
Trs	17.799 11.866 9.484 8.259	2.026 2.064	2.73	0.014 **0.207** 0.433 0.590	5.767	2.80	0.680	71.4%	2.45
Trs+ECM	17.278 10.904 8.487 7.279	1.869 1.910	2.85	0.012 0.159 0.350 0.506	5.565	2.76	0.730	78.4%	2.73
Trs+Dual	18.321 12.083 9.484 8.142	1.973 2.005	3.02	0.012 0.195 0.430 0.594	5.543	2.79	0.739	78.6%	2.64
Trs+RL	17.245 10.594 8.058 6.818	1.830 1.860	2.69	0.011 0.169 0.408 0.603	5.567	2.56	0.701	75.1%	2.78
Ours (hard)	**19.715 13.026 10.287 8.859**	**2.122 2.155**	**3.41**	**0.014** 0.206 **0.451 0.608**	**5.884**	**2.96**	0.764	81.3%	2.93
Ours (soft)	19.324 12.755 10.131 8.736	2.083 2.121	3.28	0.013 0.201 0.426 0.587	5.793	2.92	**0.779**	**83.1%**	**3.18**

Table 1: The performance of comparing methods on automatic metrics and human evaluation. The first six rows indicate the performance of baselines. Ours (hard) and Ours (soft) denote our model with hard memory reading and soft memory reading respectively. In automatic metrics, the underline results indicate that our improvements compared with baselines are statistically significant under t-test ($p < 0.05$). For human evaluation, the Fleiss' kappa (Fleiss, 1971) among different annotators is 0.42, which shows a moderate agreement among annotators.

	Appropriateness		*Diversity*		*Emotion*	
	Bleu1 Bleu2 Bleu3 Bleu4	Nist3 Nist4	Dist1 Dist2 Dist3 Dist4	Ent	Sent	Posi%
Ours (hard)	**19.715 13.026 10.287 8.859**	**2.122 2.155**	**0.014 0.206 0.451 0.608**	**5.884**	**0.764**	**81.3%**
Ours − SM	19.576 12.714 10.041 8.627	2.081 2.114	0.012 0.169 0.419 0.576	5.603	0.720	76.9%
Ours − CCM fixing	18.230 12.429 9.945 8.679	2.022 2.091	0.014 0.202 0.438 0.591	5.778	0.749	80.3%
Ours $-\mathcal{L}_{sent}$	19.352 12.586 9.941 8.532	2.055 2.076	0.013 0.196 0.431 0.597	5.743	0.684	75.9%
Ours − Gaussian	19.512 12.481 10.087 8.596	2.084 2.094	0.012 0.198 0.436 0.594	5.739	0.731	79.5%
Ours − EmoMemory	19.543 12.486 10.085 8.693	2.054 2.089	0.013 0.202 0.435 0.596	5.741	0.741	80.1%

Table 2: The performance over our different variants. Ours − SM, Ours $-\mathcal{L}_{sent}$, and Ours − CCM fixing indicates our variants without the strategy mixer (SM), the loss of sentiment $\mathcal{L}_{sent}$, fixing the *CCM*'s parameters in the second training phase, Gaussian distribution in SM, and emotion-specific memory.

2020) uses RL to encourage a transformer to output positive responses.

- *Ours*. **Ours (Hard)** and **Ours (Soft)** denote our proposed model with hard read ($c = 1$ in the recognition network) and soft read ($c = 10$).

We evaluate all methods with both automatic and human evaluations. The automatic evaluations consist of three aspects. First, we evaluate **Appropriateness** in *Bleu-N* (Papineni et al., 2002) and *Nist-N* (Doddington, 2002). Second, we measure **Diversity** with *Dist-N* (Li et al., 2016) and Entropy (*Ent*) (Mou et al., 2016). Third, we measure how positive the generated responses are (**Emotion**). *Sent* is the average sentiment scores of the generated responses evaluated by a sentiment classifier, which marks the sentiment of a sentence ranging from 0 (negative) to 1 (positive). *Posi%* denotes the percentage of the responses being recognized as positive by the sentiment classifier.

The human evaluations involve: quality *Qual*, diversity *Div*, and the emotion reflected from the generated responses *Emo*. *Emo* score covers: 1. whether the response is emotionally positive; 2. whether the response is empathetic (cottoned with query's emotion). We hire five annotators and each annotator evaluates 250 randomly selected test samples. The annotation scores range from 1 to 5 (See details about the setting in Appendix C).

4.2 Overall Performance

Table 1 shows the performances of our models and baselines in automatic evaluations. In our applications, the transformer framework is much more suitable than Seq2Seq according to the comparisons among row $1 \sim 4$, so we implement our model and most baselines based on the transformer framework for a fair comparison.

Row 2 and Row 4 to 6 of Table 1 show the baselines considering the emotion. ECM models (Zhou et al., 2018) (**S2S+ECM** and **Trs+ECM**) are more skilled at generating positive responses rather than emotion-independent models (**S2S** and **Trs**). **Trs+RL** (Shin et al., 2020) outperforms **Trs** but slightly underperforms **Trs+ECM** in *Emotion*. The reason is RL encourages positive utterances but it's hard to train due to the differentiation difficulty on generation model (Yu et al., 2017). In terms of appropriateness and diversity, we observe slight performance drops on **S2S+ECM**, **Trs+ECM**, and **Trs+RL** compared to their emotion-independent variants, and the similar phenomenon can be found in their original paper (Shin et al., 2020). **Trs+Dual** (Shen and Feng, 2020), the strongest baseline, further enhances the performance on all aspects.

Our proposed models surpass all the baselines at most metrics. Our models make clear improvements in *Appropriateness* and *Emotion*. Even if

	Appropriateness		*Diversity*		*Emotion*	
	Bleu1 Bleu2 Bleu3 Bleu4	Nist3 Nist4	Dist1 Dist2 Dist3 Dist4	Ent	Sent	Posi%
$c = N$	17.749 10.967 9.282 8.073	1.969 2.035	0.015 0.227 0.486 0.689	5.926	0.702	73.4%
$c = 500$	18.357 12.121 9.856 8.571	1.971 2.013	**0.015 0.252 0.533 0.714**	**6.073**	0.693	72.6%
$c = 100$	19.094 12.449 9.937 8.603	2.045 2.098	0.013 0.213 0.475 0.659	5.924	0.732	78.9%
$c = 10$ (Ours (soft))	19.324 12.755 10.131 8.736	2.083 2.121	0.013 0.201 0.426 0.587	5.793	**0.779**	**83.1%**
$c = 1$ (Ours (hard))	**19.715 13.026 10.287 8.859**	**2.122 2.155**	0.014 0.206 0.451 0.608	5.884	0.764	81.3%

Table 3: The results of our models with different c in memory read. $c = N$ shows the model reads all the memory slots, where K is the number of slots. $c = 10$ and $c = 1$ are our model with soft and hard read shown in Table 1.

improving *Diversity* is not our purpose and our constraints on the emotion and responding strategies naturally limit the diversity, our model still works well on diversity owing to *Strategy Mixer*. **Ours (soft)** does better in *Emotion* and **Ours (hard)** obtains a better performance in *Appropriateness* and *Diversity*. We further analyze it in Sec. 4.4. Our two variants exceed the baselines on human evaluations. Note that *Emo* considers not only positive emotion but also empathy (correlated to query's emotion) in responses. **Ours (soft)** outperforming **Ours (hard)** in *Emo* indicates a mixture of emotions is better than a single one.

4.3 Ablation Study

Table 2 shows the ablation study on our proposed components. **Ours − SM** denotes our model's variant without a strategy mixer, where the model directly feeds the memory outputs to *CCM*. This variant underperforms our model (**Ours (hard)**) verifying the effectiveness of the strategy mixer proposed in Sec. 3.4. The reason is our strategy mixer learns to mix the strategies via gradient descent while the strategy vector $\boldsymbol{z}$ of **Ours − SM** is from the memory directly which is non-differentiable for end-to-end training.

Ours − fixing CCM means *CCM*'s (Sec. 3.5) parameters are not frozen in the second training phase. It optimizes the transformer by treating it as a policy model and regarding text generation as a sequence of actions. The strategy inevitably faces large action space and complicated back-propagation through the transformer, thus **Ours − fixing CCM** results remain suboptimal. As **Ours − fixing CCM**'s training strategy is similar to our baseline **Trs+RL** (Shin et al., 2020), the results confirm that **Trs+RL**'s strategy is not so effective.

The variant without loss of sentiment score (**Ours** $-\mathcal{L}_{sent}$) behaves similarly to our full model except in *Emotion*. It shows that the supervision from *PSE* (Sec. 3.6) is necessary to lead to positive conversations and does not affect the appropriateness and diversity so much. **Ours − Gaussian** and **Ours − EmoMemory** show the importance of using Gaussian distribution and the emotion-specific memory module in SM.

4.4 Analysis about the Memory Reading

We can control the way of reading the query-response memory by varying the number of c. Table 3 shows the model performances with different c. $c = 1$ and $c = 10$ is identical to **Ours (hard)** and **Ours (soft)**. $c = N$ indicates an extreme setting that our model reads all memory slots, where $K = 1000$ in our experiments. In general, the larger c tends to result in high diversity, since the model reads information from more various resources (more memory slots). The smaller c leads to higher performance on *Appropriateness* and *Emotion*, because they leverage information from a few relevant memory slots.

For the extreme settings, the variant with $c = N$ reads the whole memory and it leads to overwhelming the important information from the memory. The result on $c = 1$ shows that the responding strategy with a mixture of emotions instead of a single one harms the overall quality but helps the emotional generation.

4.5 Case Study

We conduct the case study on two cases (cases and detailed analyses in Appendix E). Our model gets visible improvements over the baselines. **Ours**'s generated responses show empathy by agreeing with users and guiding positive conversations.

5 Conclusion

In this paper, we propose an emotional chatbot with the ability to make empathetic and emotionally positive responses. We construct an emotion-specific query-response memory to abstract and memorize the correlation between users' queries and the potential responses. The model fuses the memory outputs into a latent distribution and samples an implicit responding strategy from the distribution. The above operations are supervised by a sentiment

evaluator to encourage positive emotions. Our experiments show our model's strengths in appropriateness, diversity, and generation of positive responses. In the future, we will extend our model to multi-turn conversation scenarios.

6 Ethical Considerations

The target of this paper is to build an empathetic and emotionally positive dialogue system. There are several concerns about applying this paper to real use, which may lead to ethical issues. First, we should carefully choose the applications of this paper. Our model is designed to behave as a chatbot instead of a therapeutic system. Our systems can not replace psychologists to conduct any treatment. We want to remind the users who may use our model that a patient with psychological illness should see a psychologist instead of regarding our model as a treatment.

Second, the training of a conversation model may lead to privacy disclosure because the conversation samples are sometimes from personal conversations. In this work, we try to avoid the issues mentioned above. The data source used in this paper comes from a published dataset and does not involve privacy issues for the data collection. We want to emphasize that other researchers, who want to follow and re-implementation our model, should carefully choose the training data to take care of the privacy concerns.

Third, our work validates the proposed method and baseline models on human evaluation which involves manual labor. We hire five annotators to score the generated sentences, and the hourly pay is set to 15 US$ per person, which is higher than the local statutory minimum wage.

7 Acknowledgements

Research on this paper was supported by National Natural Science Foundation of China (Grant No. 62106275 and 62025208), Natural Science Foundation of Hunan Province (Grant No. 2022JJ40558), and Hong Kong Research Grants Council (Grant No. 16204920).

References

Yu Cao, Wei Bi, Meng Fang, and Dacheng Tao. 2020. Pretrained language models for dialogue generation with multiple input sources. In *Findings of EMNLP*, pages 909–917.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*.

Pierre Colombo, Wojciech Witon, Ashutosh Modi, James Kennedy, and Mubbasir Kapadia. 2019. Affect-driven dialog generation. In *ACL*, pages 3734–3743.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. In *ICLR*.

Zhenrong Deng, Hongquan Lin, Wenming Huang, Rushi Lan, and Xiaonan Luo. 2020. Emotional dialogue generation based on conditional variational autoencoder and dual emotion framework. *WCMC*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *HLT*, pages 138–145.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, and Shuming Shi. 2019a. Generating multiple diverse responses for short-text conversation. In *AAAI*, volume 33, pages 6383–6390.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, Guodong Zhou, and Shuming Shi. 2019b. A discrete cvae for response generation on short-text conversation. In *EMNLP*, pages 1898–1908.

Jun Gao, Yuhan Liu, Haolin Deng, Wei Wang, Yu Cao, Jiachen Du, and Ruifeng Xu. 2021. Improving empathetic response generation by recognizing emotion cause in conversations. In *Findings of EMNLP*, pages 807–819.

Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2019. Gsn: a graph-structured network for multi-party dialogues. In *AAAI*, pages 5010–5016.

Charles Lee Isbell, Michael Kearns, Dave Kormann, Satinder Singh, and Peter Stone. 2000. Cobot in lambdamoo: A social statistics agent. In *AAAI*, pages 36–41.

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

Hao Jiang, Yutao Zhu, Xinyu Zhang, Zhicheng Dou, Pan Du, Te Pi, and Yantao Jia. 2021. Emotion eliciting machine: Emotion eliciting conversation generation based on dual generator. *arXiv preprint arXiv:2105.08251*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

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

Qintong Li, Hongshen Chen, Zhaochun Ren, Pengjie Ren, Zhaopeng Tu, and Zhumin Chen. 2020. Empdg: Multi-resolution interactive empathetic dialogue generation. In *COLING*, pages 4454–4466.

Qintong Li, Piji Li, Zhaochun Ren, Pengjie Ren, and Zhumin Chen. 2022. Knowledge bridging for empathetic dialogue generation. In *AAAI*.

Zhaojiang Lin, Andrea Madotto, Jamin Shin, Peng Xu, and Pascale Fung. 2019. Moel: Mixture of empathetic listeners. In *EMNLP*, pages 121–132.

Zhaojiang Lin, Peng Xu, Genta Indra Winata, Farhad Bin Siddique, Zihan Liu, Jamin Shin, and Pascale Fung. 2020. Caire: An end-to-end empathetic chatbot. In *AAAI*, volume 34, pages 13622–13623.

Siyang Liu, Chujie Zheng, Orianna Demasi, Sahand Sabour, Yu Li, Zhou Yu, Yong Jiang, and Minlie Huang. 2021. Towards emotional support dialog systems. In *ACL*.

Nurul Lubis, Sakriani Sakti, Koichiro Yoshino, and Satoshi Nakamura. 2018a. Eliciting positive emotion through affect-sensitive dialogue response generation: A neural network approach. In *AAAI*, volume 32.

Nurul Lubis, Sakriani Sakti, Koichiro Yoshino, and Satoshi Nakamura. 2018b. Unsupervised counselor dialogue clustering for positive emotion elicitation in neural dialogue system. In *SIGdial*, pages 161–170.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *COLING*, pages 3349–3358.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *ACL*, pages 5370–5381.

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

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

Lei Shen and Yang Feng. 2020. Cdl: Curriculum dual learning for emotion-controllable response generation. In *ACL*, pages 556–566.

Jamin Shin, Peng Xu, Andrea Madotto, and Pascale Fung. 2020. Generating empathetic responses by looking ahead the user's sentiment. In *ICASSP*, pages 7989–7993. IEEE.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *NeurIPS*, volume 28, pages 3483–3491.

Yiping Song, Zequn Liu, Wei Bi, Rui Yan, and Ming Zhang. 2020. Learning to customize model structures for few-shot dialogue generation tasks. In *ACL*, pages 5832–5841.

Zhenqiao Song, Xiaoqing Zheng, Lu Liu, Mu Xu, and Xuan-Jing Huang. 2019. Generating responses with a specific emotion in dialog. In *ACL*, pages 3685–3695.

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

Zhiliang Tian, Wei Bi, Xiaopeng Li, and Nevin L Zhang. 2019. Learning to abstract for memory-augmented conversational response generation. In *ACL*, pages 3816–3825.

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 *NeurIPS*, pages 5998–6008.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *AAAI*, pages 3351–3357.

Chen Xu, Jianyu Zhao, Rang Li, Changjian Hu, and Chuangbai Xiao. 2021. Change or not: A simple approach for plug and play language models on sentiment control. In *AAAI*, volume 35, pages 15935–15936.

Linxiao Yang, Ngai-Man Cheung, Jiaying Li, and Jun Fang. 2019. Deep clustering by gaussian mixture variational autoencoders with graph embedding. In *ICCV*, pages 6440–6449.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, volume 31.

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? In *ACL*, pages 2204–2213.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *ACL*, pages 654–664.

Chujie Zheng, Yong Liu, Wei Chen, Yongcai Leng, and Minlie Huang. 2021. Comae: A multi-factor hierarchical framework for empathetic response generation. In *Findings ACL-IJCNLP*, pages 813–824.

Peixiang Zhong, Chen Zhang, Hao Wang, Yong Liu, and Chunyan Miao. 2020. Towards persona-based empathetic conversational models. In *EMNLP*, pages 6556–6566.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *AAAI*, pages 730–738.

Xianda Zhou and William Yang Wang. 2018. Mojitalk: Generating emotional responses at scale. In *ACL*, pages 1128–1137.

A Information about the Datasets

A.1 Conversation Dataset

Following (Shang et al., 2015), we use the conversation dataset from weibo.com with 1.25M samples. The validation set and testing test take 5k and 1.5k samples respectively. Particularly, we crawl the pairs of a post and a reply from Weibo [2], a Chinese social media site. Then, we clean the dataset by following steps. First, we clean the conversation samples by removing the kaomoji, specific symbols, and repeated punctuation in responses. Second, we tokenized the sentences into Chinese characters and built a vocabulary consisting of the top 12K Chinese characters and covering 99.9% characters in the corpus. All the out-of-vocabulary words are replaced with a special character UNK. Then, we filter conversation samples with empty utterances (query or response) and set the maximal length of utterances as 80. Finally, we detect the abnormal speakers (e.g. advertising robots) according to their conversations.

A.2 Sentiment Classification Dataset

We train our *PSE* on a public sentiment classification dataset [3], which consists of 10K labeled samples collected from Weibo. After training, *PSE* can obtain an accuracy of 99%, indicating *PSE* is qualified to evaluate generated utterances.

A.3 Emotion Detection Dataset

To train the emotion detector *ED*, we use a Chinese emotion classification dataset, NLPCC-2014, from [4]. This dataset is also collected from Weibo. There are eight emotion categories (Anger, Disgust, Fear, Happiness, Like, Sadness, Surprise, and None (Neutral)) in total. The dataset consists of 48K labeled samples.

B Hyper-parameters and Codes

For the baselines involving Seq2Seq (**S2S** and **S2S+ECM**), we follow the setting of (Zhou et al., 2018) and its repository [5], where its encoder and decoder are 2-layer GRUs with the hidden dimension of 512. The word embedding dimension is 300. The models use SGD with a learning rate of 0.5 and a batch size of 128. For the methods involving

[2] weibo.com

[3] github.com/SophonPlus/ChineseNlpCorpus

[4] "Emotion Analysis in Chinese Weibo Texts" in tcci.ccf.org.cn/conference/2014/pages/page04_dg.html

[5] github.com/1YCxZ/ECM-seq2seq

transformer (including ours), we implement and follow the setting of "transformer base" (Vaswani et al., 2017) in the original paper. The model dimension and embedding dimension are 512, the stacked-layer number is 6, and the head number is 8. We use Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-9}$. The maximum length of queries and responses is limited to 80 for all the methods. We do not explicitly restrict the priority of the empathetic and positive emotions and the weight α in loss function is 0.9. The batch size is 64. The memory size K is 1000, and the c for memory soft reading is 10. The dimension of the $\boldsymbol{z}$, key, and values in the memory is 512. We infer responses with topK sampling with the size of 20. The *PSE* and *ED* utilize the pretrained BERT in the Chinese version [6]. We train the models on GPU (V100) with 32GB memory. The runnning time of the our model ranges from 1.5 to 3 days. For the hyper-parameters, we tuned the memory size K by trying $K = 1, 10, 100, 500, N$. For the topK sample size, we tried 10, 20, and 40. For the weight $alpha$ of the loss function, we tried 0.5, 0.7, 0.9, 1, 1.2, 1.5, and 5.

C Details about the Human Evaluation

Following the previous papers (Gao et al., 2019b,a; Tian et al., 2019), we hired five annotators from a language labeling company, which are professional in annotations for both industry and academia. The company assigned us some annotators specialized in evaluating samples considering linguistics and psychology. Each annotator evaluates 250 randomly selected samples. The number of the annotators and the quantity of the samples are in the same range as the related papers (Gao et al., 2019b,a; Tian et al., 2019; Song et al., 2020). Different models' outputs for each sample are shuffled among models and the model names are not accessible to the annotators. The annotators are required to view all information about the current sample, including the input query and the ground-truth output. We note that our experiment setting is based on single-turn conversation, so the contexts from the previous turns are not available in both the automatic evaluation and the human evaluation.

When we conduct the human annotation, we ask the annotators to obey the following instructions.

- **"Qual" score:** A response coherent to the conversational contexts and appropriate to the current topic without typos should be marked as 5 points. A valid response that only satisfies the user's query should be marked as 3 points. An irrelevant response should be scored as 1. An incomplete response that the annotators cannot get the speaker's meaning should also be assigned as 1. Points 2 and 4 are for decision dilemmas.
- **"Div" score:** 5 points stand for an output response with at least two clauses from different topics, where the topic may be transferred from 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. Such a case should be scored as 5 points). A normal response of a single clause or a single topic should be marked as 3 points. 1 point is for the universal reply (i.e., "I do think so ¡') or the response containing no more than three unique words (i.e., "That is OK‘') should be assigned to point 1. Points 2 and 4 are for decision dilemmas.
- **"Emo" score:** An output response that satisfies the two aspects (empathetic and emotionally positive) should be assigned as a full mark (5 points). A response that satisfies only one aspect should be scored as 3 points. A response that cannot satisfy any one aspect should be scored as 1 point. Points 2 and 4 are for decision dilemmas. For example, when the input query is "Kobe Bryant had a plane crash. We lost him, so sad". 5 points are for the response of "Sorry to hear that crash. He's a legend and will be my idol forever. Hope his family and fans can recover soon". 3 points are for the response of "Yes, we lost him. I'm extremely heartbroken these days". 2 points are for the response of "Come on, cheer up. We still love NBA".

D Deduction for the Loss in Strategy Mixer

The detailed deduction of the $\mathcal{L}_{\text{SM}}$ loss mentioned in the Strategy Mixer section is as follows. The $\mathcal{L}_{\text{SM}}$ is the evidence lower bound (ELBO) of the maximum log-likelihood of $P(r|q)$, where q denotes the query, r denotes the response, M denotes the memory to estimate the r, and z is the latent variable.

[6] huggingface.co/bert-base-chinese

Method	Case1	Emotion
Query	Oh, how boring the sport is! I am watching! (哦，多么无聊的运动！我在看.)	Disappointed
S2S	Yes! We, we are together! (是的！我们是一起的！)	Submissive
S2S+ECM	Yes! Thanks! （是的！谢谢！）	Submissive
Trs	I do think so, my friends and I are surprising! （我觉得也是，我的朋友和我都很惊讶！）	Submissive
Trs+ECM	Thank you for your appreciation, our players are so happy, our players are excellent. （相信你的赞赏，我们运动员很开心，很优秀。）	Positive
Trs+Dual	Thank you, my sister, you give me a better future, come on. Thanks for your caring. （谢谢你，姐姐，你给我一个好的未来，加油，谢谢你的关心。）	Positive
Trs+RL	Yes! I do not go there, you said you went there last year, let's went there at leisure. （是的！我不会去，你说过你去年要去，我们一起去度假吧。）	Plain
Ours (hard)	Yes! Our players are really hardworking. They are the best! （是的！我们的运动员真的很努力。他们是最棒的！）	Submissive, Positive
Ours (soft)	Yes, such a boring sport! Make sure to enjoy yourself! (是的，这运动真无聊！你一定要玩的开心呀！)	Empathetic, Positive
Method	Case2	Emotion
Query	The two guys are blowing! I really want to say they will lose the game! I don't believe that they can lose less than 10 points today.（这俩人在吹牛！我真想说他们会输！我不信他们能输低于10分）	Angry
S2S	Ours, that's ours. （我们的，那是我们的）	Plain
S2S+ECM	Yes, I agree. （是的，我同意）	Submissive
Trs	I also want to play, I do not know, I am not saying losing the match. （我也想打比赛，我不知道，我不是在说要输掉比赛）	Plain
Trs+ECM	That thing is possible, really possible. Do not like playing basketball.（那是可能的，真的可能。不喜欢打篮球。）	Plain
Trs+Dual	Thanks, thank you, we are family, I always think so since we should cheer for them. （谢谢你，我们是一家人，我觉得应该为他们加油。）	Positive
Trs+RL	Does reply to the two guys can make the are several people play better?（重复那俩人说的话可以让他们打的更好吗）	Ironic
Ours (hard)	How much do we want to scold! It's funny! So do you, so do we. No need to say that. Let's cheer for them. （我们是有多想责骂他们呀！很有趣，你也想，我也想。没必要说这些，让我们为他们加油吧。）	Empathetic Positive
Ours (soft)	The two guys are saying "Let's cheer for them and play together." The players are hardworking. （这俩人在说："我们为他们加油，与他们一起打比赛吧"。这些运动员真的很努力）	Empathetic Positive

Figure 3: Two cases of the comparing methods. Colored texts indicate the emotional phrase in utterances and its corresponding emotion category, where we mark the emotion categories of the sentences analyzing the cases. The first row is the user's query and the other rows display responses generated by those methods. The original utterances are in Chinese and we translate them into English.

$$
\begin{aligned}
& \log P_\theta(r|q) \\
= \; & \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log P_\theta(r|q)] \\
= \; & \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log \frac{P_\theta(r|\boldsymbol{z},q)P_\theta(\boldsymbol{z}|q)}{P_\theta(\boldsymbol{z}|r,q)}] \\
= \; & \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log \\
& \frac{P_\theta(r|\boldsymbol{z},q)P_\theta(\boldsymbol{z}|q)}{P_\theta(\boldsymbol{z}|r,q)}\frac{Q_\phi(\boldsymbol{z}|M,q)}{Q_\phi(\boldsymbol{z}|M,q)}] \\
= \; & \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log P_\theta(r|\boldsymbol{z},q)] \\
& -\mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log \frac{Q_\phi(\boldsymbol{z}|M,q)}{P_\theta(\boldsymbol{z}|q)}] \\
& +\mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log \frac{Q_\phi(\boldsymbol{z}|M,q)}{P_\theta(\boldsymbol{z}|r,q)}] \\
= \; & \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log P_\theta(r|\boldsymbol{z},q)] \\
& -KL[Q_\phi(\boldsymbol{z}|M,q)||P_\theta(\boldsymbol{z}|q)] \\
& +KL[Q_\phi(\boldsymbol{z}|M,q)||P_\theta(\boldsymbol{z}|r,q)] \\
\geq \; & \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log P_\theta(r|\boldsymbol{z},q)] \\
& -KL[Q_\phi(\boldsymbol{z}|M,q)||P_\theta(\boldsymbol{z}|q)] \\
= \; & \mathcal{L}_{\text{SM}}
\end{aligned}
\tag{6}
$$

In Eq.1, to make the equation more clear, we rename the parameters of the decoder is θ and rename the parameters of the prior as φ. Hence, we obtain the Eq.1 as follows,

$$
\begin{aligned}
\mathcal{L}_{\text{SM}} &= \mathbb{E}_{\boldsymbol{z}\sim Q_\phi(\boldsymbol{z}|M,q)}[\log P_\theta(r|\boldsymbol{z},q)] \\
&- KL[Q_\phi(\boldsymbol{z}|M,q)||P_\varphi(\boldsymbol{z}|q)] \\
&\leq \log P(r|q)
\end{aligned}
$$

E Case Study

Fig. 3 shows two cases mentioned in Section 4.5, which compares the performance of all the methods. In the two queries, users express the emotion of disappointed and angry respectively. **S2S** and **Trs** offer general responses that simply submissively follow the user's queries. **S2S+ECM**, **Trs+ECM**, and **Trs+Dual** incorporate the positive emotion into the responses but sacrifice the consistency between queries and responses. Interestingly, **Trs+RL** generates an ironic response, indicating the explicit feedback makes a big impact on the emotion of the response. Our methods achieve the best performance among all the methods, especially **Ours (Soft)**. The generated responses agree with the user's points and show empathy with the user by continuing their topic. Then, responses also guide the conversation to be positive.