EM Pre-training for Multi-party Dialogue Response Generation

Yiyang Li[1,2] and Hai Zhao[1,2,*]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University

[2] Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University eric-lee@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

1 Introduction

Inspired by the tremendous success in pre-training large language models (PLMs) in general domains (Devlin et al., 2019; Clark et al., 2020; Radford et al., 2018), efforts have been made to train PLMs for dialogue response generation (Zhang et al., 2020; Bao et al., 2020; Chen et al., 2022). However, they constrain the dialogues to be either two-party, or sequential structured (i.e. each utterance replies directly to its previous utterance). Different from them, a multi-party dialogue can involve multiple interlocutors, where each interlocutor can reply to any preceding utterances, making the response relations of the dialogue tree-structured and much more complicated (Zhang et al., 2018; Le et al., 2019; Shi and Huang, 2019; Wang et al., 2020). Besides, the speaker and addressee of a response utterance should be specified before it is generated in multi-party scenario, making the annotated data for multi-party dialogue response generation (MPDRG) less available.

Figure 1 illustrates an example of MPDRG task taken from the Ubuntu IRC benchmark (Hu et al., 2019). The upper part shows the tree-structured addressee relations of the dialogue, where the arrows point from addressees to speakers, and different colors represent different interlocutors. The middle part displays the content of the dialogue history, where U_7 is the response to be generated. The addressee (U_6) and the speaker (#4) of it are given, and the content of this response is the target of our model. The lower part gives the human response, which is also called the ground truth reference.

Previous works on MPDRG fine-tune generative PLMs on small multi-party dialogue datasets with explicit addressee annotations. They utilize the response annotations to form a tree-structured response graph, then encode the dialogue history using either homogeneous or heterogeneous Graph Neural Networks (GNNs) (Hu et al., 2019; Gu et al., 2022). Nevertheless, none of them make attempts to pre-train a response generation model for multiparty dialogues due to the lack of large-scale corpora with annotated addressee labels.

To solve the aforementioned problem of data scarcity, we propose an EM approach that iteratively performs the expectation steps to generate addressee labels, and the maximization steps to optimize a response generation model. Specifically, we treat the addressee of each utterance in the dialogue history as a discrete latent variable z. During the E-steps, given the current dialogue history c_t and the the response utterance r_t, we model the distribution of the current addressee z_t as $p(z_t|c_t,r_t;\vartheta)$, where ϑ is the current model parameters. During the M-steps, we sample (c_t,r_t,z_t) triplets from distributionmize the generative modelsamples. With the iteration number

[*] Corresponding author. This paper was partially supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 92–103
July 9-14, 2023 ©2023 Association for Computational Linguistics

increasing, the $pp((zr_t|_tc|c_t,r_t,z_t;_t;\vartheta\vartheta))$ and opti-on these

accuracy of latent variable prediction and the quality of generated responses will grow together. It is worth noting that during these iterations, annotated addressee labels are not required, which makes it possible to leverage the huge amount of multi-party dialogue corpora without addressee labels. We provide theoretical analyses to prove the feasibility of our EM method, and conduct experiments on the Ubuntu IRC benchmark, which is used in previous works (Hu et al., 2019; Gu et al., 2022).

The contributions of our work can be summarized as the following three folds:

- To the best of our knowledge, we are the first to study the pre-training of multi-party dialogue response generation, which is much more challenging and complicated than two-party dialogues.
- We put forward an EM approach to alleviate the scarcity of multi-party dialogue data with addressee labels, making it possible to pre-train a model with huge amount of unlabeled corpora. • We provide theoretical analyses to prove the feasibility of our EM pre-training method, and experimental results on the Ubuntu IRC benchmark show our pre-trained model achieves state-of-theart performance compared with previous works.

2 Related Works

2.1 Pre-training for Response Generation

In recent years, researchers have gradually drawn their attention from retrieval-based dialogue systems to generation-based ones. Thanks to the huge amount of two-party dialogue corpora, various PLMs for two-party dialogue response generation have been proposed.

Zhang et al. (2020) propose DialoGPT, which utilizes the sequential response chains in the Reddit Corpus to pre-train an auto-regressive response generation model based on the architecture of GPT (Radford et al., 2018). Different from their work, which focuses on sequential dialogue history, our work aims to solve the case where the agent can respond to any previous utterance in a tree-structured dialogue history.

Bao et al. (2020) propose PLATO, which models the conversational intents as K discrete latent

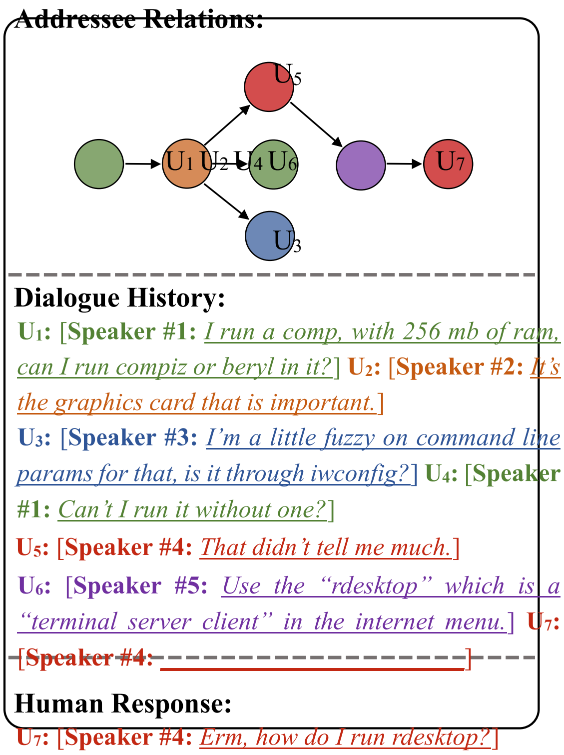

Figure 1: An example of multi-party dialogue response generation task, better view in color.

variables, then utilizes response selection, bag-ofwords prediction, and language modeling objectives to train the model. DialogVED (Chen et al., 2022) further extends the discrete latent variables to continuous ones, and models them with a multivariable Gaussian distribution. It utilizes KL divergence reduction to optimize the parameters of the latent distribution and applies masked language modeling, response generation, and bag-of-words prediction to train the whole model. PLATO and DialogVED focus on two-party conversations, and the conversational intents they put forward have no corresponding concepts of actual entities (e.g., intent to argue, intent to end a conversation, and so on). Distinct from their works, we lay emphasis on multi-party dialogues, and the latent variables of our method have actual meanings: variable $z_t = j$ indicates that the addressee of the response at the t_{th} turn is the j_{th} utterance.

2.2 Multi-party Dialog Response Generation

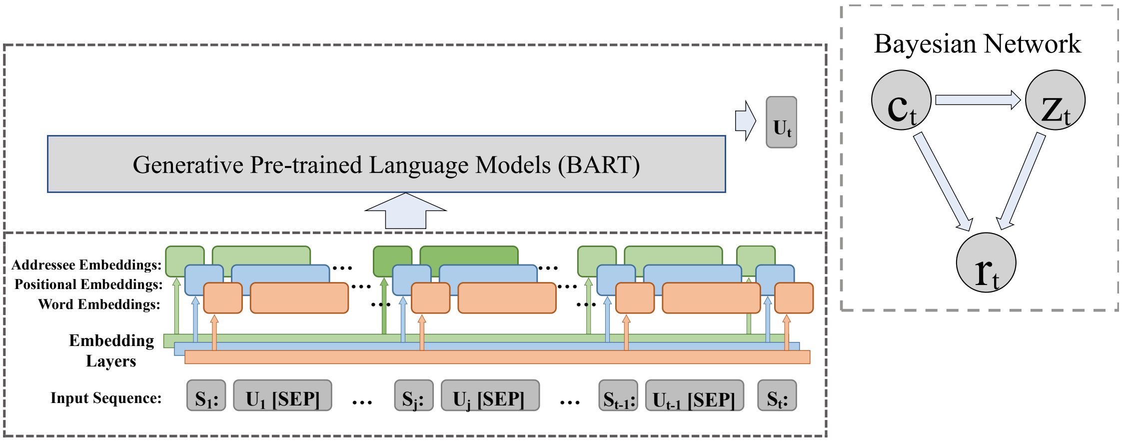

Figure 2: The overview of our model architecture. The left part shows how we incorporate the addressee information into response generation by adding addressee embeddings. The right part illustrates a Bayesian Network of how a response is generated given the current dialogue history c_t and the addressee z_t.

Several previous works have studied the MPDRG task. Hu et al. (2019) extract a subset of the Ubuntu Dialogue Corpus (Lowe et al., 2015) with explicit addressee labels to construct the Ubuntu IRC benchmark, where they propose a Graph Structured Neural Network (GSN) for dialogue modeling. Specifically, they first treat each utterance of a dialogue as a node, and the addressee relations as edges to construct a dialogue graph, then make use of GNNs to encode the dialogue history. Finally, they adopt a Gated Recurrent Unit (GRU) with cross attention as the decoder to generate responses. Gu et al. (2022) put forward HeterMPC, which models the dialogue history as a heterogeneous graph. In detail, they first design six types of edges: reply and replied-by, address and addressed-by, speak and spoken-by, among two kinds of nodes: interlocutor nodes and utterance nodes, and then encode the dialogue history using Transformers (Vaswani et al., 2017) together with heterogeneous GNNs. Finally, they utilize a Transformer Decoder to generate responses. Instead of fine-tuning models on a small dataset with annotated addressee labels as these existing work did, our work focuses on the utilization of large unlabeled corpora to pre-train a response generation model for multi-party dialogues.

3 Methodology

To design a model for multi-party dialogue response generation and make it compatible with the EM training algorithm, there are two important things to consider: how to model $p(r^t|c^t,z^t;\vartheta)$ in the maximization step, and how to compute

ption, we will first address these two problems, then$(z_t|c_t,r_t;\vartheta)$ in the expectation step. In this sec-

mathematically derive the feasibility of our EM pre-training algorithm.

3.1 Task Formulation

Given an input sequence of the dialogue history and the speaker of the response at time step t, X = {together with the addressee of the responseS_1:U_1[SEP]S_2:U_2[SEP]...S_{t-1}:U_{t-1}[SEP]S_{z_t} =t:j},, our goal is to train a model that can generate an response Y = U_t. Here each S_i is the name of the speaker at time step i, which is represented as *Speaker #S_i* like those in Figure 1.

$U_i = \{w_{i1}, w_{i2}, \ldots, w_{in_i}\}$ is the content of the i_{th} utterance with n_i words. $z_t = j$ represents that S_t speaks to S_j, who utters U_j, and [SEP] is a special token that indicates the end of a dialogue turn.

3.2 Addressee Modeling

In this section, we answer the first question: how to model $p(r_t|c_t,z_t;\vartheta)$, or in other words, how to incorporate the addressee information $z_t = j$ into the process of generating a response r_t. We design a straightforward method that adds addressee embeddings to the positional encodings and word embeddings, before they are further encoded by a PLM. The left part of Figure 2 illustrates this method, where we use an embedding look-up table with 2 entries to indicate whether a word belongs to the addressee utterance or not. Specifically, if a word is in the addressee utterance, it will get its addressee embedding from entry 1, otherwise from entry 0. Since addressee modeling is not the key contribution of this work, we just adopt the most straightforward and effective way. In our experiments, we use BART (Lewis et al., 2020) as the backbone PLM, following previous works (Gu et al., 2022). Due to the page limit, the proverbial architecture of Transformer and BART are omitted here.

3.3 Latent Variable Prediction

In this section, we answer the second question: how to compute $p(z_t|c_t,r_t;\vartheta)$ in the expectation step, or in other words, how to predict the distribution of the unlabeled addressee z_t, given the current dialogue context c_t, response r_t, under parameters ϑ. The solution to this question is essentially the most important part of our method since it delicately solves the problem of data scarcity in MPDRG.

Let's consider what humans will do to participate in a multi-party conversation. First, we will read the dialogue history c_t, then choose an addressee z_t to reply. Once c_t and z_t are determined, we will utter a response according to the content of the whole dialogue and the addressee utterance. The right part of Figure 2 gives the Bayesian Network of the above process, where the joint distribution of (c_t,z_t,r_t) can be factorized as:

$$p(c,z,r) = p(c) \cdot p(z|c) \cdot p(r|c,z) \quad (1)$$

Here we omit the subscript t and model parameters ϑ for simplicity. Given Eq. (1), $p(z|c,r;\vartheta)$ can be derived as:

$$\begin{aligned} p(z|c,r) &= \frac{p(c,z,r)}{p(c,r)} \\ &= \frac{p(c)\cdot p(z|c)\cdot p(r|c,z)}{p(c)\cdot p(r|c)} \\ &= \frac{p(z|c)\cdot p(r|c,z)}{p(r|c)} \end{aligned} \quad (2)$$

We assume that the probability of choosing any previous utterance as the addressee is the same given the current dialogue history, which means $p(z|c)$ obeys a uniform distribution. Meanwhile,is independent of z, leaving the denominatoronly the term $p(rp|(c,zr|c))$. Now, we can induce that:

$$p(z|c,r) \propto p(r|c,z) \quad (3)$$

Therefore, for each $z^i, i = 1,2,...,t-1$, we have:

$$p(z^i|c,r) = \frac{p(r|c,z^i)}{\sum_{j=1}^{t-1} p(r|c,z^j)} \quad (4)$$

In practice, we can use the generative model $p(r^t|c^t,z^t;\vartheta)$ to compute the probability distribu-by Eq. (4). tion of $p(z^t|c^t,r^t;\vartheta)$

3.4 Expectation-Maximization Process

Figure 3 illustrates the overview of our EM training process. During the E-steps, we compute the probability distribution of the latent variable (the addressee z). During the M-steps, we sample (c,r,z) triplets from this distribution and optimize the generative model by standard training algorithms. The Expectation Step is to compute the conditional distribution of the latent variable z_t, given the observed data (c_t,r_t) and the current model

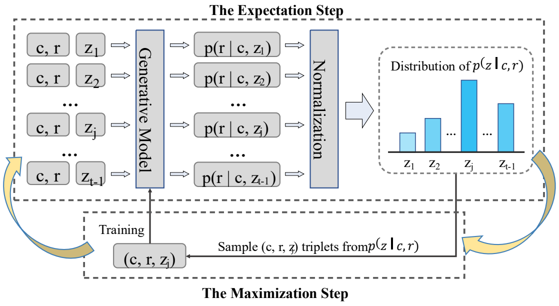

Figure 3: The overview of the EM process, where the expectation steps and maximization steps are performed alternately and iteratively.

parameters ϑ, where Eq. (4) gives a reasonable approximation of this value. Specifically, for a sample (c_t, r_t), with the model parameters ϑ fixed, we first calculate the un-normalized probability of each of the i_{th} $(i < t)$ utterance being the addressee: $p(r_t|c_t, z_t^i; \boldsymbol{\theta})$ using Eq. (3), then normalize them

to get the conditional distribution of z_t using Eq. (4). Once $P(z_t|c_t, r_t; \vartheta)$ is obtained, we sample (c_t, r_t, z_t) triplets from this distribution, which is further used in the maximization step.

The Maximization Step is analogical to the normal training process. Given the sampled $\{(c_t^k, r_t^k, z_t^k)\}_{k=1}^N$ triplets, where N is the total number of samples, our goal is to minimize the auto-regressive language modeling loss:

$$\mathcal{L}_G = -\sum_{k=1}^{N}\sum_{i=1}^{n_k}\log p\left(w_i^k \mid w_{<i}^k, c_t^k, z_t^k; \boldsymbol{\theta}\right) \quad (5)$$

where w_i^k is the i_{th} word in the response of the k_{th} sample: $r_t^k = \{w_i^k\}_{i=1}^{n_i}$, and n_i is the length of this response.

Compared with the vanilla EM algorithm, there are several differences in our implementations. First of all, we do not use the initial model to generate the training data for the first round of the maximization step. Instead, we utilize the discourse parser provided by Shi and Huang (2019) to predict the addressee of each utterance in the unlabeled corpus to get a coarse initial training dataset. The reason for this initialization method is that the initialization of training data (or model parameters) is vital to the EM method, which helps it converge to a better point. Second, rather than sampling z_t from its conditional distribution, we adopt a hard EM approach which takes the value z_t^i with highest probability as the predicted label, where $i = \operatorname{argmax}_i p(z_t^i|c_t, r_t; \vartheta)$. This hard EM approach is proved as more effective to boost the performance (Min et al., 2019). Finally, to ensure the quality of the generated training data in the maximization step, we set a hyper-parameter $\alpha \in [0,1]$ to control the proportion of training data that is actually used. Specifically, we first rank the prediction confidence of each z_t^k according to the value of $p(z^{tk}|c^{kt}, r^{tk}; \vartheta)$, then pick the top $\alpha \times N$ samples with the highest confidence scores. In our experiments, α is dynamically set to ensure the addressee prediction accuracy of the selected samples is over 80% in an annotated validation set.

3.5 Proof of Feasibility

In a multi-party dialogue corpus without annotated addressee labels, a usual solution to train a response generation model is to maximize the marginal loglikelihood (or incomplete log-likelihood) over all possible addressees:

$$\ell(c,r;\vartheta) = \log p(r|c;\vartheta) = \log \sum_i p(r,z_i|c;\vartheta) \quad (6)$$

However, this objective is hard to optimize since the distribution of z is hard to obtain. Here, we define an expected complete log-likelihood where our estimation of $p(z_t|c_t, r_t; \vartheta)$ can come to rescue:

$$\hat{\ell}(c,r;\boldsymbol{\theta}) = q(z_i)\sum_i \log p(r,z_i|c;\vartheta)$$
$$q(z) = p(z_t|c_t, r_t; \vartheta) \quad (7)$$

Our new objective now becomes maximizing the expected complete log-likelihood. The relation between ℓ and $\hat{\ell}$ can be derived as follows:

$$\begin{aligned}\ell(c,r;\vartheta) &= \log \sum_i p(r,z_i|c;\vartheta) \\ &= \log \sum_i q(z_i) \cdot \frac{p(r,z_i|c;\vartheta)}{q(z_i)} \\ &\geq \sum_i q(z_i) \cdot \log \frac{p(r,z_i|c;\boldsymbol{\theta})}{q(z_i)} \\ &= \sum_i q(z_i) \cdot \log p(r,z_i|c;\vartheta) - \sum_i q(z_i) \cdot \log q(z_i) \\ &= \hat{\ell}(c,r;\vartheta) + H_{q(z)}\end{aligned} \quad (8)$$

where the third line is derived from the *Jensen Inequality*, and $H_{q(z)}$ is the entropy of the distribution of z. Since $H_{q(z)} \geq 0$, we can derive that

$\hat{\ell}(c,r;\vartheta) \leq \ell(c,r;\vartheta)$, which means $\hat{\ell}$ is the lower bound of ℓ. By maximizing the lower bound $\hat{\ell}$, we can indirectly maximize ℓ, which is originally hard to optimize. Another important observation is hat $\hat{\ell} = \ell$ if and only if $q(z) = p(z_t|c_t,r_t;\vartheta)$, which is exactly what we calculate during the E-steps in Eq. (7). Though the derivation of the posterior distribution of z is not exact since we assume uniform prior in Eq. (2), it is still much closer to the real distribution compared to random $q(z)$.

Table 1: Results on the Ubuntu IRC benchmark, where the upper part presents models of previous works, the middle part shows our backbone model BART together with our method under different settings, and the lower part shows the ablation studies.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
GPT-2 (Radford et al., 2018)	10.37	3.60	1.66	0.93	4.01	9.53
GSN (Hu et al., 2019)	10.23	3.57	1.70	0.97	4.10	9.91
HeterMPC$_{BART}$ (Gu et al., 2022)	12.26	4.80	2.42	1.49	4.94	11.20
BART (Lewis et al., 2020)	11.25	4.02	1.78	0.95	4.46	9.90
Pre-training Only (PO)	11.78	4.67	2.38	1.41	4.98	11.19
Fine-tuning Only (FO)	11.47	5.11	2.98	2.11	5.23	11.31
Pre-training + Fine-tuning (PF)	**12.31**	**5.39**	**3.34**	**2.45**	**5.52**	**11.71**
FO + Reply-Chain	9.11	3.52	1.99	1.35	4.32	9.36
PO w/o EM	10.03	3.90	2.03	1.18	4.56	9.66
PF w/o EM	11.39	5.04	3.02	2.15	5.27	11.20
Denoising + Fine-tuning	11.49	5.08	3.02	2.13	5.25	11.28

It is worth noting that the global optimal point is not guaranteed to be reached by this algorithm, and it depends heavily on the initialization of model parameters or the training data for the first round of the maximization step. This explains the reason why we utilize a discourse parser to get a coarse initial training dataset instead of using the expectation step at the first iteration in Section 3.4.

4 Experiments

In this section, we first introduce the datasets to pre-train and evaluate our model, then present the experimental results and comparisons with previous methods.

4.1 Datasets and Experimental Setups

For pre-training, we adopt the second version of Ubuntu Dialogue Corpus (Lowe et al., 2015), which contains no annotated addressee labels. The original dataset contains 1M dialogues for training, and 0.5M dialogues for validation and testing, respectively. Dialogues that contain less than 4 turns, or have overlap with the dataset for the downstream task (the Ubuntu IRC benchmark, Hu et al. 2019), are excluded from the pre-training data. After filtering, we eventually get a pre-training dataset that contains 764,373 dialogues.

For fine-tuning, we follow previous works (Hu et al., 2019; Gu et al., 2022) to adopt the Ubuntu IRC benchmark, which is constructed by extracting all utterances with response addressees indicated by the “@" symbol in the Ubuntu Dialogue Corpus. In total, this dataset consists of 311,725 dialogues for training, and 5,000 dialogues for validation and testing, respectively. It is worth noting that this dataset contains addressee labels for every single utterance in the dialogue history, which are utilized by previous methods, yet not by ours.

For both pre-training and fine-tuning, BART (Lewis et al., 2020) is used as the backbone model. Before pre-training, we initialize the pre-trained weights from BART-base. During the process of pre-training, we evaluate our model on the validation set of the Ubuntu IRC benchmark, and the best checkpoint is saved for the fine-tuning process.

4.2 Baseline Models and Evaluation Metrics

Table 1 shows the results of our method and previous models, where GPT-2, GSN, and HeterMPC (Radford et al., 2018; Hu et al., 2019; Gu et al., 2022) are introduced in section 2.1 and 2.2, respectively. BART is a sequence-to-sequence model with encoder-decoder Transformer architecture and is trained using denoising objectives. Following Hu et al. (2019), we also adopt BLEU-1 to BLEU-4, METEOR, and ROUGE-L as the automatic evaluation metrics, which can be calculated using the *pycocoevalcap* package. Besides automatic evaluation, human evaluation is also conducted and will be introduced in Section 4.4.

4.3 Automatic Evaluation Results

Let's firstly focus on the upper and middle part of Table 1, where we present the results of previous models and our methods. Three settings of our method based on BART are experimented with: pre-training only (PO), fine-tuning only (FO), and pre-training-fine-tuning (PF). Results of PO are obtained by directly using the pre-trained model to generate the response for each dialogue. FO means the checkpoint of BART is directly finetuned on the Ubuntu IRC benchmark without pretraining. PF follows a pre-training-fine-tuning paradigm, where the best checkpoint of the pretraining process is further fine-tuned on the downstream dataset.

Three observations can be seen from the table. First of all, solely pre-training with our proposed EM method with unlabeled corpus is already able to achieve comparable results with the previous state-of-the-art (SOTA) models. It is surprising since the pre-training requires no annotated addressee labels, while previous models not merely utilize the addressee information of the response utterance, but also make use of the addressee labels of the dialogue history to form a response graph. Second, fine-tuning our model on the downstream dataset with the ground truth addressee labels yields better results compared with pre-training only. Since it uses the ground truth addressee labels of responses, the results of it can be regarded as an upper bound of what the EM training can achieve. Besides, FO outperforms the previous SOTA model by large margins with even simpler architecture and fewer annotations (without addressee labels in the dialogue history), demonstrating the effectiveness of our proposed addressee embeddings. Finally, by further fine-tuning the pre-trained checkpoint with the ground truth addressee labels, we achieve the best performance on all metrics, which shows the transferability of our pre-trained model.

Model	Score	Kappa	Best (%)
Human References	2.20	0.56	28.00
BART	1.68	0.45	8.00
$\text{HeterMPC}_{\text{BART}}$	1.88	0.48	8.00
Ours (PF)	**1.92**	0.47	**28.00**

Table 2: Human evaluation results, where *Score* is the average score and *Best* means the ratio of each system being the best response.

4.4 Human Evaluation Results

For human evaluation, we recruit a team with 8 members who have at least a Bachelor's degree in Computer Science and are familiar with Ubuntu and Linux. We randomly sample 100 examples from the testing set, then ask the team members to score each prediction and select the best one. The quality scores are considered in terms of three independent aspects: 1) relevance, 2) fluency and 3) informativeness. They are scored from 0-3 and the average values were reported. The evaluation results are shown in Table 2, where our model (Pre-training + Fine-tuning) constantly outperforms vanilla BART and the previous SOTA model $\text{HeterMPC}_{\text{BART}}$. We also report the Fleiss's Kappa to indicate the agreement between annotators. Besides, the ratio of our predictions being the best response is the same as that of human responses, demonstrating the high quality of the generated responses of our model.

5 Analysis

In order to get more insights into the proposed EM pre-training method, we dive deeper into it by conducting extensive analyses.

5.1 Ablation Study

We conduct ablation studies to investigate the contribution of our different designs, whose results are tabulated in the lower part of Table 1.

Firstly, let's focus on the first line of the lower part. To study whether other utterances that are not in the reply chain of the current addressee can help to generate a better response, we extract the reply train by traversing from the current leave utterance (the response) up to the root node (the first utterance), then train a model by inputting this chain only. We see a large performance drop on all metrics in this setting, demonstrating the significance of the side information provided by the whole context.

Second, let's pay attention to the second and third lines of the lower part. In order to study the effect of the EM pre-training process, which is the key contribution of our work, we remove this process and pre-train a model using only the addressee labels obtained from the discourse parser (i.e. the initial training data used in the first iteration of our EM approach). A sharp performance drop is observed compared with PO and PF with our proposed EM pre-training strategy, demonstrating the significance of our design. Without the iterative EM procedure, the noisy addressee labels obtained from the discourse parser can cause error propagation, which makes the model learn noisy features to predict a response, and hurts the performance.

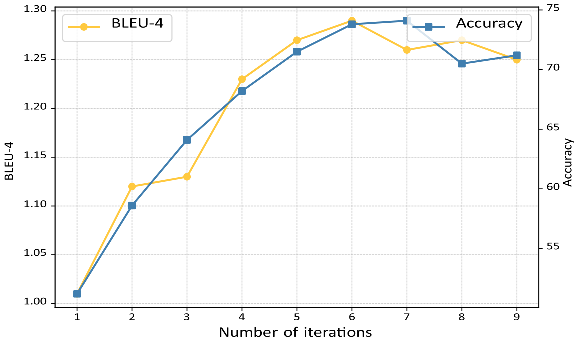

Figure 4: Line chart of the BLEU-4 score and addressee prediction accuracy with the increase of EM iterations.

Finally, aiming at investigating whether the performance gains come from seeing more in-domain data in the pre-training process, we use the same pre-training data to train another model with the denoising objectives proposed in BART (Lewis et al., 2020), then also fine-tune it on the Ubuntu IRC benchmark. The last line of the lower part presents the results, where we observe nearly the same performance compared with FO. This observation indicates that simply performing domain adaptation using the general pre-training objectives is insufficient to benefit the MPDRG task.

5.2 Response Generation vs. Addressee Prediction

In Section 3.3, we prove that $p(z|c,r) \propto p(r|c,z)$. To verify the correctness of this equation and also to investigate the training process of our EM strategy, we draw the line chart of the BLEU-4 score and addressee prediction accuracy of the top-30% confidence samples on the validation set with the increasing of pre-training iterations. The addressees are predicted using Eq. (4), where we take the z^i with the highest conditional probability as the predicted addressee.

Figure 4 illustrates the trending of the BLEU-4 score and addressee prediction accuracy. On the one hand, we see that the trending of both metrics is consistent, which means with a more powerful response generation model comes a higher addressee prediction accuracy. This observation verifies the correctness of Eq. (3). On the other hand, with the increasing of iterations, both metrics grow mutually, then reach their tops at around the 6_{th} iteration, demonstrating the effectiveness of the EM process.

Addressee relations and dialogue history are in Figure 1.

Human Response:
U_7: [**Speaker #4:** *Erm, how do I run rdesktop?*]

Generated Responses:
Our Method: [Speaker #4: *Well, how do I install rdesktop from the terminal?*]
Baseline Model: [Speaker #4: *I tried but it didn't work.*]

Figure 5: The first example of Case Studies, which shows the generated responses of our model and the baseline model.

5.3 Case Studies

To understand the effect of our method intuitively, we sample two cases from the testing set and present them in this section.

Figure 5 illustrates an example whose addressee relations and dialogue history are shown in Figure 1. This conversation is about how to run the *compiz* or *beryl* in a *comp* with 256MB RAM. *Speaker #2* points that *it's the graphic card that is important*, but *Speaker #4* seems unsatisfied by saying *that didn't tell me much*. After that, *Speaker #5* suggests using the *rdesktop* and *Speaker #4* replies him/her. Our model is able to capture the key information *rdesktop* and *terminal* in the addressee utterance U_6, and generate a proper response *Well, how do I install rdesktop from the terminal*, which is very close to the human answer and even better with more information *from the terminal*. On the contrary, the baseline model (BART) fails to capture the addressee information and just replies with a safe response *I tried but it didn't work*. This case shows the great significance of modeling the addressee information, and also demonstrates the effectiveness of our model design.

Figure 6 presents another example sampled from the testing set, where we investigate how different addressee labels affect the generated responses. In the figure, different colors represent different utterances in the *Dialogue History* part, and different responses generated by giving the corresponding utterances as addressees in the *Generated Responses* part. This conversation is about discussing the file system in Ubuntu that can share on a network with windows machines. When the addressee is given as U_1, our model suggests using *samba*, which is a solution to the question of U_1. Responses to U_2 and U_3 are like safe responses, but they make sense in their contexts: the former expresses its confusion about a confusing utterance (U_2), and the latter expresses its gratitude to the suggestion in

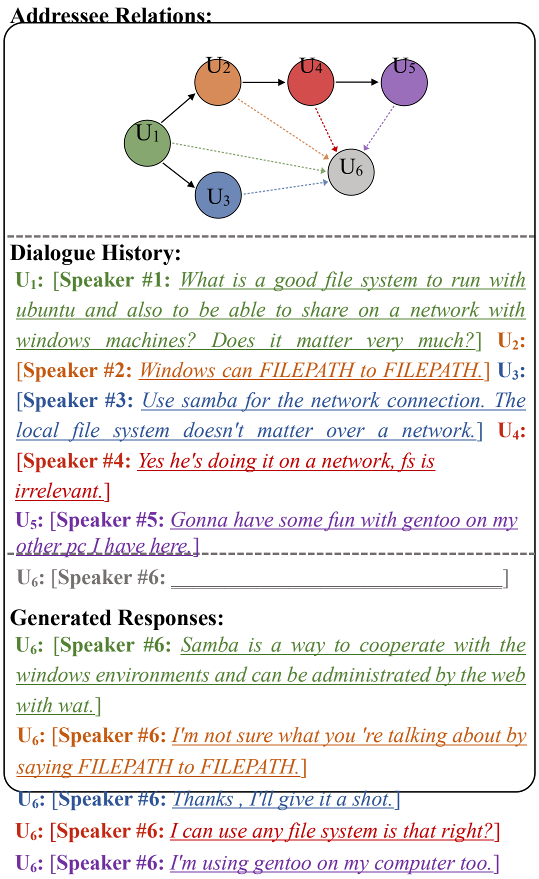

Figure 6: The second example of Case Studies, which illustrates the generated response of our model given different addressee labels. Better view in color.

U_3. Response to U_4 states his/her understanding towards U_4, and questions if his/her understanding is right. Response to U_5 acknowledges the solution *gentoo* in U_5 by saying *using gentoo on my computer too*. In general, this case demonstrates the ability of our model to generate diverse responses according to the specified addressees and contexts of the dialogue history.

5.4 Response Parser: A Byproduct for Free

Another contribution of our EM pre-training is that a response parser can be freely obtained. This byproduct comes from Eq. (4), where given a response generation model with addressee modeling, we can predict the addressee for each utterance in the dialogue. Previous literature has studied and proved that explicitly modeling the structural information is beneficial to understanding specific structured data. (Li et al., 2020, 2022a,b). In this context, the response parser can be used to infer the discourse structures, which contributes to boosting the performance of some multi-party dialogue comprehension tasks like response selection and question answering. (Jia et al., 2020; Li and Zhao, 2021; Ma et al., 2022)

6 Conclusion

Most multi-party dialogue corpora are not annotated with addressee labels, making them unable to support the pre-training of response generation models. To solve this problem, we design a simple yet effective way to model the addressee of a response as a latent variable and propose an EM pre-training approach that iteratively performs the expectation steps to generate addressee labels, and the maximization steps to optimize a response generation model. Mathematical derivation, experimental results on the Ubuntu IRC benchmark, and extensive analyses have justified the theoretical feasibility and actual effectiveness of our method.

Limitations

First, Due to the lack of datasets to evaluate the MP-DRG task, we perform our experiments only on the Ubuntu IRC benchmark and pre-train our model only on the domain of Ubuntu chats. However, the potential of our approach goes far beyond that since it is applicable to any open-domain multi-party dialogue dataset. In the future work, we will consider applying our method in more open-domain conversational datasets, such as the transcripts of TV series or movies.

Additionally, the pre-training process solely relies on the addressee information of individual turns, disregarding the reply-to relations within the dialogue history. This oversight prevents the model from benefiting from valuable contextual cues necessary for a comprehensive understanding of the multi-party dialogue. In our future work, we will explore the integration of discourse-level reply-to relations into the pre-training process to further enrich the capabilities of the model.

References

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. PLATO: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96, Online. Association for Computational Linguistics.

Wei Chen, Yeyun Gong, Song Wang, Bolun Yao, Weizhen Qi, Zhongyu Wei, Xiaowu Hu, Bartuer Zhou, Yi Mao, Weizhu Chen, Biao Cheng, and Nan Duan. 2022. DialogVED: A pre-trained latent variable encoder-decoder model for dialog response generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4852–4864, Dublin, Ireland. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pretraining text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

Jia-Chen Gu, Chao-Hong Tan, Chongyang Tao, ZhenHua Ling, Huang Hu, Xiubo Geng, and Daxin Jiang. 2022. HeterMPC: A heterogeneous graph neural network for response generation in multi-party conversations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5086–5097, Dublin, Ireland. Association for Computational Linguistics.

Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2019. GSN: A graph-structured network for multi-party dialogues. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5010–5016. ijcai.org.

Qi Jia, Yizhu Liu, Siyu Ren, Kenny Zhu, and Haifeng Tang. 2020. Multi-turn response selection using dialogue dependency relations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1911–1920, Online. Association for Computational Linguistics.

Ran Le, Wenpeng Hu, Mingyue Shang, Zhenjun You, Lidong Bing, Dongyan Zhao, and Rui Yan. 2019. Who is speaking to whom? learning to identify utterance addressee in multi-party conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1909–1919, Hong Kong, China. Association for Computational Linguistics.

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

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020. Molweni: A challenge multiparty dialogues-based machine reading comprehension dataset with discourse structure. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2642–2652, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yiyang Li, Hongqiu Wu, and Hai Zhao. 2022a. Semantic-preserving adversarial code comprehension. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3017– 3028, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yiyang Li and Hai Zhao. 2021. Self- and pseudo-selfsupervised prediction of speaker and key-utterance for multi-party dialogue reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2053–2063, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yiyang Li, Hai Zhao, and Zhuosheng Zhang. 2022b. Back to the future: Bidirectional information decoupling network for multi-turn dialogue modeling. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2761–2774, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. 2022. Structural characterization for dialogue disentanglement. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 285–297, Dublin, Ireland. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *OpenAI Technical Report*.

Zhouxing Shi and Minlie Huang. 2019. A deep sequential model for discourse parsing on multi-party dialogues. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7007–7014. AAAI Press.

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

Weishi Wang, Steven C.H. Hoi, and Shafiq Joty. 2020. Response selection for multi-party conversations with dynamic topic tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6581–6591, Online. Association for Computational Linguistics.

Rui Zhang, Honglak Lee, Lazaros Polymenakos, and Dragomir R. Radev. 2018. Addressee and response selection in multi-party conversations with speaker interaction rnns. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5690–5697. AAAI Press.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

ACL 2023 Responsible NLP Checklist AFor

every submission:

3 A1. Did you describe the limitations of your work?

The last Section. A2. Did you discuss any potential risks

of your work? *Not applicable. Left blank.*

3 A3. Do the abstract and introduction summarize the paper's main claims? *Section 1.*

7 A4. Have you used AI writing assistants when working on this paper? *Left blank.*

B 3 Did you use or create scientific artifacts? *Section 3.*

3 B1. Did you cite the creators of artifacts you used? *Section 4.*

7 B2. Did you discuss the license or terms for use and / or distribution of any artifacts? *They are publicly available and can be found on github.*

B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)? *Not applicable. Left blank.*

B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it? *Not applicable. Left blank.*

3 B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.? *Section 4.*

7 B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be. *Left blank.*

C 3 Did you run computational experiments?

Section 4.

7 C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used? *They can be found on our code.*

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

3 C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values? *Section 4.*

3 C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run? *Section 4.*

3 C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4.

D 3 Did you use human annotators (e.g., crowdworkers) or research with human participants? *Section 4.*

3 D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.? *Section 4.*

3 D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)? *Section 4.*

D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used? *Not applicable. Left blank.* D4. Was the data collection protocol approved (or determined exempt) by an ethics review board? *Not applicable. Left blank.*

7 D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
This will violate the double blind policy.