Multi-Action Dialog Policy Learning from Logged User Feedback

Shuo Zhang[1], Junzhou Zhao[1*], Pinghui Wang[1*], Tianxiang Wang[1], Zi Liang[1], Jing Tao[1], Yi Huang[2], Junlan Feng[2]

[1] MOE KLINNS Lab, Xi'an Jiaotong University, Xi'an 710049, P. R. China
[2] JIUTIAN Team, China Mobile Research
{zs412082986, liangzid}@stu.xjtu.edu.cn, {junzhou.zhao, phwang, jtao}@mail.xjtu.edu.cn,
2673021501@qq.com, {huangyi, fengjunlan}@chinamobile.com

Abstract

Multi-action dialog policy, which generates multiple atomic dialog actions per turn, has been widely applied in task-oriented dialog systems to provide expressive and efficient system responses. Existing policy models usually imitate action combinations from the labeled multi-action dialog examples. Due to data limitations, they generalize poorly toward unseen dialog flows. While reinforcement learning-based methods are proposed to incorporate the service ratings from real users and user simulators as external supervision signals, they suffer from sparse and less credible dialog-level rewards. To cope with this problem, we explore to improve multi-action dialog policy learning with explicit and implicit turn-level user feedback received for historical predictions (i.e., logged user feedback) that are cost-efficient to collect and faithful to real-world scenarios. The task is challenging since the logged user feedback provides only partial label feedback limited to the particular historical dialog actions predicted by the agent. To fully exploit such feedback information, we propose BanditMatch, which addresses the task from a feedback-enhanced semi-supervised learning perspective with a hybrid objective of semi-supervised learning and bandit learning. BanditMatch integrates pseudo-labeling methods to better explore the action space through constructing full label feedback. Extensive experiments show that our BanditMatch outperforms the state-of-the-art methods by generating more concise and informative responses. The source code and the appendix of this paper can be obtained from https://github.com/ShuoZhangXJTU/BanditMatch.

1 Introduction

Dialog policy, which takes the response by deciding the next system action, is critical in task-oriented dialog systems as it determines service quality (Zhang et al. 2020). To further strengthen the expressive power of the agent, recent works have investigated multi-action dialog policy learning (MADPL), which simultaneously generates multiple actions as the current system response (Shu et al. 2019; Jhunjhunwala, Bryant, and Shah 2020; Zhang et al. 2022). MADPL is usually formulated as a multi-label classification problem and thus solved by supervised learning (SL) based methods that imitate action combinations from labeled dialog examples (Shu et al. 2019; Jhunjhunwala, Bryant, and Shah 2020; Li, Kiseleva, and de Rijke 2020). However, the complex action combinations can exponentially enlarge the output space. SL-based models trained on limited training corpus alone do not generalize well to real-world dialog flows (Jhunjhunwala, Bryant, and Shah 2020).

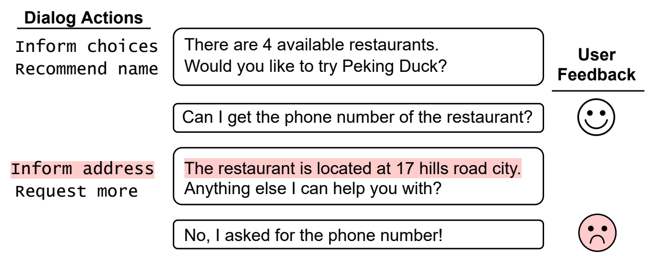

Figure 1: Example dialog fragment under multi-action dialog policy. Failing to generate a proper dialog action combination can trigger negative user feedback.

Some reinforcement learning (RL) methods are proposed to address the above issue by learning through interacting with real users while suffer from sparse and unstable reward signals (Takanobu, Zhu, and Huang 2019; Li et al. 2020). The reward, which users rate at the end of the dialog, is too expensive to collect and can be unreliable due to the existence of malicious users (Horton 2016). While user simulators are proposed to replace real users, they require tons of human work to build and suffer from the discrepancy with real users. To cope with this problem, we propose to improve MADPL with the logged user feedback received for each system response during online interactions, such as explicit turndowns (e.g., "No, I asked for Chinese food!") and implicit behaviors of retries and interruptions (see Fig. 1). The logged positive and negative user feedback, known as "bandit feedback", is cost-efficient to collect and faithful to real-world scenarios, and has been widely exploited in various applications such as machine translation (Kreutzer et al. 2018) and spoken language understanding (Falke and Lehnen 2021).

MADPL from logged user feedback is challenging since the feedback provides only partial information limited to the particular dialog actions predicted during historical in-

*Corresponding Author

teraction. The feedback for all the other predictions that the system could have made is typically not known. Moreover, the logged predictions are influenced by the historical policy deployed in the system (i.e., the logging policy), which may over or under-represent specific actions, yielding a biased data distribution (Swaminathan and Joachims 2015; Gao et al. 2021). Some straightforward ways could be derived from direct method (DM) or counterfactual risk minimization (CRM) algorithms while suffer from *extrapolation error*, which is analogous to offline RL (Jaques et al. 2020) (refer to Section 2 for detailed discussions). Specifically, task-oriented dialogs share the "one-to-many" nature that various appropriate responses may exist under the same context (Huang et al. 2020). Accordingly, in the multi-action setting, various action combinations could be considered appropriate under the same context, which are mostly not explored by the logging policy. With the biased and insufficient logged dataset, existing methods will learn arbitrarily bad estimates of the reward of the unexplored region of dialog state-action space. Thus, when facing real-world scenarios, they can generate noisy dialog actions under-explored during training and result in redundant conversations.

In this work, we cast the task of MADPL from logged user feedback as a feedback-enhanced semi-supervised learning problem and propose a method BanditMatch to solve this problem. BanditMatch integrates pseudo-labeling methods to better explore the action space through constructing full label feedback. Specifically, the logged dataset is partitioned into two subsets of full feedback and partial feedback examples, corresponding to positive and negative user feedback. The partial feedback examples with high-confidence predictions are assigned with a pseudo label to learn from full label feedback, while only the rest unconfident ones from partial feedback. We further propose a feedback-enhanced thresholding (FET) strategy that derives dynamically adjusted adaptive thresholds based on the model performance evaluated with the logged examples. Unlike fixed pre-defined thresholds, FET resists ignoring the partial feedback examples with correct pseudo labels or selecting those with wrong ones. A diagram of BanditMatch is shown in Fig. 2.

Our main contributions are summarized as follows:

- We explore to improve MADPL with logged bandit feedback. Compared with fully labeled expert examples and sparse real user ratings, bandit feedback is cost-efficient to collect and faithful to real-world scenarios.
- We propose BanditMatch to solve MADPL from logged user feedback from a semi-supervised learning perspective. Our method can effectively increase the generalization ability while alleviating the extrapolation error caused by limited exploration, and has a broader positive impact on other feedback-related applications.
- We conduct extensive experiments on the benchmark MultiWOZ dataset, and empirical results show that our BanditMatch achieves state-of-the-art performance in task success rate while generating much more concise and informative responses ($9\% \sim 23\%$ increase in Inform F1).

2 Related Work

Multi-Action Dialog Policy Learning (MADPL). Recent efforts have been made to learn a multi-action dialog policy for task-oriented dialogs where the agent can produce multiple actions per turn to expand its expressive power. Early attempts (Gordon-Hall, Gorinski, and Cohen 2020) consider the top-k most frequent dialog action combinations in the dataset and simplify MADPL as a multi-class classification problem. However, the expressive power and flexibility of the dialog agent are severely limited. Recent works (Shu et al. 2019; Li, Kiseleva, and de Rijke 2020) typically cast MADPL as a multi-label classification problem and address it with supervised learning. However, the complex action combinations can exponentially enlarge the output space (Jhunjhunwala, Bryant, and Shah 2020). The limited coverage of the output action space of the existing dialog corpus will greatly hinder SL-based methods' performance. Jhunjhunwala, Bryant, and Shah (2020)) addresses this problem by data augmentation through interactive learning. The problem can also be partially alleviated through RL-based and adversarial learning based methods (Takanobu, Zhu, and Huang 2019; Li, Kiseleva, and de Rijke 2020) by allowing the model to explore more dialog scenarios. Unlike existing works that rely on limited training corpus or unstable and sparse rating rewards from real users or user simulators, we explore to improve MADPL with logged bandit feedback that is cost-efficient to collect and faithful to real-world scenarios.

Learning from Logged User Feedback. The user feedback logged during online interaction (e.g., the number of ranked articles the user read), known as "bandit feedback", can be recorded at little cost and in huge quantities (Joachims, Swaminathan, and De Rijke 2018). Learning from bandit feedback is typically addressed from two lines of algorithms: direct method (DM) and counterfactual risk minimization (CRM). DM learns to directly predict user rewards with a regression estimator and takes as a prediction the action with the highest predicted reward (Beygelzimer and Langford 2009). However, the estimator can be misspecified by the partial observability of the environment (Sachdeva, Su, and Joachims 2020). To address these issues, domain adaptation methods are proposed to improve the regression model, where the source domain is observational data and the target domain is a randomized trial (Johansson, Shalit, and Sontag 2016; Gao et al. 2021). Despite their success, DMs do not directly apply to our multi-label classification setting as they ignore inter-label dependencies and can fail to generate action combinations with the arg-max operation.

CRM, also known as batch learning from bandit feedback, optimizes the policy model by maximizing its reward estimated with a counterfactual risk estimator (Dudík, Langford, and Li 2011; Swaminathan and Joachims 2015; Johansson, Shalit, and Sontag 2016; Joachims, Swaminathan, and De Rijke 2018; Swaminathan et al. 2017), while requires the strong assumption that the logging policy has full support for every policy in the policy space. With deficient support common in real-world systems, vanilla methods suffer from extrapolation error, learning arbitrarily bad

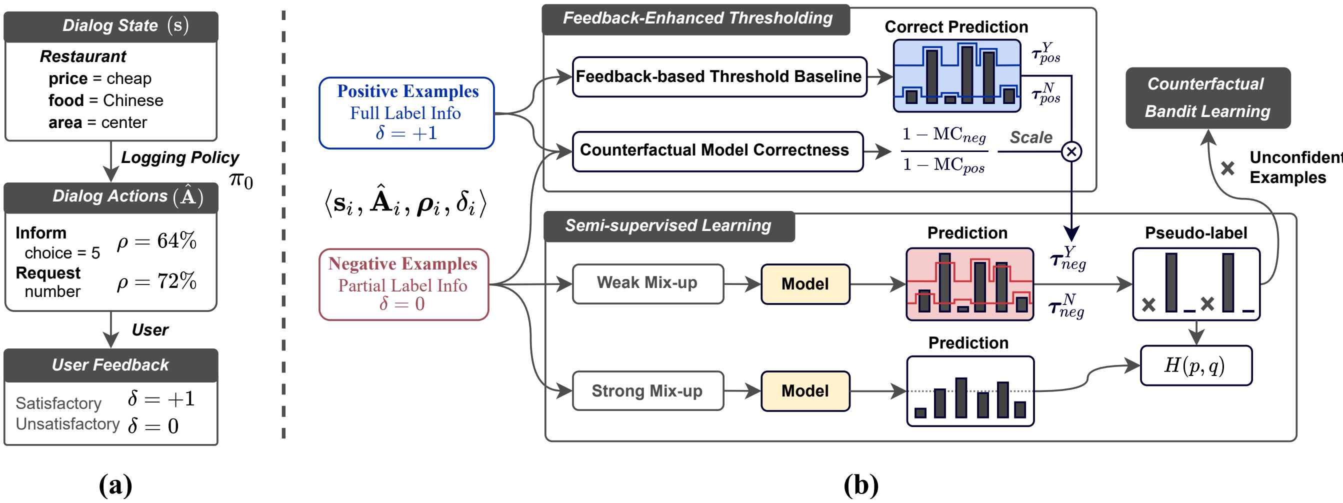

Figure 2: (a) Data logging procedure. During online interaction, an online agent π_0 (i.e., logging policy) responds to the dialog state $\mathbf{s}$ that summarizes the dialog history with multiple dialog actions $\hat{\mathbf{A}}$. We collect $(\mathbf{s}, \hat{\mathbf{A}})$ together with the corresponding prediction probability $\boldsymbol{\rho}$ and user feedback δ as logged bandit data. (b) Diagram of BanditMatch. The user feedback examples are first fed into the current policy to learn the feedback-enhanced thresholds. Specifically, the threshold baselines are set based on the average of probabilities of correctly predicted classes in positive examples. The feedback-enhanced thresholds for the negative examples are derived by scaling the baselines by comparing the model correctness for negative and positive examples. Then, BanditMatch partitions the negative examples into confident and unconfident ones based on the thresholds. Finally, the positive examples and confident negative examples are leveraged by SSL to learn from full label feedback while the rest unconfident ones by bandit learning to fully exploit the feedback information and, in turn, enhance pseudo-labeling.

estimates of the value of responses not contained in the data (Jaques et al. 2020). To address this problem, Sachdeva, Su, and Joachims (2020) proposes regression extrapolation that estimates the reward for under-explored examples with a learned model. Just as DM methods, it does not directly apply to multi-label classification. Policy space restriction methods are derived from offline RL, restricting the model from staying close to its own logging policy through KL-control (Jaques et al. 2020; Sachdeva, Su, and Joachims 2020; Zhang et al. 2021b). However, KL-control depends on the logging policy quality and does not work well for low-quality ones. In comparison, we address the extrapolation error with pseudo-labeling that improves the logging policy step-by-step. It works for low-quality logging policies and alleviates the cold start problem.

Semi-Supervised Learning. Semi-supervised learning is a mature field with a vast diversity of approaches. We focus on pseudo-labeling methods that use hard artificial labels converted from model predictions, and refer the reader to (Yang et al. 2021) for broader introductions. Specifically, the confidence-based thresholding, which uses unlabeled data only when the predictions are sufficiently confident, is presented in state-of-the-art methods UDA (Xie et al. 2020) and FixMatch (Sohn et al. 2020), where a fixed threshold is chosen to determine confident examples. In addition, several works have investigated the dynamic thresholds (Xu et al. 2021; Zhang et al. 2021a). In our work, it is the first attempt to learn an adaptive confidence threshold from the user feedback data. Besides, the unconfident examples are also leveraged to improve the model through counterfactual bandit learning and, in turn, enhance pseudo-labeling.

3 Problem Formulation

We regard MADPL as a multi-label classification task. Given dialog state $\mathbf{s}_i$, we want to predict a macro system action $\mathbf{A}_i$ that contains several atomic dialog actions. Following (Li, Kiseleva, and de Rijke 2020), each atomic dialog action is a concatenation of domain name, action type, and slot name, e.g., "hotel-inform-area".

Unlike supervised learning with expert examples $(\mathbf{s}_i, \mathbf{A}_i)$, we want to improve MADPL with logged user feedback $(\mathbf{s}_i, \hat{\mathbf{A}}_i, \boldsymbol{\rho}_i, \delta_i)$. $\hat{\mathbf{A}}_i$ is the historical prediction made by a logging policy π_0 for input $\mathbf{s}_i$ with probabilities $\boldsymbol{\rho}_i = \pi_0(\hat{\mathbf{A}}_i \mid \mathbf{s}_i)$, and $\delta_i \in \{0, +1\}$ is the user feedback received for that prediction. See Fig. 2 for the data logging procedure. We denote the data with positive and negative feedback as *positive examples* and *negative examples*, respectively. Note that the logged user feedback provides only partial information limited to the actions taken by π_0 and excludes all other actions the system could have taken. While positive examples can be viewed as fully labeled ones as in traditional supervised learning, the ground-truth labels for negative examples are typically unknown.

4 Algorithm

We propose BanditMatch to address the MADPL task. BanditMatch consists of three modules, i.e., the *feedback-enhanced thresholding* (FET) module, the *semi-supervised* *learning* (SSL) module, and the *counterfactual bandit learning* (CBL) module, as shown in Fig. 2. Unlike straightforward bandit learning from partial label feedback for all the negative examples, we propose to learn from full label feedback for the negative examples with high-confident predictions through pseudo-labeling. Specifically, the bandit data is partitioned into three parts: fully labeled positive examples D_L, pseudo-labeled confident negative examples D_P, and partially labeled unconfident negative examples D_U. D_P and D_U are split by comparing current prediction probability with the adaptive decision boundaries learned by FET. We further propose a hybrid learning objective to fully explore all the available label feedback for updating model parameters, where the positive examples D_L and negative confident examples D_P are used to learn through SSL, while the rest negative unconfident examples D_U through CBL.

4.1 Feedback-Enhanced Thresholding

Recent SSL algorithms (Sohn et al. 2020; Xie et al. 2020) select high-confidence unlabeled data with a fixed confidence threshold for each class. However, the learning status varies for different dialog scenarios due to their various learning difficulty. A fixed confidence threshold that treats all examples equally can be an over-high standard for under-learned examples and an over-low standard for over-learned ones. This may lead to eliminating examples with correct pseudo labels and selecting those wrong ones. Noticing this weakness, we propose the *feedback-enhanced thresholding* (FET) module to evaluate model performance with the feedback data and adjust the confidence threshold for each individual class at each single training step.

Ideally, we want to obtain negative examples correctly predicted by the updated policy π, and compute the threshold on the negative examples by averaging the corresponding confidence scores. Though appealing, this is impractical since the ground-truth labels for negative examples are unavailable. To address this issue, we propose to estimate such average confidence scores with correctly predicted positive examples. Intuitively, the better the model performs, the more credible its predictions are, and the lower the confidence threshold that can be set for it. Considering an extreme case, we can set a low threshold (e.g., 50%) for an always-correct oracle policy because its predictions are 100% reliable. In other words, the confidence thresholds for specific examples are negatively related to the correctness of the policy on those examples. Therefore, taking thresholds on correctly predicted positive examples $\boldsymbol{\tau}_{pos}$ as baselines, we propose to obtain the thresholds on negative examples $\boldsymbol{\tau}_{neg}$ by scaling the baselines with model correctness $\mathrm{MC}(\cdot)$, as:

$$\boldsymbol{\tau}^{Y}_{neg} \triangleq \boldsymbol{\tau}^{Y}_{pos}\frac{1-\mathrm{MC}_{neg}(\pi)}{1-\mathrm{MC}_{pos}(\pi)}$$

$$\boldsymbol{\tau}^{N}_{neg} \triangleq \vec{1}-(\vec{1}-\boldsymbol{\tau}^{N}_{pos})\frac{1-\mathrm{MC}_{neg}(\pi)}{1-\mathrm{MC}_{pos}(\pi)}$$

where $\vec{1}$ denotes the one-vector. Note we address a multi-label classification problem where class presence is independent so both positive and negative labels are necessary for training. Thus, two sets of thresholds $\boldsymbol{\tau}^{Y(N)}_{*} = (\tau^{Y(N)}_{*,1}, \ldots, \tau^{Y(N)}_{*,C})$ are used to select (ban) a specific dialog action a_c.

Specifically, we first apply the updated policy π on positive examples D_L and obtain the correctly predicted ones $D_T = \{(\mathbf{s}_i, \hat{\mathbf{A}}_i), i = 1, \ldots, N_T\}$. The thresholds for positive examples are then calculated by:

$$\tau^{Y}_{pos,c} \triangleq \frac{1}{N^{gc}_{T}}\sum_{i=1}^{N_T}\mathbb{1}(a_c \in \hat{\mathbf{A}}_i)\pi(a_c|\mathbf{s}_i)$$

$$\tau^{N}_{pos,c} \triangleq \frac{1}{N^{lc}_{T}}\sum_{i=1}^{N_T}\mathbb{1}(a_c \notin \hat{\mathbf{A}}_i)\pi(a_c|\mathbf{s}_i)$$

where N^{gc}_{T} (resp. N^{lc}_{T}) is the number of correct predictions that contain (resp. exclude) the c-th dialog action, $\mathbb{1}(\cdot)$ is the indicator function. We further construct the model correctness with its normalized expected feedback estimated from the bandit data, i.e.,

$$\mathrm{MC}_{pos}(\pi) \triangleq \frac{1}{|D_T|}\sum_{x_i \in D_T}\sum_{\hat{a}_j \in \hat{\mathbf{A}}_i}\mathrm{Attr}_{pos}(\hat{a}_j, \hat{\mathbf{A}}_i)\frac{\pi(\hat{a}_j|\mathbf{s}_i)}{\pi_0(\hat{a}_j|\mathbf{s}_i)}$$

$$\mathrm{MC}_{neg}(\pi) \triangleq \frac{1}{|B_N|}\sum_{x_i \in B_N}\sum_{\hat{a}_j \in \hat{\mathbf{A}}_i}\mathrm{Attr}_{neg}(\hat{a}_j, \hat{\mathbf{A}}_i)\frac{1-\pi(\hat{a}_j|\mathbf{s}_i)}{1-\pi_0(\hat{a}_j|\mathbf{s}_i)}$$

where B_N are the negative examples in the current batch. We use attribution functions to determine which individual dialog action caused the error in the predicted $\hat{\mathbf{A}}_i$. Specifically, for positive examples we simply average over the predicted dialog actions as they contribute equally to the success, i.e., $\mathrm{Attr}_{pos}(\cdot, \hat{\mathbf{A}}_i) \triangleq 1/|\hat{\mathbf{A}}_i|$. For negative examples, we follow the idea that the policy might be self-aware of some mistakes and distributes the feedback proportional to the uncertainty reflected in the propensities (Falke and Lehnen 2021), i.e.,

$$\mathrm{Attr}_{neg}(\hat{a}_j, \hat{\mathbf{A}}_i) \triangleq \frac{\pi_0(\hat{a}_j|\mathbf{s}_i)}{\sum_{a^*_k \in \hat{\mathbf{A}}_i}\pi_0(a^*_k|\mathbf{s}_i)}$$

4.2 Feedback-Enhanced Semi-Supervised Multi-Action Dialog Policy Learning

Given the feedback-enhanced thresholds, we are now able to perform semi-supervised learning and learn from full label information with the fully labeled positive examples D_L and pseudo-labeled confident negative examples D_P. To make the most of all the available label information, we propose to improve the pseudo-labeling model with the rest unconfident negative examples D_U through bandit learning with CRM. A KL control method is further applied to prevent the policy from choosing unrealistic dialog action combinations. Unlike prior SSL algorithms (Sohn et al. 2020; Zhang et al. 2021a), we do not pre-split the data examples but determine them in each batch adaptively during training.

Semi-supervised Learning We select FixMatch (Sohn et al. 2020), a simple yet effective SSL example, as the base of the proposed SSL algorithm. FixMatch first generates pseudo-labels using the model's predictions on weakly augmented unlabeled examples, where a fixed threshold is set to select high-confidence predictions. The model is then trained to predict the pseudo-label when fed a strongly-augmented version of the same example. For our SSL algorithm, we extend FixMatch to a multi-label classification setting with dialog states as training examples and apply the feedback-enhanced thresholds instead of fixed ones.

Specifically, following FixMatch, our SSL consists of two binary cross-entropy loss terms $\mathcal{L}_L$ and $\mathcal{L}_P$, which are applied to positive examples and confident negative examples, respectively. Specifically, $\mathcal{L}_L$ is the standard binary cross-entropy loss on weakly augmented labeled examples, as:

$$\mathcal{L}_L = \frac{\sum_{i=1}^{B} \mathbb{1}(\delta_i = 1) H(\hat{\mathbf{A}}_i, \pi(\mathbf{A} \mid \omega(\mathbf{s}_i, \alpha_w)))}{\sum_{i=1}^{B} \mathbb{1}(\delta_i = 1)}$$

where B is the batch size, and ω performs mix-up operation on $\mathbf{s}_i$. Unlike pictures or natural language sentences, dialog states are usually represented with a dictionary object and encoded with one-hot vectors (Takanobu, Zhu, and Huang 2019; Li, Kiseleva, and de Rijke 2020). We thus consider mix-up (Zhang et al. 2017) as datatype-agnostic data augmentation for consistency regularization. It mixes the training examples as:

$$\mathbf{s}_i = \lambda \mathbf{s}_i + (1-\lambda)\mathbf{s}_j$$

where λ is sampled from a Beta distribution, i.e., $\lambda = \max(\texttt{Beta}(\alpha_w, \alpha_w), 1 - \texttt{Beta}(\alpha_w, \alpha_w))$, and α_w is the shape parameter of the distribution for weak augmentation.

Unlike the multi-class classification setting in (Sohn et al. 2020) that accepts or rejects a whole example, the feedback-enhanced thresholds filter a set of confidently accepted (rejected) atomic dialog actions (i.e., confident classes) for each example. By viewing multi-label classification with $|\mathbf{A}|$ classes as $|\mathbf{A}|$ binary classifications, we obtain a pseudo label for the c-th class with the i-th weakly-augmented example as $\hat{q}_{i,c} = \mathbb{1}(\pi(a_c \mid \omega(\mathbf{s}_i, \alpha_w)) > 0.5)$, and enforce the cross-entropy loss on confident classes as:

$$\mathcal{L}_P = \frac{\sum_{i=1}^{B} \sum_{c=1}^{C} \text{Conf}_{i,c} H(\hat{q}_{i,c}, \pi(a_c \mid \omega(\mathbf{s}_i, \alpha_s)))}{\sum_{i=1}^{B} \sum_{c=1}^{C} \text{Conf}_{i,c}}$$

where α_s is the shape parameter of the Beta distribution for strong augmentation. The confidence indicator is calculated by comparing the prediction of weakly augmented example and the feedback enhanced thresholds, i.e., $\text{Conf}_{i,c} = \mathbb{1}(\delta_i = 0 \wedge \pi(a_c \mid \omega(\mathbf{s}_i, \alpha_w)) \notin [\tau^N_{neg,c}, \tau^Y_{neg,c}])$.

Counterfactual Bandit Learning For the unconfident examples, the cross-entropy loss cannot be used to guide the training process since the predictions can be erroneous. We thus apply CRM to improve the pseudo-labeling model through counterfactual bandit learning. The goal of learning is to find a policy π that maximizes the expected payoff, as:

$$R(\pi) = \mathbb{E}_{\mathbf{s} \sim \texttt{Pr}(S)} \mathbb{E}_{\mathbf{A} \sim \texttt{Pr}(\mathbb{A}|\mathbf{s})} [\delta(\mathbf{s}, \mathbf{A})]$$

Specifically, we aim to minimize the negative expected feedback estimated with the pseudoinverse estimator (PI) (Swaminathan et al. 2017), defined as:

$$\mathcal{L}_B = -\frac{\sum_{i=1}^{B} \delta_i \cdot (1 + \sum_{c=1}^{C} \text{Unconf}^+_{i,c} \cdot (\frac{\pi(a_c|\mathbf{s}_i)}{\rho_{i,c}} - 1))}{\sum_{i=1}^{B} \sum_{c=1}^{C} \text{Unconf}^+_{i,c}}$$

where $\text{Unconf}^+_{i,c}$ is an indicator function that filters positive examples besides unconfident negative ones to avoid a degenerate training objective (Swaminathan and Joachims 2015). Notably, the bandit learning process can boost model performance in the early training stage when confident examples are rare and effectively improve pseudo-labeling.

KL Control from the Logging Policy The joint bandit learning and semi-supervised learning allow a controlled exploration of state-action space. However, it still can generate unrealistic action combinations. To address this issue, we directly incorporate knowledge of the logging policy, which imitates expert dialog action combinations, into the training process through KL control (Jaques et al. 2020), i.e.,

$$\mathcal{L}_K = D_{KL}[\pi(\mathbf{s}) || \pi_0(\mathbf{s})]$$

where $D_{KL}[q||p] = \sum_x q(x)(\log q(x) - \log p(x))$ calculates the KL divergence between probability distributions.

Overall Objective Function Our proposed BanditMatch algorithm optimizes end-to-end with the overall loss as the weighted sum of four losses:

$$\mathcal{L} = \mathcal{L}_{\mathcal{L}} + \lambda_p \mathcal{L}_{\mathcal{P}} + \lambda_b \mathcal{L}_{\mathcal{B}} + \lambda_k \mathcal{L}_{\mathcal{K}}$$

where λ_p, λ_b, and λ_k are hyper-parameters to balance each term's intensity. We find that simply setting them to 1 can lead to proper performance through experimental evaluation.

5 Experiments

5.1 Settings

Data We use MultiWOZ 2.0 (Budzianowski et al. 2018), a large-scale multi-domain benchmark dataset, to validate the effectiveness of our method and apply the agenda-based user simulator as the interaction environment to evaluate the generalization ability.

To simulate MADPL from logged user feedback with the above dataset, we follow Falke and Lehnen (2021) and consider a simplified scenario where the user always provides correct feedback. The user feedback δ_i is determined by comparing the logging policy prediction and the ground truth. Specifically, we split the train set into $p\%$ labeled data (D_S), on which the logging policy is trained with supervised learning, and use the rest to create bandit data (D_B). For each state s_i in D_B, we follow the multi-label classification setting and take as agent prediction the actions whose prediction probability is greater than 0.5 (Rizve et al. 2021). The user feedback $\delta_i = 1(0)$ if the predicted labels do (not) match the ground truth. Note that δ_i is set to 0 rather than -1 for negative feedback to avoid a degenerate training objective (Swaminathan and Joachims 2015).

In real-world scenarios, the user feedback can be captured by an extra module that detects user behaviors such as paraphrasing, retries, etc (Falke et al. 2020). The corresponding last system responses are then determined as positive (negative) if there are no (any) detected negative behaviors. The feedback can be wrong due to the inaccuracy of the detection and the randomness of user behaviors. Such a noisy user feedback setting has been explored by Agarwal and Manwani (2021), where the idea of unbiased estimation of the feedback can be further integrated to improve BanditMatch.

Category	Agent	Turn	Match	Inform Rec	**Inform F1**	**Success%**
SL	Full SL	11.46 ±0.56	0.68 ±3.9%	0.81 ±3.2%	**0.81** ±2.1%	67.3 ±3.69
	Logging Policy	13.65 ±0.72	0.61 ±7.1%	0.73 ±4.9%	0.79 ±3.2%	54.3 ±5.47
RL	PPO [1]	12.9	-	-	-	33.8
	ALDM[1]	14.9	-	-	-	36.4
	GDPL[1]	11.8	-	-	-	50.2
SSL	FixMatch	14.01 ±2.89	0.56 ±15.8%	0.68 ±18.3%	0.69 ±8.5%	50.3 ±19.3
	Act-VRNN [1]	**8.4**	-	-	-	72.4
CRM	IPS	13.60 ±0.14	0.79 ±0.4%	0.84 ±0.4%	0.6 ±0.3%	71.3 ±0.89
	+ KL control	11.86 ±0.09	0.80 ±1.0%	**0.89** ±0.5%	0.67 ±0.5%	75.5 ±0.95
	BanditNet	14.37 ±0.07	0.8 ±0.3%	0.87 ±0.4%	0.53 ±0.2%	74.3 ±0.54
	+ KL control	13.10 ±0.06	**0.83** ±0.3%	0.87 ±0.2%	0.6 ±0.2%	75.8 ±0.04
SSL+CRM	**BanditMatch**	10.32 ±0.71	0.79 ±2.94%	**0.89** ±2.1%	0.76 ±2.2%	**76.7** ±2.83
	- MC scale	10.45 ±0.44	0.76 ±4.5%	0.88 ±1.8%	0.75 ±2.2%	75.6 ±2.36
	- FET	10.64 ±0.29	0.77 ±6.9%	0.88 ±2.8%	0.73 ±2.1%	74.4 ±4.76
	- CBL	11.54 ±1.68	0.69 ±10.9%	0.81 ±8.5%	0.79 ±2.0%	65.9 ±10.7
	- KL control	10.2 ±0.39	0.76 ±1.7%	0.87 ±3.0%	0.72 ±1.8%	73.1 ±2.7
	- all	11.5 ±1.02	0.69 ±5.2%	0.81 ±3.8%	0.78 ±1.1%	64.5 ±4.3

Table 1: Interactive evaluation results with 10% annotated examples. We simulate 1,000 dialogs per run and report the mean and standard deviation over 5 runs. We focus on comparing Inform F1 (low Inform F1 indicates dialog redundancy) and Success.

Baselines We compare BanditMatch with state-of-the-arts: 1) RL methods: PPO (Schulman et al. 2017), ALDM (Liu and Lane 2018), and GDPL (Takanobu, Zhu, and Huang 2019); 2) CRM methods: IPS (Swaminathan and Joachims 2015) and BanditNet (Joachims, Swaminathan, and De Rijke 2018); and 3) SSL methods: Act-VRNN (Huang et al. 2020) and FixMatch (Sohn et al. 2020), under the fully SL-based skyline (Li, Kiseleva, and de Rijke 2020).

The SL-based skyline is trained on all the trainset examples. Other models are further fine-tuned based on the logging policy. Specifically, the RL methods are fine-tuned with the same agenda-based user simulator for evaluation, while the rest methods with the bandit data D_B.

Evaluation Metrics We perform the automatic evaluation on task completion quality through dialog interaction with the user simulator. We use *Match*, *Turn*, *Success*, *Inform Recall*, and *Inform F1* as evaluation metrics: *Match* evaluates whether the booked entities match the indicated constraints (e.g., the cheapest restaurant in town). *Inform Recall* and *Inform F1* evaluate whether all the requested information (e.g., hotel address) has been provided. *Turn* and *Success* evaluate the efficiency and level of task completion of dialog agents, respectively. Specifically, a dialog is successful if all the requested information is provided (i.e., *Inform Recall* $=1$) and all the entities are correctly booked (i.e., *Match* $=1$).

Note that the partial user feedback information can result in successful but highly redundant conversations (see Introduction). For example, the agent may inform the user about tons of details about a hotel when the user only asks for its address. Such redundancy is reflected by *Inform F1*, which we thus consider as a critical metric besides *Success*.

[1]Results reused from (Huang et al. 2020)

5.2 Algorithm Evaluation

Overall Comparisons. Table 1 shows the performance of different methods. We observe that our method achieves the highest performance in the task success by $0.9\% \sim 5.4\%$ while achieving a significant $9\% \sim 23\%$ increase in Inform F1 score. This demonstrates that BanditMatch can effectively explore more state-action space to resist extrapolation error and generate more concise responses through pseudo-labeling confident examples. Compared with the SSL-based FixMatch, BanditMatch reaches a substantially better and stable performance in all evaluation metrics on account of the proper use of bandit user feedback. Besides, BanditMatch generally outperforms the Full SL method (which uses full label information) but fails to reach the same inform F1 score. This is because SL can over-fit specific dialog action combinations from limited demonstrations and ignore all the other possible valid responses that the agent could have taken, while CRM allows BanditMatch to explore unseen combinations.

Ablation Study. To gain deeper insight into the contributions of different components involved in our approach, we conduct ablation studies by considering the following variants: 1) a variant without MC scaling and directly uses SL-based minimum thresholds; 2) a variant without FET and applies fixed pre-defined thresholds; 3) a variant without CBL for unconfident examples; 4) a variant without KL control; 5) a variant without all the proposed modules and strategies, i.e., a FixMatch model with additional labeled examples (logged positive examples). We also report CRM method variants to make a fair comparison over KL control. The results are merged in Table 1.

We see that BanditMatch consistently outperforms its ablated variants. Specifically, we find that: 1) The FET strat-

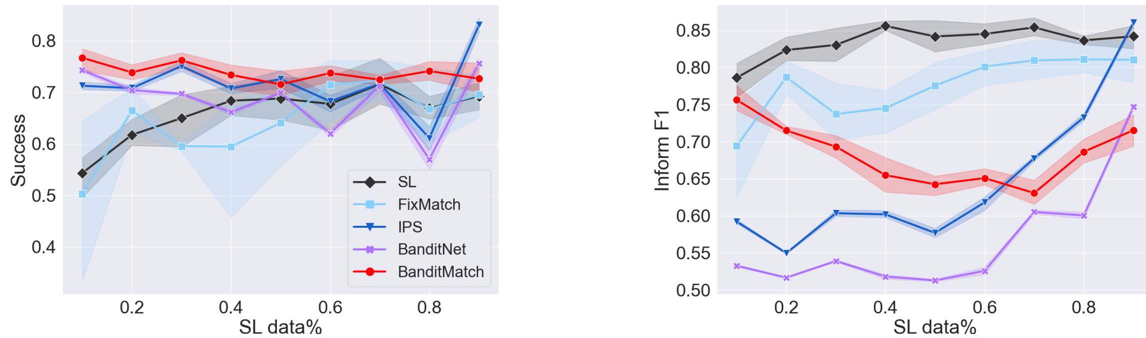

Figure 3: Agent performance under different percentage of expert examples (SL data%).

Dialog pair	Win	Lose	Tie
Ours vs SL	56.6	23.9	19.5
Ours vs BanditNet	60.3	25.9	13.8
Ours vs FixMatch	52.6	28.0	19.4

Table 2: Human evaluation results (mean over 10 judges).

[User]: I'd like to find an expensive restaurant in the centre.
[BanditNet]: There are 4 available restaurants. How about ugly duckling? That is an expensive Chinese restaurant in the centre area, located at ... phone number as ... postcode is .... Would you like to make a reservation? What time would you like the reservation for? Is there anything else I can help you with today?
[Ours]: Found 4 such places. What type of food do you like?

Table 3: Case system response.

egy (including MC scale) contributes to overall performance gain (especially on Inform F1 score) by increasing the number of confident examples. 2) CBL plays a key role in performance gain and model stabilization as it exploits the feedback from unconfident examples to enhance pseudo-labeling by reducing the confirmation bias. 3) KL control also works by improving Inform F1 score like pseudo-labeling while less effective. In addition, the vanilla combination of CRM and KL control still suffers from severe extrapolation error.

Human Evaluation. Automatic evaluation only measures part of the agent's performance (e.g., Success for the level of task completion). It may fail to consider other aspects for assisting real users (e.g., redundant or inappropriate reply). Thus, we conduct a human study to fully evaluate system performance. Our BanditMatch method is compared with three methods: SL, FixMatch, and BanditNet. Following prior works (Takanobu, Zhu, and Huang 2019; Li, Kiseleva, and de Rijke 2020), for each comparison pair, we randomly example 100 user goals and present the generated dialog examples from the two agents. Dialog actions are translated into human-readable utterances with a rule-based language generation module. We show each example to 10 judges to select the dialogs that provide a better user experience.

In Table 2 and 3, we see that our method outperforms the baseline methods. Especially compared to BanditNet, though comparable in success rate, our BanditMatch method provides high-quality service by generating concise and informative system responses.

Discussion. In what follows, we study how many expert examples are enough for proper MADPL from logged user feedback. Given the MultiWOZ dataset, we build 10 random splits based on the percentage of expert examples. The expert examples are used to initiate the logging policy, and the rest to create bandit data. We report the results in Fig. 3.

We observe that BanditMatch achieves an improved and stable success rate across different SL data percentages. The Inform F1 score first decreases and then goes up with the growth of SL data percentage. We speculate that the reason is that with the decrease of bandit data, the estimation of FET tends to be biased and increases the percentage of unconfident data, making the performance more influenced by CBL. Note that we can make a trade-off between conciseness and information richness by adjusting the weight for the CBL loss. With respect to vanilla CRM methods, we see substantially better inform F1 score in the high SL percentage region. In other words, CRM methods benefit from the SL stage and fail without a favorable logging policy.

6 Conclusion and Future Work

This paper studies the problem of multi-action dialog policy learning from logged user feedback. To address this problem, we present a feedback-enhanced semi-supervised algorithm BanditMatch, which complements the partial labels through pseudo-labeling. Benefiting from fully utilizing all the label information through joint semi-supervised and counterfactual bandit learning, BanditMatch is able to alleviate the extrapolation error and improve model performance towards unseen dialog flows while generating more concise and informative responses. Extensive experiments demonstrate the superiority of our proposed method. In the future, to improve data privacy for the user feedback, we plan to explore the federated bandit learning scenario where the personal user feedback is stored locally in each client and not exchanged or transferred.

Acknowledgments

This work was supported in part by National Key R&D Program of China (2021YFB1715600), National Natural Science Foundation of China (61902305, 62272372, 61922067), and MoE-CMCC "Artifical Intelligence" Project (MCM20190701).

References

Agarwal, M.; and Manwani, N. 2021. Learning Multiclass Classifier Under Noisy Bandit Feedback. In *PAKDD*, 448–460. Springer.

Beygelzimer, A.; and Langford, J. 2009. The offset tree for learning with partial labels. In *SIGKDD*, 129–138.

Budzianowski, P.; Wen, T.-H.; Tseng, B.-H.; Casanueva, I.; Ultes, S.; Ramadan, O.; and Gašić, M. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv:1810.00278*.

Dudík, M.; Langford, J.; and Li, L. 2011. Doubly robust policy evaluation and learning. *arXiv:1103.4601*.

Falke, T.; Boese, M.; Sorokin, D.; Tirkaz, C.; and Lehnen, P. 2020. Leveraging user paraphrasing behavior in dialog systems to automatically collect annotations for long-tail utterances. In *ICCL*, 21–32.

Falke, T.; and Lehnen, P. 2021. Feedback Attribution for Counterfactual Bandit Learning in Multi-Domain Spoken Language Understanding. In *EMNLP*, 1190–1198.

Gao, R.; Biggs, M.; Sun, W.; and Han, L. 2021. Enhancing Counterfactual Classification via Self-Training. *arXiv:2112.04461*.

Gordon-Hall, G.; Gorinski, P. J.; and Cohen, S. B. 2020. Learning dialog policies from weak demonstrations. *arXiv:2004.11054*.

Horton, H. 2016. Microsoft deletes "teen girl" ai after it became a hitler-loving sex robot in 24 hours. Telegraph UK.

Huang, X.; Qi, J.; Sun, Y.; and Zhang, R. 2020. Semi-supervised dialogue policy learning via stochastic reward estimation. *arXiv:2005.04379*.

Jaques, N.; Shen, J. H.; Ghandeharioun, A.; Ferguson, C.; Lapedriza, A.; Jones, N.; Gu, S. S.; and Picard, R. 2020. Human-centric dialog training via offline reinforcement learning. *arXiv:2010.05848*.

Jhunjhunwala, M.; Bryant, C.; and Shah, P. 2020. Multi-Action Dialog Policy Learning with Interactive Human Teaching. In *SIGdial*, 290–296.

Joachims, T.; Swaminathan, A.; and De Rijke, M. 2018. Deep learning with logged bandit feedback. In *ICLR*.

Johansson, F.; Shalit, U.; and Sontag, D. 2016. Learning representations for counterfactual inference. In *ICML*, 3020–3029. PMLR.

Kreutzer, J.; Khadivi, S.; Matusov, E.; and Riezler, S. 2018. Can neural machine translation be improved with user feedback? *arXiv:1804.05958*.

Li, Z.; Kiseleva, J.; and de Rijke, M. 2020. Rethinking supervised learning and reinforcement learning in task-oriented dialogue systems. *arXiv:2009.09781*.

Li, Z.; Lee, S.; Peng, B.; Li, J.; Kiseleva, J.; de Rijke, M.; Shayandeh, S.; and Gao, J. 2020. Guided dialog policy learning without adversarial learning in the loop. *arXiv:2004.03267*.

Liu, B.; and Lane, I. 2018. Adversarial learning of task-oriented neural dialog models. *arXiv:1805.11762*.

Rizve, M. N.; Duarte, K.; Rawat, Y. S.; and Shah, M. 2021. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. *arXiv:2101.06329*.

Sachdeva, N.; Su, Y.; and Joachims, T. 2020. Off-policy bandits with deficient support. In *SIGKDD*, 965–975.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv:1707.06347*.

Shu, L.; Xu, H.; Liu, B.; and Molino, P. 2019. Modeling Multi-Action Policy for Task-Oriented Dialogues. *arXiv:1908.11546*.

Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C. A.; Cubuk, E. D.; Kurakin, A.; and Li, C.-L. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *NIPS*, 33: 596–608.

Swaminathan, A.; and Joachims, T. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *JMLR*, 16(1): 1731–1755.

Swaminathan, A.; Krishnamurthy, A.; Agarwal, A.; Dudik, M.; Langford, J.; Jose, D.; and Zitouni, I. 2017. Off-policy evaluation for slate recommendation. *NIPS*, 30.

Takanobu, R.; Zhu, H.; and Huang, M. 2019. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. *arXiv:1908.10719*.

Xie, Q.; Dai, Z.; Hovy, E.; Luong, T.; and Le, Q. 2020. Unsupervised data augmentation for consistency training. *NIPS*, 33: 6256–6268.

Xu, Y.; Shang, L.; Ye, J.; Qian, Q.; Li, Y.-F.; Sun, B.; Li, H.; and Jin, R. 2021. Dash: Semi-supervised learning with dynamic thresholding. In *ICML*, 11525–11536. PMLR.

Yang, X.; Song, Z.; King, I.; and Xu, Z. 2021. A survey on deep semi-supervised learning. *arXiv:2103.00550*.

Zhang, B.; Wang, Y.; Hou, W.; Wu, H.; Wang, J.; Okumura, M.; and Shinozaki, T. 2021a. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *NIPS*, 34.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv:1710.09412*.

Zhang, S.; Zhao, J.; Wang, P.; Li, Y.; Huang, Y.; and Feng, J. 2022. "Think Before You Speak": Improving Multi-Action Dialog Policy by Planning Single-Action Dialogs. In *IJCAI*, 4510–4516.

Zhang, S.; Zhao, J.; Wang, P.; Xu, N.; Yang, Y.; Liu, Y.; Huang, Y.; and Feng, J. 2021b. Learning to Check Contract Inconsistencies. In *AAAI*, 14446–14453.

Zhang, Z.; Takanobu, R.; Zhu, Q.; Huang, M.; and Zhu, X. 2020. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences*, 1–17.