Advancing Semi-Supervised Task Oriented Dialog Systems by JSA Learning of Discrete Latent Variable Models

Yucheng Cai, Hong Liu, Zhijian Ou*
Speech Processing and Machine Intelligence Lab
Tsinghua University, Beijing, China
cyc22@mails.tsinghua.edu.cn
liuhong21@mails.tsinghua.edu.cn
ozj@tsinghua.edu.cn

Yi Huang, Junlan Feng
China Mobile Research Institute
Beijing, China
{huangyi,fengjunlan}
@chinamobile.com

Abstract

Developing semi-supervised task-oriented dialog (TOD) systems by leveraging unlabeled dialog data has attracted increasing interests. For semi-supervised learning of latent state TOD models, variational learning is often used, but suffers from the annoying high-variance of the gradients propagated through discrete latent variables and the drawback of indirectly optimizing the target log-likelihood. Recently, an alternative algorithm, called joint stochastic approximation (JSA), has emerged for learning discrete latent variable models with impressive performances. In this paper, we propose to apply JSA to semi-supervised learning of the latent state TOD models, which is referred to as JSA-TOD. To our knowledge, JSA-TOD represents the first work in developing JSA based semi-supervised learning of discrete latent variable conditional models for such long sequential generation problems like in TOD systems. Extensive experiments show that JSA-TOD significantly outperforms its variational learning counterpart. Remarkably, semi-supervised JSA-TOD using 20% labels performs close to the full-supervised baseline on MultiWOZ2.1.

1 Introduction

Task-oriented dialog (TOD) systems are designed to help users to achieve their goals through multiple turns of natural language interaction. The system needs to parse user utterances, track dialog states, query a task-related database (DB), decide actions and generate responses, and to do these iteratively across turns. The information flow in a task-oriented dialog is illustrated in Figure 1.

Recent studies recast such information flow in a TOD system as conditional generation of tokens and base on pretrained language models (PLMs) such as GPT2 (Radford et al., 2019) and T5 (Raffel et al., 2020) as the model backbone. Fine-tuning a

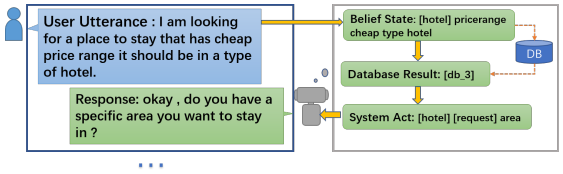

Figure 1: The information flow in a task-oriented dialog. Square brackets denote special tokens in GPT2.

PLM over annotated dialog datasets such as MultiWOZ (Budzianowski et al., 2018) via supervised learning has shown promising results (Hosseini-Asl et al., 2020; Peng et al., 2020; Yang et al., 2021; Liu et al., 2022), but requires manually labeled dialog states and system acts (if used).

Notably, there are often easily-available unlabeled dialog data such as in customer-service logs and online forums. This has motivated the development of semi-supervised leaning (SSL) for TOD systems, which aims to leverage both labeled and unlabeled dialog data. A broad class of SSL methods builds a latent variable model (LVM) of observations and labels and blends unsupervised and supervised learning. Unsupervised learning with a LVM usually maximizes the marginal log-likelihood, which is often intractable to compute. Variational learning (Kingma and Welling, 2014) introduces an auxiliary inference model and, instead, maximizes the evidence lower bound (ELBO) of the marginal log-likelihood. This approach of variational learning of LVMs has been studied for semi-supervised TOD systems such as in Jin et al. (2018); Zhang et al. (2020b); Liu et al. (2021); Li et al. (2021). Particularly, discrete latent variables are mostly used, since dialog states and system acts are often modeled as taking discrete values.

However, for variational learning of discrete latent variable models, the Monte-Carlo gradient estimator for the inference model parameter is known to have high-variance. Most previous studies use the Gumbel-Softmax trick (Jang et al., 2017) or the

*Corresponding author.

Proceedings of the SIGdial 2022 Conference, pages 456–467
Heriot-Watt University, Edinburgh, UK. 07-09, September, 2022

Straight-Through trick (Bengio et al., 2013) empirically, which in fact are biased estimators. Another drawback of variational learning is that it indirectly optimizes the lower bound of the target marginal log-likelihood, which leaves an uncontrolled gap between the target and the bound, depending on the expressiveness of the inference model.

Recently, an alternative algorithm, called joint stochastic approximation (JSA) (Xu and Ou, 2016; Ou and Song, 2020), has emerged for learning discrete latent variable models with impressive performances. JSA directly optimizes the marginal likelihood and completely avoids gradient propagation through discrete latent variables. In this paper, we propose to apply JSA to semi-supervised learning of the latent state TOD models, which is referred to as JSA-TOD. We develop recursive turn-level Metropolis Independence Sampling (MIS) to enable the successful application of JSA, which needs posterior sampling of the latent states from the whole dialog session. To our knowledge, JSA-TOD represents the first work in developing JSA based semi-supervised learning of discrete latent variable conditional models for such long sequential generation problems like in TOD systems.

Extensive experiments show that JSA-TOD significantly outperforms its variational learning counterpart in semi-supervised learning. Remarkably, semi-supervised JSA-TOD using 20% labels performs close to the supervised-only baseline using 100% labels on MultiWOZ2.1. The code and data are released at https://github.com/cycrab/JSA-TOD.

2 Related Work

2.1 Semi-Supervised TOD Systems

There are increasing interests in developing SSL methods for TOD systems, which aims to leverage both labeled and unlabeled data. Roughly speaking, there are two broad classes of SSL methods - the pretraining-and-finetuning approach and the latent variable modeling approach. With the development of pretrained language models such as GPT2 (Radford et al., 2019) and T5 (Raffel et al., 2020), the pretraining-and-finetuning approach based on backbones of PLMs has shown excellent performance for TOD systems (Hosseini-Asl et al., 2020; Yang et al., 2021; Lee, 2021).

Discrete latent variable models have been used for semi-supervised TOD systems (Jin et al., 2018; Zhang et al., 2020b)[1], initially based on LSTM architectures. Recently, discrete latent variable models based on PLMs have been studied in Liu et al. (2021), combining the strengths of PLMs and LVMs for semi-supervised TOD systems. However, previous studies all resort to variational methods for learning latent variable models, which suffers from the high-variance of the gradients propagated through discrete latent variables and the drawback of indirectly optimizing the target log-likelihood.

2.2 Joint Stochastic Approximation for Learning Latent Variable Models

Traditionally, variational methods minimize the "exclusive Kullback-Leibler (KL) divergence" $KL[p||q] \triangleq \int q \log \left(\frac{q}{p}\right)$, where p and q are shorthands for the true posterior (of the latent variable given the observation) and its approximation (also called the inference model) respectively, in learning a latent variable model. Recently, the JSA algorithm has been developed (Xu and Ou, 2016; Ou and Song, 2020), which proposes to minimize the "inclusive KL" $KL[p||q] \triangleq \int p \log \left(\frac{p}{q}\right)$, which has good statistical properties that makes it more appropriate for certain inference and learning problems, particularly for those using discrete latent variables. Similar idea has been studied in a concurrent and independent work (Naesseth et al., 2020). More investigations and extensions along this direction have been examined (Kim et al., 2020, 2022).

In Song and Ou (2020), JSA is applied to semi-supervised sequence-to-sequence learning, which consistently outperforms variational learning on two semantic parsing benchmark datasets. However, both generative model and inference model in (Song and Ou, 2020) are LSTM-based and much simpler than the ones in this work; its model complexity is similar to a single turn in a TOD system. Another difference is that this paper represents the first application of JSA in its conditional sequential version, since the latent state TOD model is a conditional sequential generative model.

[1]There are other previous studies of using discrete latent variable models in TOD systems, for example, Wen et al. (2017); Zhao et al. (2019); Bao et al. (2020). But most of them are mainly designed to improve response generation and diversity, instead of towards semi-supervised learning. See Zhang et al. (2020b); Liu et al. (2021) for more review of related work in latent variable models for dialogs.

3 Preliminary: Joint Stochastic Approximation (JSA)

Stochastic approximation (SA) refers to an important family of iterative stochastic optimization algorithms for stochastically solving a root finding problem, which has the form of expectations being equal to zeros (Robbins and Monro, 1951). Within the SA framework, the joint stochastic approximation (JSA) algorithm is recently developed (Xu and Ou, 2016; Ou and Song, 2020) for learning a broad class of latent variable models, particularly for learning models with discrete latent variables. Interestingly, JSA amounts to coupling an SA version of Expectation-Maximization (SAEM) (Delyon et al., 1999; Kuhn and Lavielle, 2004) with an adaptive Markov Chain Monte Carlo (MCMC) procedure. Based on JSA, the annoying difficulty of propagating gradients through discrete latent variables and the drawback of indirectly optimizing the target log-likelihood can be gracefully addressed.

Consider a latent variable generative model $p_\theta(z, x)$ for observation x and latent variable z, with parameter θ. Like in variational methods, JSA also jointly trains the target model $p_\theta(z, x)$ together with an auxiliary amortized inference model $q_\phi(z|x)$. The difference is that JSA directly maximizes w.r.t. θ the marginal log-likelihood and simultaneously minimizes w.r.t. ϕ the inclusive KL divergence $KL(p_\theta(z|x)||q_\phi(z|x))$ between the posterior and the inference model, pooled over the training dataset:

$$\begin{cases} \min_\theta \frac{1}{n}\sum_{i=1}^n \log p_\theta(x^{(i)}) \\ \min_\phi \frac{1}{n}\sum_{i=1}^n KL[p_\theta(z^{(i)}|x^{(i)})||q_\phi(z^{(i)}|x^{(i)})] \end{cases} \tag{1}$$

where the training dataset consists of n independent and identically distributed (IID) data-points $\{x^{(1)}, \cdots, x^{(n)}\}$.

The optimization problem Eq. (1) can be solved by setting the gradients to zeros and applying the SA algorithm to find the root for the resulting simultaneous equations, which has the exact form of expectations equal to zeros:

$$\begin{cases} \frac{1}{n}\sum_{i=1}^n E_{p_\theta(z^{(i)}|x^{(i)})}\left[\nabla_\theta \log p_\theta(x^{(i)}, z^{(i)})\right] = 0 \\ \frac{1}{n}\sum_{i=1}^n E_{p_\theta(z^{(i)}|x^{(i)})}\left[\nabla_\phi \log q_\phi(z^{(i)} \mid x^{(i)})\right] = 0 \end{cases} \tag{2}$$

The resulting JSA algorithm, as summarized in Algorithm 1, iterates Monte Carlo sampling and parameter updating. In each iteration, we draw a training observation $x^{(\kappa)}$ and then sample $z^{(\kappa)}$ through Metropolis Independence Sampling (MIS), with $p_\theta(z^{(\kappa)}|x^{(\kappa)})$ as the target distribution and $q_\phi(z|x^{(\kappa)})$ as the proposal:

1) Propose $z \sim q_\phi(z|x^{(\kappa)})$;

2) Accept $z^{(\kappa)} = z$ with probability

$$min\left\{1, \frac{w(z)}{w(\bar{z}^{(\kappa)})}\right\}$$

where $w(z) = \frac{p_\theta(z|x^{(\kappa)})}{q_\phi(z|x^{(\kappa)})} \propto \frac{p_\theta(z,x^{(\kappa)})}{q_\phi(z|x^{(\kappa)})}$ is the usual importance ratio between the target and the proposal distribution and $\bar{z}^{(\kappa)}$ denotes the cached latent state for observation $x^{(\kappa)}$.

Algorithm 1 The JSA algorithm

repeat

Monte Carlo sampling:

Draw κ over $1, \cdots, n$, pick the data-point $x^{(\kappa)}$ along with the cached $\bar{z}^{(\kappa)}$, and use MIS to draw $z^{(\kappa)}$;

Parameter updating:

Update θ by ascending: $\nabla_\theta \log p_\theta(z^{(\kappa)}, x^{(\kappa)})$;

Update ϕ by ascending: $\nabla_\phi \log q_\phi(z^{(\kappa)}|x^{(\kappa)})$;

until convergence

The JSA algorithm can be intuitively understood as a stochastic extension of the well-known EM algorithm (Dempster et al., 1977). Since the latent variable $z^{(\kappa)}$ is unknown for data-point $x^{(\kappa)}$, the Monte Carlo sampling step in JSA fills the missing value for $z^{(\kappa)}$ through sampling $p_\theta(z^{(\kappa)}|x^{(\kappa)})$, which is analogous to the E-step in EM. Then in the parameter updating step, $z^{(\kappa)}$ is treated as if being known, and used to optimize over θ and ϕ by performing gradient ascent using $\nabla_\theta \log p_\theta(z^{(\kappa)}, x^{(\kappa)})$ and $\nabla_\phi \log q_\phi(z^{(\kappa)}|x^{(\kappa)})$ respectively. This is analogous to the M-step in EM, but with the proposal q_ϕ being adapted as well. In summary, we could refer to the underlying mechanism of JSA as Propose, Accept/Reject, and Optimize (or, for short, the PARO mechanism), which establishes JSA as a simple, solid and effective approach to learning discrete latent variable models.

4 Method

4.1 Definition of Discrete Latent Variables in TOD systems

In a TOD system, let u_t denote the user utterance, b_t the dialog state, db_t the DB result, a_t the system

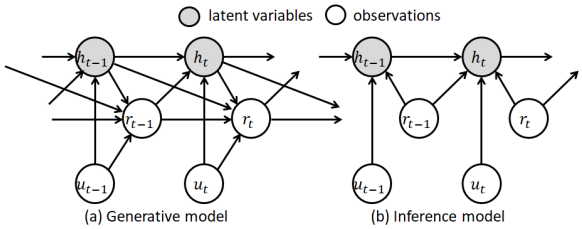

Figure 2: The probabilistic graphical model of Markov latent state generative model (a) and inference model (b) for TOD systems. u_t and r_t are user utterance and system response respectively. The latent variables $h_t = \{b_t, a_t\}$ are the concatenation of dialog state and system act, which specifically are represented by token sequences in our experiments.

act and r_t the delexicalized response, respectively, at turn t. In this work, all these variables are converted to token sequences, like in DAMD (Zhang et al., 2020a). As shown in Figure 1, the workflow for a TOD system is, for each dialog turn t, to generate b_t, a_t and r_t, given u_t and dialog history $u_1, r_1, \cdots, u_{t-1}, r_{t-1}$. The database result db_t is deterministically obtained by querying database using the predicted b_t, and thus could be omitted in the following probabilistic modeling of a TOD system for simplicity.

Let $h_t = \{b_t, a_t\}$ denote the concatenation of dialog state and system act. Specifically, dialog state b_t and system act a_t are represented by sequences of labels, for example, $[train]$ day $monday$ $[hotel]$ $pricerange$ $cheap$ and $[train]$ $[inform]$ $choice$ $departure$ $[request]$ $destination$, respectively. Notably, h_t's are observed in labeled dialogs, but they become latent variables in unlabeled dialogs in training and need to be generated in testing. With this definition of h_t's, latent variable models can be developed for TOD systems, which will be described shortly in the next subsection.

Remarkably, the above definition of latent variables as sequences of labels in this paper is similar to Zhang et al. (2020b); Liu et al. (2021). An important feature of such latent variables is that they are sensible and interpretable, which correspond to meaningful annotations according to the task knowledge. It is only in unlabeled dialogs that they become unobservable. This is different in nature from some other previous studies of using latent variables in TOD models (Wen et al., 2017; Zhao et al., 2019; Bao et al., 2020), where the latent variables are just assumed to be K-way categorical variables and learned in a purely data driven way.

4.2 A Probabilistic Latent State TOD Model

With the above introduction of latent variables and motivated by recent studies (Zhang et al., 2020b; Liu et al., 2022), the workflow of a TOD system could be described by a conditional sequential generative model with latent variables h_t's as follows for T turns, with parameter θ:

$$p_\theta(h_{1:T}, r_{1:T}|u_{1:T}) = \prod_{t=1}^{T} p_\theta(h_t, r_t|u_1, h_1, r_1, \cdots, u_{t-1}, h_{t-1}, r_{t-1}, u_t) \quad (3)$$

$$= \prod_{t=1}^{T} p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t) \text{(by Markov assumption)} \quad (4)$$

Here Eq. (3) and Eq. (4) could be collectively referred to as *latent state TOD models*, being non-Markov and Markov respectively. Eq. (3) represents non-Markov latent state models, which, with different further instantiations, are used in recent PLM-based TOD systems such as in Hosseini-Asl et al. (2020); Yang et al. (2021); Liu et al. (2021). In contrast, Eq. (4) makes the Markov assumption that the conditional generation of current h_t and r_t (when given u_t) depends on the dialog history only through h_{t-1} and r_{t-1} at the immediately preceding turn. Markov models have been employed in LSTM-based TOD systems such as in Lei et al. (2018); Zhang et al. (2020a); Zhang et al. (2020b). A recent study in Liu et al. (2022) revisits Markovian generative architectures (MGAs) for PLM backbones (GPT2 and T5) and shows their efficiency advantages in memory, computation and learning over non-Markov models.

4.3 Model Instantiation and Supervised Learning

In our experiments, we mainly consider MGA based latent state TOD systems (Liu et al., 2022), which are illustrated in Figure 2 as directed probabilistic graphical models. The conditional distribution $p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t)$ is instantiated as

$$p_\theta(b_t, a_t, r_t|b_{t-1}, r_{t-1}, u_t) \quad (5)$$

which is realized based on a GPT2 backbone in our experiments. The concatenation $b_{t-1} \oplus r_{t-1} \oplus u_t$ is used as the conditioning input, and the output $b_t \oplus a_t \oplus r_t$ is generated token-by-token in an autoregressive manner, where $\oplus$ denotes the concatenation of token sequences.

In order to perform unsupervised learning over unlabeled dialogs (to be detailed below), we introduce an inference model $q_\phi(h_{1:T}|u_{1:T}, r_{1:T})$ as follows to approximate the true posterior $p_\theta(h_{1:T}|u_{1:T}, r_{1:T})$:

$$q_\phi(h_{1:T}|u_{1:T}, r_{1:T}) = \prod_{t=1}^{T} q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t) \tag{6}$$

The conditional $q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t)$ is instantiated as

$$q_\phi(b_t, a_t|b_{t-1}, r_{t-1}, u_t, r_t) \tag{7}$$

which is realized based on a GPT2 backbone as well in our experiments.

When labeled dialog data are available, the supervised training of the latent state generative model p_θ in and inference model q_ϕ can be decomposed into turn-level teacher-forcing, since the latent states h_t's are known (labeled) for all turns and the model likelihoods decomposes over turns, as shown in Eq. (4) and Eq. (6).

4.4 JSA Learning over Unlabeled Dialogs

Suppose that we have n unlabeled dialogs $\left\{(u^{(i)}_{1:T_i}, r^{(i)}_{1:T_i})|i = 1, \cdots, n\right\}$, i.e., user utterances and system responses are available for each dialog, but without any annotations of the latent states. The training instances are indexed by the superscripts, and T_i denote the number of turns in the i-th training instance. The unsupervised learning of the latent state TOD model over such unlabeled data can be realized by applying the JSA algorithm, and more specifically its conditional version, to maximize the conditional marginal log-likelihood $\log p_\theta(r_{1:T}|u_{1:T})$.

The objective functions in JSA learning can be developed as follows, similar to Eq. (1):

$$\begin{cases} \min\limits_{\theta} \dfrac{1}{n}\sum\limits_{i=1}^{n} \log p_\theta(r^{(i)}_{1:T_i}|u^{(i)}_{1:T_i}) \\ \min\limits_{\phi} \dfrac{1}{n}\sum\limits_{i=1}^{n} KL[p_\theta(h^{(i)}_{1:T_i}|u^{(i)}_{1:T_i}, r^{(i)}_{1:T_i}) \\ \qquad\qquad ||q_\phi(h^{(i)}_{1:T_i}|u^{(i)}_{1:T_i}, r^{(i)}_{1:T_i})] \end{cases}$$

where we substitute observation x by $r_{1:T}$ and latent variable z by $h_{1:T}$, all conditioned on $u_{1:T}$.

Basically, JSA learning iterates Monte Carlo sampling and parameter updating, as outlined in Algorithm 1. In each iteration, we randomly pick a training instance $(u_{1:T}, r_{1:T})$ along with the cached latent state $\bar{h}_{1:T}$, and we need to draw a posterior sample $h_{1:T} \sim p_\theta(h_{1:T}|u_{1:T}, r_{1:T})$. Remarkably, it can be shown in Appendix A that for the posterior $p_\theta(h_{1:t}|u_{1:t}, r_{1:t})$ induced from the joint distribution in Eq. (4), the following recursion holds:

$$\begin{aligned} &p_\theta(h_{1:t}|u_{1:t}, r_{1:t}) \\ \propto &p_\theta(h_{1:t-1}|u_{1:t-1}, r_{1:t-1}) p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t) \end{aligned} \tag{8}$$

Based on such recursion, we can develop a recursive turn-level MIS sampler, as shown in Algorithm 2, which recursively runs MIS sampler turn-by-turn and finally obtains a valid posterior sample for the whole dialog session, i.e., $h_{1:T} \sim p_\theta(h_{1:T}|u_{1:T}, r_{1:T})$.

Suppose that we have obtained a sample for the previous t-1 turns, i.e., $h_{1:t-1} \sim p_\theta(h_{1:t-1}|u_{1:t-1}, r_{1:t-1})$. Then, we perform MIS sampling as follows, with $p_\theta(h_{1:t}|u_{1:t}, r_{1:t})$ as the target distribution and

$$p_\theta(h_{1:t-1}|u_{1:t-1}, r_{1:t-1}) q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t) \tag{9}$$

as the proposal distribution:

1) Propose $h'_t \sim q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t)$. Thus, $(h_{1:t-1}, h'_t)$ is a valid sample proposed from the proposal distribution as shown in Eq. (9);

2) Simulate $\xi \sim Uniform[0, 1]$ and let

$$h_t = \begin{cases} h'_t, & \text{if } \xi \leq min\left\{1, \dfrac{w(h_{1:t-1}, h'_t)}{w(h_{1:t-1}, \bar{h}_t)}\right\} \\ \bar{h}_t, & \text{otherwise} \end{cases} \tag{10}$$

where the importance ratio between the target and the proposal distribution

$$\begin{aligned} &w(h_{1:t-1}, h_t) \\ =&\frac{p_\theta(h_{1:t}|u_{1:t}, r_{1:t})}{p_\theta(h_{1:t-1}|u_{1:t-1}, r_{1:t-1}) q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t)} \\ \propto&\frac{p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t)}{q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t)} \end{aligned} \tag{11}$$

After we obtain the sampled latent state $h_{1:T}$ from Algorithm 2, we perform parameter updating, as outlined in Algorithm 1. The sampled latent state $h_{1:T}$ is treated as if being known, and we can calculate the gradients of $\log p_\theta(h_{1:T}, r_{1:T}|u_{1:T})$ and $\log q_\phi(h_{1:T}|u_{1:T}, r_{1:T})$ w.r.t. θ and ϕ according to Eq. (4) and Eq. (6) respectively, as if we calculate gradients in supervised training. Thanks

Algorithm 2 Recursive turn-level MIS sampler

Input: A T-turn dialog $(u_{1:T}, r_{1:T})$ with cached latent state $\bar{h}_{1:T}$, generative model p_θ in Eq. (4), inference model q_ϕ in Eq. (6).

for $t = 1$ to T **do**

Propose $h'_t \sim q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t)$;

Accept h'_t as h_t, or reject h'_t and keep $\bar{h}_t$ as h_t, according to Eq. (10);

end for

Return: $h_{1:T}$, as a posterior sample from $p_\theta(h_{1:T}|u_{1:T}, r_{1:T})$ and used as the new cached latent state.

to the PARO mechanism of JSA, we have such a conceptual simplicity for learning seemly complex conditional sequential latent variable model.

4.5 Semi-Supervised TOD Systems via JSA

Now we have introduced the method of building latent state TOD systems (Eq. (4) and Eq. (6)) with JSA learning (Algorithm 1 and Algorithm 2), which is referred to JSA-TOD. Semi-supervised learning over a mix of labeled and unlabeled data could be readily realized in JSA-TOD by maximizing the weighted sum of $\log p_\theta(h_{1:T}, r_{1:T}|u_{1:T})$ (the conditional joint log-likelihood) over labeled data and $\log p_\theta(r_{1:T}|u_{1:T})$ (the conditional marginal log-likelihood) over unlabeled data.

The semi-supervised training procedure of JSA-TOD is summarized in Algorithm 3. Specifically, we first conduct supervised pre-training of both the generative model p_θ and the inference model q_ϕ on labeled data in JSA-TOD. Then we randomly draw supervised and unsupervised mini-batches from labeled and unlabeled data. For labeled dialogs, the latent states h_t's are given (labeled). For unlabeled dialogs, we apply the recursive turn-level MIS sampler (Algorithm 2) to sample the latent states h_t's[2] and treat them as if being given. The gradients calculation and parameter updating are then the same for labeled and unlabeled dialogs. Such simplicity in application is an appealing property of JSA, apart from its superior performance, as we show later in experiments.

[2]Sampling is empirically implemented via greedy decoding in our experiments.

Algorithm 3 Semi-supervised training in JSA-TOD

Input: A mix of labeled and unlabeled dialogs.

Run supervised pre-training of θ and ϕ on labeled dialogs;

repeat

Draw a dialog $(u_{1:T}, r_{1:T})$;

if $(u_{1:T}, r_{1:T})$ is not labeled **then**

Generate $h_{1:T}$ by applying the recursive turn-level MIS sampler (Algorithm 2);

end if

$J_\theta = 0, J_\phi = 0$;

for $i = 1, \cdots, T$ **do**

$J_\theta += \log p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t)$;

$J_\phi += \log q_\phi(h_t|h_{t-1}, r_{t-1}, u_t, r_t)$;

end for

Update θ by ascending: $\nabla_\theta J_\theta$;

Update ϕ by ascending: $\nabla_\phi J_\phi$;

until convergence

return θ and ϕ

5 Experiments

5.1 Experiment settings

Experiments are conducted on MultiWOZ2.1 (Eric et al., 2020), which is an English multi-domain dialogue dataset of human-human conversations, collected in a Wizard-of-Oz setup with 10.4k dialogs over 7 domains. The dataset was officially randomly split into a train, test and development set, which consist of 8434, 1000 and 1000 dialog samples, respectively. The dialogs in the dataset are all labeled with dialog states and system acts at every turn. Compared to MultiWOZ2.0, MultiWOZ2.1 removed some noisy state values. Following (Liu et al., 2022), some inappropriate state values and spelling errors are further corrected. Dialog responses are delexicalized to reduce surface language variability. We implement domain-adaptive pre-processing like in DAMD (Zhang et al., 2020a). More implementation details for our experiments are available in Appendix B.

For evaluation in MultiWOZ2.1, there are mainly four metrics for corpus based evaluation (Mehri et al., 2019). *Inform Rate* measures how often the entities provided by the system are correct; *Success Rate* refers to how often the system is able to answer all the requested attributes by user; *BLEU Score* is used to measure the fluency of the generated responses by analyzing the amount of n-gram overlap between the real responses and the gener-

Table 1: Main results on MultiWOZ2.1 for comparison between supervised-only, variational, and JSA methods. Results are reported as the mean and standard deviation from 3 runs with different random seeds.

Proportion	Method	Inform	Success	BLEU	Combined
100%	Sup-only	84.50±0.29	72.77±0.50	18.96±0.36	97.59±0.54
20%	Sup-only	75.70±1.87	61.07±2.21	16.66±0.29	85.05±2.16
	Variational	81.83±1.55	67.67±0.50	17.88±0.95	92.63±0.30
	JSA	83.25± 0.65	71.40±1.20	18.72±0.07	**96.04±0.85**
15%	Sup-only	80.00±0.43	55.57±1.22	16.20±0.24	79.00±1.51
	Variational	80.85±0.65	67.67±0.88	17.68±0.29	91.86±0.19
	JSA	83.23±0.53	71.97±1.27	18.59±0.19	**95.47±0.73**
10%	Sup-only	67.57±0.39	50.03±1.09	15.31±0.28	74.11±0.59
	Variational	80.67±1.33	66.97±1.23	17.34±0.56	91.15±1.80
	JSA	81.97±0.79	70.40±0.99	18.09±0.38	**94.27±1.23**
5%	Sup-only	49.73±2.45	33.67±1.79	14.07±0.13	55.77±1.94
	Variational	74.17±0.53	59.93±0.34	16.06±0.69	83.11±1.04
	JSA	72.37±1.19	59.73±0.92	18.57±0.54	**84.62±0.43**

ated responses; *Combined Score* is computed as (BLEU + 0.5 * (Inform + Success)). To avoid any inconsistencies in evaluation, we use the evaluation scripts in Nekvinda and Dušek (2021), which are now also the standardized scripts adopted in the MultiWOZ website.

5.2 Main Results

In the semi-supervised experiments, we randomly draw some proportions (5%, 10%, 15% and 20%) of the labeled dialogs from the MultiWOZ2.1 training set, with the rest dialogs in the training set treated as unlabeled, and conduct semi-supervised experiments. Specifically, the number of dialogs kept as labeled under these proportions are 1686, 1265, 843, and 421, respectively, while the rest dialogs are used as unlabeled (i.e., the original labels of dialog states and system acts at all turns are removed for those dialogs in the training set).

The main results are shown in Table 1. For model instantiations, we use the GPT2 based Markov generative model and inference model, as introduced in Liu et al. (2022). It has been shown in Liu et al. (2022) that using Markovian generative architecture achieves better results than non-Markov models in the low-resource setting for both supervised-only learning and semi-supervised variational learning, which makes it a strong baseline to compare. We first train the generative model and inference model on only the labeled data, which is referred to as "Supervised-only" (Sup-only for short). Then, we perform semi-supervised training on both labeled and unlabeled data. Using the variational method in (Liu et al., 2021, 2022), we get the baseline results of "Variational", where the Straight-Through trick is used to propagate the gradients through discrete latent variables. Using the JSA method proposed in Algorithm 3, we get the results of "JSA". We conduct the experiments with 3 random seeds and report the mean and standard deviation in Table 1.

From Table 1, we can see that both the Variational and the JSA methods outperform the Supervised-only method substantially across all label proportions. This clearly demonstrate the advantage of semi-supervised TOD systems. Remarkably, semi-supervised JSA-TOD using 20% labels performs close to the supervised-only baseline using 100% labels on MultiWOZ2.1.

When comparing the two semi-supervised methods, JSA performs better than Variational significantly across almost all label proportions in terms of all four metrics (Inform Rate, Success Rate, BLEU, and Combined Score). Exceptionally, in the case of 5% labels, the Inform Rate of JSA is worse than that of Variational, the Success Rates are close; Nevertheless, the Combined Score of JSA is significantly better. Presumably, this is because we use the Combined Scores to monitor the training, apply early stopping and select the model with the best Combined Score on the validation set. Such model selection put more priority on the overall performance in terms of Combined Scores.

Further, the results in Table 1 are pooled over all label proportions and all random seeds, and the matched-pairs significance tests (Gillick and Cox, 1989) are conducted to compare JSA and Variational for Inform, Success and BLEU respectively. The p-values are 9.27×10^{-2}, 2.576×10^{-14}, and 2.939×10^{-39} respectively, which show that JSA significantly outperforms Variational.

5.3 Ablation study and analysis

Notably, the JSA and the variational methods in our experiments use the same model instantiations for p_θ and q_ϕ. The only difference lies in the learning methods they used. In the following, we provide ablation study to illustrate the superiority of JSA over variational in learning latent state TOD models.

The importance of Metropolis Independence Sampling in JSA. In JSA, we need to use Monte Carlo sampling, particularly the Metropolis Independence Sampling (MIS) to decide whether or not to update the cached latent states h_t's. A naive method is to always accept the labels proposed by the inference model, which is somewhat like self-training (Rosenberg et al., 2005). Another simple method is to run session-level MIS, with the whole

Table 2: Ablation results for using different methods to update latent states h_t's (label proportion: 10%, random seed:11)

Method	Inform	Success	BLEU	Combined
Without MIS	71.10	59.80	18.71	84.16
Session-level MIS	78.70	65.50	17.20	89.30
Recursive turn-level MIS	82.80	71.80	18.56	**95.86**

Eq. (4) as the target distribution and the whole Eq. (6) as the proposal distribution. The whole $h_{1:T}$ is proposed via ancestral sampling and then get accepted/rejected. The results from one run with each different method are shown in Table 2. Both MIS based methods significantly improves the results, which clearly reveals the importance of using MIS in JSA. By the accept/reject mechanism, we accept latent states which have higher importance ratios and exploit them to update both generative model and inference model, and at the same time, we also explore the state space by randomly accepting latent states which have lower importance ratios. Exploitation and exploration of the latent states seems to be well balanced in JSA, which may explain its good performance. Our proposed recursive turn-level MIS in Algorithm 2 clearly outperforms the session-level MIS, since it samples in a much lower dimensional state space.

The latent state prediction performance of inference model. In both variational and JSA learning, the inference model q_ϕ, which is introduced to approximate the true posterior, plays an important role. The latent states inferred from q_ϕ are used, either directly as in variational learning or after accepted/rejected as in JSA learning, to optimize the generative model p_θ. We measure the quality of the latent states predicted from q_ϕ by label precision/recall/F1, compared to oracle b_t and a_t (excluding db_t) . We compare different q_ϕ obtained from three training methods - Supervised-only, Variational, and JSA. Note that at the end of running any particular training method, we obtain not only p_θ but also q_ϕ. The performances of p_θ over the test set are shown in Table 1. The testing performances of q_ϕ obtained from one run of each different method are shown in Table 3. It can be seen that semi-supervised variational learning does not improve the prediction ability of the inference model, compared to the inference model trained only on the labeled data. In contrast, the prediction performance of the inference model is increased significantly by semi-supervised JSA learning, which is in line with the superior results of JSA's generative model as shown in Table 1.

Table 3: Performance comparison of inference models from different methods, measured by latent state prediction precision/recall/F1 over the test set.

Label Proportion	Method	Precision	Recall	F1
20%	Supervised-only	0.928	0.908	0.918
	Variational	0.924	0.900	0.912
	JSA	**0.936**	**0.925**	**0.931**
15%	Supervised-only	0.924	0.891	0.907
	Variational	0.917	0.872	0.894
	JSA	**0.934**	**0.910**	**0.922**
10%	Supervised-only	0.916	0.868	0.891
	Variational	0.887	0.880	0.883
	JSA	**0.930**	**0.898**	**0.914**
5%	Supervised-only	0.894	0.804	0.847
	Variational	0.891	0.838	0.864
	JSA	**0.904**	**0.863**	**0.883**

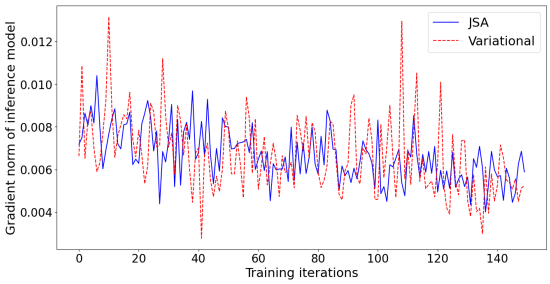

Figure 3: Comparison of the gradient norms from the inference models during training, using variational and JSA methods respectively (label proportion: 10%).

The variance of the gradients from inference model. The gradients for the inference model parameters in variational learning are known to have high-variance, due to gradient propagation through discrete latent variables, while JSA avoids such drawback. From one run of semi-supervised learning under 10% labels, we plot the gradient norms for the inference model parameters, from using the variational and the JSA methods respectively, which are shown in Figure 3. For clarity of comparison, we normalize the sum of the gradient norms over all iterations to be one. It can be clearly seen from Figure 3 that the gradients during variational training are more noisy than those in JSA tranining. Specifically, the variances of the time-series of the gradient norms in Figure 3 are 3.097×10^{-6} and 1.527×10^{-6} for the variational and the JSA methods respectively.

Pretrained models on external dialog corpora can be further improved by JSA learning for semi-supervised TOD systems. Pretraining and LVM based learning are two broad classes of semi-supervised methods. Recently, pretraining on external dialog copora has also shown to be promising

Table 4: MultiWOZ2.1 testing results for different methods (label proportion: 3%). "+Pretrained model" means that the method is initialized from the pretrained models over four external dialog corpra.

Method	Inform	Success	BLEU	Combined
Supervised-only	38.70	25.60	16.42	48.57
+ Pretrained model	57.70	39.30	14.15	62.65
Semi-supervised JSA	55.50	44.80	16.93	67.08
+ Pretrained model	**73.00**	**58.60**	**18.61**	**84.41**

for building TOD systems for low-resource scenarios (Peng et al., 2020; Su et al., 2021). In this section, we show that JSA learning can be used to further improve over such pretrained models. We use four dialog corpora - MSRE2E (Li et al., 2018), Frames (El Asri et al., 2017), TaskMaster (Byrne et al., 2019) and SchemaGuided (Rastogi et al., 2020), which consist of 16545 dialogs with human annotations on belief states and dialog acts, and we follow the preprocessing in (Su et al., 2021). Generative model and inference model, initialized from GPT2, are pretrained separately on those four corpora, the same as that in supervised-pretraining. Then, we conduct semi-supervised training with only 3% labels in MultiWOZ2.1 (i.e., 240 labeled dialogs with the rest being unlabeled). The results in Table 4 show that semi-supervisded JSA on top of pretrained models obtains the best result. This is an encouraging result from using 3% labels, which is close to the naive supervised-only method using 20% labels as shown in Table 1.

6 Conclusion and Future Work

This paper represents a progress towards building semi-supervised TOD systems by learning latent state TOD models. Traditionally, variational learning is often used; notably, the recently emerged JSA method has been shown to surpass variational learning, particularly in learning of discrete latent variable models. This paper represents the first application of JSA in its conditional sequential version, particularly for such long sequential generational problems like in TOD systems. Extensive experiments clearly show the superiority of JSA-TOD over its variational learning counterpart, not only in benchmark metrics for semi-supervised TOD systems but also from the latent state prediction performances and the variances of the gradients of the inference model. Since discrete latent variable models are widely used in many natural language procession tasks, we hope the results presented in this paper will encourage the community to further explore the applications of JSA and improve upon current approaches.

7 Acknowledgements

This research was funded by Joint Institute of Tsinghua University - China Mobile Communications Group Co., Ltd., Beijing, China.

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 (ACL)*.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. 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 4516–4525.

Bernard Delyon, Marc Lavielle, and Eric Moulines. 1999. Convergence of a stochastic approximation version of the EM algorithm. *Annals of statistics*, pages 94–128.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39.

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: A corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *LREC*.

Laurence Gillick and Stephen J Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *International Conference on Acoustics, Speech, and Signal Processing,*, pages 532–535. IEEE.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*.

Xisen Jin, Wenqiang Lei, Zhaochun Ren, Hongshen Chen, Shangsong Liang, Yihong Zhao, and Dawei Yin. 2018. Explicit state tracking with semi-supervision for neural dialogue generation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*.

Hyunsu Kim, Juho Lee, and Hongseok Yang. 2020. Adaptive strategy for resetting a non-stationary markov chain during learning via joint stochastic approximation. In *Third Symposium on Advances in Approximate Bayesian Inference*.

Kyurae Kim, Jisu Oh, and Hongseok Kim. 2022. Markov-chain monte carlo score estimators for variational inference with score climbing.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations (ICLR)*.

Estelle Kuhn and Marc Lavielle. 2004. Coupling a stochastic approximation version of EM with an MCMC procedure. *ESAIM: Probability and Statistics*, 8:115–131.

Yohan Lee. 2021. Improving end-to-end task-oriented dialog system with a simple auxiliary task. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1296–1303.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Dongdong Li, Zhaochun Ren, Pengjie Ren, Zhumin Chen, Miao Fan, Jun Ma, and Maarten de Rijke. 2021. Semi-supervised variational reasoning for medical dialogue generation. In *ACM SIGIR*, pages 544–554.

Xiujun Li, Yu Wang, Siqi Sun, Sarah Panda, Jingjing Liu, and Jianfeng Gao. 2018. Microsoft dialogue challenge: Building end-to-end task-completion dialogue systems. *arXiv preprint arXiv:1807.11125*.

Hong Liu, Yucheng Cai, Zhenru Lin, Zhijian Ou, Yi Huang, and Junlan Feng. 2021. Variational latent-state GPT for semi-supervised task-oriented dialog systems. *arXiv preprint arXiv:2109.04314*.

Hong Liu, Yucheng Cai, Zhijian Ou, Yi Huang, and Junlan Feng. 2022. Revisiting Markovian generative architectures for efficient task-oriented dialog systems. *arXiv preprint arXiv:2204.06452*.

Shikib Mehri, Tejas Srinivasan, and Maxine Eskenazi. 2019. Structured fusion networks for dialog. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*.

Christian Naesseth, Fredrik Lindsten, and David Blei. 2020. Markovian score climbing: Variational inference with KL(p||q). *Advances in Neural Information Processing Systems*, 33:15499–15510.

Tomáš Nekvinda and Ondřej Dušek. 2021. Shades of bleu, flavours of success: The case of multiwoz. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 34–46.

Zhijian Ou and Yunfu Song. 2020. Joint stochastic approximation and its application to learning discrete latent variable models. In *Conference on Uncertainty in Artificial Intelligence*, pages 929–938. PMLR.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics (TACL), 2021*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8689–8696.

Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. 2005. Semi-supervised self-training of object detection models. *WACV/MOTION*.

Yunfu Song and Zhijian Ou. 2020. Semi-supervised seq2seq joint-stochastic-approximation autoencoders with applications to semantic parsing. *IEEE Signal Processing Letters*, 27:31–35.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2021. Multi-task pre-training for plug-and-play task-oriented dialogue system. *CoRR*, abs/2109.14739.

Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve J. Young. 2017. Latent intention dialogue models. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

Haotian Xu and Zhijian Ou. 2016. Joint stochastic approximation learning of Helmholtz machines. In *ICLR Workshop Track*.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. UBAR: Towards fully end-to-end task-oriented dialog system with GPT-2. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Yichi Zhang, Zhijian Ou, Min Hu, and Junlan Feng. 2020b. A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020a. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*.

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*.

A Proof of Eq. (8)

First, we have

$$
\begin{aligned}
&p_\theta(h_{1:t}, r_{1:t}|u_{1:t}) \\
=&p_\theta(h_{1:t-1}, r_{1:t-1}|u_{1:t-1}, u_t) \\
&\times p_\theta(h_t, r_t|u_{1:t-1}, u_t, h_{1:t-1}, r_{1:t-1}) \\
=&p_\theta(h_{1:t-1}, r_{1:t-1}|u_{1:t-1}, \cancel{u_t}) \\
&\times p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t, \cancel{h_{1:t-2}, r_{1:t-2}, u_{1:t-1}}) \\
=&p_\theta(h_{1:t-1}, r_{1:t-1}|u_{1:t-1}) p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t)
\end{aligned}
$$

It can be seen that in simplifying the above equations, those conditional independence properties hold for our generative model Eq. (4). Then,

$$
\begin{aligned}
&p_\theta(h_{1:t}|u_{1:t}, r_{1:t}) = \frac{p_\theta(h_{1:t}, r_{1:t}|u_{1:t})}{p_\theta(r_{1:t}|u_{1:t})} \\
=&\frac{p_\theta(h_{1:t-1}, r_{1:t-1}|u_{1:t-1}) p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t)}{p_\theta(r_{1:t}|u_{1:t})} \\
=&p_\theta(h_{1:t-1}|u_{1:t-1}, r_{1:t-1}) p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t) \\
&\times \frac{p_\theta(r_{1:t-1}|u_{1:t-1})}{p_\theta(r_{1:t}|u_{1:t})} \\
\propto&p_\theta(h_{1:t-1}|u_{1:t-1}, r_{1:t-1}) p_\theta(h_t, r_t|h_{t-1}, r_{t-1}, u_t)
\end{aligned}
$$

B Implementation Details

We implement the models with Huggingface Transformers repository of version 4.8.2. We initialize both the generative model and the inference model with DistilGPT-2, a distilled version of GPT2. For all of supervised pre-training, variational learning and JSA learning, we use the AdamW optimizer and a linear scheduler with 20% warm up steps and maximum learning rate 10^{-4}. The minibatch base size is set to be 8 with gradient accumulation steps of 4. The 3 random seeds for the results in Table 1 are 9, 10 and 11. The total epochs for supervised pre-training are 50, and those for both variational learning and JSA learning are 40. We monitor the performance on the validation set and apply early stopping (stop when the current best model is not exceeded by models in the following 4 epochs). We select the best model on the validation set, then evaluate it on test set. All our experiments are performed on a single 32GB Tesla-V100 GPU.