Interpretable methods for aligning documents to dialogue

Justin

December 27, 2022

1 Introduction

In many customer-facing dialogue applications, customer service interactions must follow a set of guidelines for safety, which have a natural sequential order. If a customer requests is locked out of their account and requests a password reset, the agent must first verify that the customer is indeed the owner of the account. This if-then structure is common to flows in guidelines.

Both human and robot agent must follow safety guidelines. As a result, safety guidelines are often written in natural language.

Our goal is to train dialogue agents that not only follow a set of guidelines, but justify their actions by pointing to the guidelines. This allows others to verify their actions, and whether the guidelines have been followed.

We utilize a generative model of dialogue, that justifies decisions by aligning to a guidelines, utilizes the sequential structure of guidelines, and does not require supervision.

Experiments show that our model is accurate, intepretable, and works at a range of supervision levels.

2 Related work

The adaptation of large languge models to task-oriented dialogue has allowed for impressive results in zero-shot generalization, where models are tested in scenarios that they have not previously seen []. The key idea behind this success is the use of a natural language interface: specify scenario-specific details using natural language, and take advantage of the generalization abilities of large language models.

3 Problem setup

The goal is to maximize the log marginal likelihood,

$$\log p(x) = \log \sum_z p(x, z), \tag{1}$$

for a latent variable model. The derivative of the above is

$$\nabla \log \sum_z p(x, z) = \frac{p(x, z)}{p(x)} \nabla \log p(x, z) = p(z \mid x) \nabla \log p(x, z), \tag{2}$$

the expected gradient under the posterior. When exact marginalization is intractable, a common approach is to introduce a variational approximation to the posterior $q(z \mid x)$ and optimize the lower bound

$$\log p(x) = \log \sum_z q(z \mid x) \frac{p(x,z)}{q(z \mid x)} \geq \sum_z q(z \mid x) \log \frac{p(x,z)}{q(z \mid x)}, \tag{3}$$

which allows for tractable Monte Carlo approximation. [1]

Empirically, we find directly optimizing this lower bound (3) to be more difficult than optimizing the marginal likelihood (1), often requiring a baseline:

$$\begin{aligned} &\nabla_q \sum_z q(z \mid x) \log \frac{p(x,z)}{q(z \mid x)} \\ &= \nabla_q \sum_z q(z \mid x) \log p(x \mid z) - KL[q(z \mid x) || p(z)] \\ &= \nabla_q \sum_z q(z \mid x)(\log p(x \mid z) - B) - KL[q(z \mid x) || p(z)], \end{aligned} \tag{4}$$

where the baseline B is not a function of z.[2] Computing this baseline B can be expensive, potentially requiring multiple evaluations of $p(x \mid z)$. A common choice of baseline is the sample-average baseline, where $B = \sum_{z' \in Z} \log p(x \mid z')$ and Z is a set of iid samples.[3] Additionally, even in scenarios where exact marginalization is tractable, prior work has included a baseline. This begs the questions: Is a baseline necessary, and why does optimizing the marginal likelihood not require a baseline?

We show that the gradient of the marginal likelihood (2) already contains a built-in baseline, and use that to derive a simple and computable lower bound of the marginal likelihood (1) whose gradient does not require the manual engineering of a baseline.

4 Logsumexp

4.1 The gradient of logsumexp approximates the exponentiated regret

We start by reviewing the properties of the gradient of the logsumexp function, before proposing an efficient estimator for variational learning.

The key component of the marginal likelihood is the logsumexp operation, a smooth version of max: $\log \sum_z \exp f(z)$.[4] Similar to the gradient of the marginal likelihood, the gradient of logsumexp is given by

$$\nabla \log \sum_{z' \in Z} \exp f(z') = \frac{e^{f(z)}}{\sum_{z' \in Z} e^{f(z')}} \nabla f(z) = e^{f(z) - \log \sum_{z' \in Z} e^{f(z')}} \nabla f(z) \approx \underbrace{e^{f(z) - \max_{z' \in Z} f(z')}}_{R} \nabla f(z).$$

The last approximation holds because logsumexp is a smooth approximation of max. We can therefore think of $\log \sum_{z' \in Z} \exp f(z') \approx \max_{z' \in Z} f(z)$ as a built-in baseline. R is also the exponentiated regret, the difference between a given $f(z)$ and the best $f(z)$. Since the regret is non-positive, the exponentiated regret is always between 0 and 1.

[1] The gap between the two is given by $KL[q(z \mid x) || p(z \mid x)]$.

[2] The derivative is a linear operator, and $\nabla \sum_z q(z \mid x) B = B \cdot \nabla \sum_z q(z \mid x) = B \cdot 0$.

[3] A more efficient alternative is the leave-one-out baseline, which is efficient if the results of $p(x \mid z)$ are already available for a set of iid z.

[4] $\log \sum \exp(f(z))$ is perhaps better known as the log-partition function.

4.2 Approximating the gradient of the marginal likelihood

Given these properties of the gradient of logsumexp, we return to variational learning.

In the marginal likelihood setting (equation 1) $f(z) = \log p(x, z)$, and the regret compares a given $\log p(x, z)$ to the best $\log p(x, z^*)$. In the variational setting (equation (3)), the gradient of the ELBO does not have a built-in baseline or a regret interpretation.

We propose to approximate the gradient of logsumexp by using a restriction of logsumexp to $\mathcal{Z}' \subseteq \mathcal{Z}$:

$$\log \sum_{z \in \mathcal{Z}'} \exp \log p(x, z). \tag{5}$$

We learn a proposal distribution $q(z \mid x)$ in order to choose $\mathcal{Z}'$.

We instead optimize q to approximate $p(z \mid x)$ by optimizing

$$\log \sum_{z} \exp \log p(x, z) - KL[p(z \mid x) || q(z \mid x)] \approx \underbrace{\log \sum_{z \in \mathcal{Z}'} \exp \log \tilde{p}(x, z)}_{\text{ML}} - \underbrace{\sum_{z \in \mathcal{Z}'} \tilde{p}(z \mid x) \log \frac{\tilde{p}(z \mid x)}{\tilde{q}(z \mid x)}}_{\text{KL}}, \tag{6}$$

where $\tilde{p}(z \mid x) = \frac{p(x,z)}{\sum_{z' \in \mathcal{Z}'} p(x,z')}$, where the normalizing constant is approximated.[5]

We optimize this in two stages:

1. Maximize ML with respect to p using $\mathcal{Z}' = \text{topk} q(z \mid x)$
2. Maximize KL with respect to q using $\mathcal{Z}' = \text{topk} p(z)$

Is this roughly a hacky contrastive wake-sleep implementation?

4.3 Gradient estimator analysis

TBD

5 Related work

Contrastive divergence uses MCMC to estimate the gradient of the log partition function. This is identical to differentiating through a Monte Carlo approximation of the log partition function, commonly known as a sampled softmax / contrastive learning.

Wake-sleep and reweighted wake-sleep optimize the evidence plus the forward KL (M-projection), which is our objective. However, the reconstruction term in the wake-sleep objectiveis identical the one used in VAEs, which suffers from poor conditioning. This conditioning is improved by leaving the logsumexp operator intact, rather lower-bounded using Jensen's inequality.

DiCE is an infinitely differentiable Monte Carlo gradient estimator. It is a surrogate objective that is written to ensure correct higher order derivatives. Our surrogate objective is also designed to ensure that the gradient holds certain properties, but we focus on controlling the conditioning of the gradient rather than on the correctness of higher order derivatives.

[5] This should lead to a derivation of contrastive divergence and contrastive wake sleep. This is probably just an autodiff trick for doing contrastive wake sleep.