Semi-Amortized Variational Autoencoders

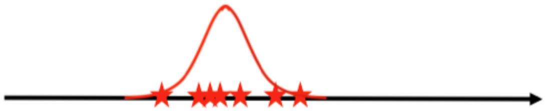
Yoon Kim, Sam Wiseman, Andrew C. Miller, David Sontag and Alexander M. Rush
presenter: Shen Yuan

October 21, 2021

- Background
- Semi-Amortized Variational Autoencoders
- Experiments

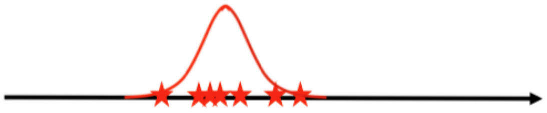
Background of Variational Inference

Here is a unknown distribution $p(\boldsymbol{x}; \theta)$, we need to calculate its parameters θ.



Background of Variational Inference

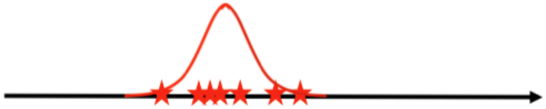
Here is a unknown distribution $p(\boldsymbol{x};\theta)$, we need to calculate its parameters θ.



- sample $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ from the distribution

Background of Variational Inference

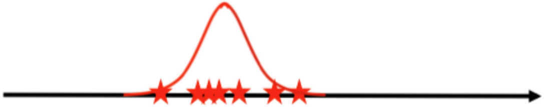
Here is a unknown distribution $p(\boldsymbol{x};\theta)$, we need to calculate its parameters θ.



- sample $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ from the distribution
- assume it is a gaussian distribution with $\theta = \{\mu, \sigma^2\}$

Background of Variational Inference

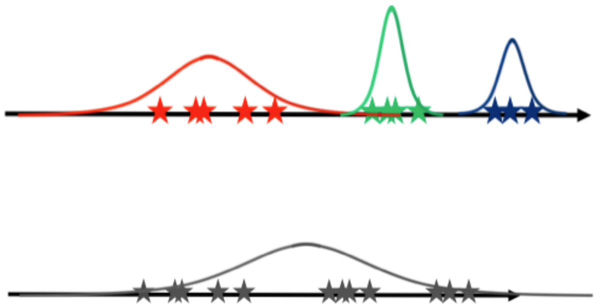
Here is a unknown distribution $p(\boldsymbol{x};\theta)$, we need to calculate its parameters θ.



- sample $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ from the distribution
- assume it is a gaussian distribution with $\theta = \{\mu, \sigma^2\}$
- maxmize its likelihood function $\hat{\theta} = \arg\max_\theta \hat{L}_n(\theta;\boldsymbol{x})$

Probabilistic latent variable models

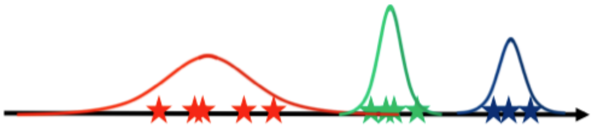
However, what happens very often is we don't actually know whether the unknown distribution consists of many other hidden distributions.



Probabilistic latent variable modelse

That is why to introduce the latent variable $\boldsymbol{z} = \{z_1, z_2, \dots, z_n\}$, which describes the probabilities of each sample generated by each hidden distribution.

$$p(\boldsymbol{x};\theta) = \sum_{z \in \boldsymbol{z}} p(\boldsymbol{x}|z;\theta)p(z) = \int p(\boldsymbol{x}|z;\theta)p(z)\mathrm{d}z$$



Probabilistic latent variable models

- sample $\boldsymbol{x} \sim p(\boldsymbol{x}; \theta)$
- calculate $p(\boldsymbol{z}|\boldsymbol{x}; \theta)$ based on $\boldsymbol{x}$
- calculate $p(\boldsymbol{x}|\boldsymbol{z}; \theta)$ based on $\boldsymbol{z}$
- get $p(\boldsymbol{x}; \theta)$

Probabilistic latent variable models

- sample $\boldsymbol{x} \sim p(\boldsymbol{x}; \theta)$
- calculate $p(\boldsymbol{z}|\boldsymbol{x}; \theta)$ based on $\boldsymbol{x}$
- calculate $p(\boldsymbol{x}|\boldsymbol{z}; \theta)$ based on $\boldsymbol{z}$
- get $p(\boldsymbol{x}; \theta)$

How to get $p(\boldsymbol{z}|\boldsymbol{x})$?

Variational inference

The goal of variational inference is to approximate a conditional density of latent variables by defining a variational family of distributions $q(\boldsymbol{z}; \lambda)$ parameterized by λ.

$$q_i(\boldsymbol{z}; \lambda_i) \sim p(\boldsymbol{z}|\boldsymbol{x}_i)$$

For each sample x_i, there is a corresponding distribution q_i to indicate which hidden distribution it belongs to.

Variational inference

$$
\begin{aligned}
\mathrm{KL}(q(\boldsymbol{z};\lambda)||p(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(q(\boldsymbol{z};\lambda)) - \log(p(\boldsymbol{z}|\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(q(\boldsymbol{z};\lambda)) - \log(\frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x})})] \\
&= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(q(\boldsymbol{z};\lambda)) - \log(p(\boldsymbol{x}|\boldsymbol{z})) - \log(p(\boldsymbol{z}))] + \log(p(\boldsymbol{x} \\
&= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(\frac{q(\boldsymbol{z};\lambda)}{p(\boldsymbol{z})}) - \log(p(\boldsymbol{x}|\boldsymbol{z}))] + \log(p(\boldsymbol{x})) \\
&= \mathrm{KL}[q(\boldsymbol{z};\lambda)||p(\boldsymbol{z})] - \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[log(p(\boldsymbol{x}|\boldsymbol{z}))] + \log(p(\boldsymbol{x})) \\
&= \log(p(\boldsymbol{x})) - \{\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(p(\boldsymbol{x}|\boldsymbol{z}))] - \mathrm{KL}[q(\boldsymbol{z};\lambda)||p(\boldsymbol{z})]\}
\end{aligned}
$$

Variational inference

$$
\begin{aligned}
\mathrm{KL}(q(\boldsymbol{z};\lambda)||p(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(q(\boldsymbol{z};\lambda)) - \log(p(\boldsymbol{z}|\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(q(\boldsymbol{z};\lambda)) - \log(\frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x})})] \\
&= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(q(\boldsymbol{z};\lambda)) - \log(p(\boldsymbol{x}|\boldsymbol{z})) - \log(p(\boldsymbol{z}))] + \log(p(\boldsymbol{x}) \\
&= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(\frac{q(\boldsymbol{z};\lambda)}{p(\boldsymbol{z})}) - \log(p(\boldsymbol{x}|\boldsymbol{z}))] + \log(p(\boldsymbol{x})) \\
&= \mathrm{KL}[q(\boldsymbol{z};\lambda)||p(\boldsymbol{z})] - \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[log(p(\boldsymbol{x}|\boldsymbol{z}))] + \log(p(\boldsymbol{x})) \\
&= \log(p(\boldsymbol{x})) - \{\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z};\lambda)}[\log(p(\boldsymbol{x}|\boldsymbol{z}))] - \mathrm{KL}[q(\boldsymbol{z};\lambda)||p(\boldsymbol{z})]\} \\
&= \log(p(\boldsymbol{x})) - \mathrm{ELBO}(\lambda, \theta, \boldsymbol{x})
\end{aligned}
$$

Stochastic Variational Inference

- sample $\boldsymbol{x} \sim p(\boldsymbol{x})$
- randomly initialize λ_0, θ
- for $k = 0, \dots, K-1$, set
  $\lambda_{k+1} = \lambda_k + \alpha \nabla_\lambda \operatorname{ELBO}(\lambda_k, \theta, \boldsymbol{x})$
- update θ based on $\nabla_\theta \operatorname{ELBO}(\lambda_k, \theta, \boldsymbol{x})$

Stochastic Variational Inference

- sample $\boldsymbol{x} \sim p(\boldsymbol{x})$
- randomly initialize λ_0, θ
- for $k = 0, \ldots, K-1$, set
  $\lambda_{k+1} = \lambda_k + \alpha \nabla_\lambda \operatorname{ELBO}(\lambda_k, \theta, \boldsymbol{x})$
- update θ based on $\nabla_\theta \operatorname{ELBO}(\lambda_k, \theta, \boldsymbol{x})$

This procedure, which is repeated for each example, is computationally expensive and requires setting step-size hyper-parameters α.

Amortized Variational Inference

Amortized Variational Inference uses a shared parametric inference model(i.e., encoder($\cdot$)) to predict the parameters λ_i for each example x_i. The word 'Amortized' means a shared model.

- sample $\boldsymbol{x} \sim p(\boldsymbol{x})$
- set $\lambda = \text{encoder}(\boldsymbol{x}; \phi)$
- update θ based on $\nabla_\theta \, \text{ELBO}(\lambda, \theta, \boldsymbol{x})$
- update ϕ based on $\nabla_\phi \, \text{ELBO}(\lambda, \theta, \boldsymbol{x})$
  $\frac{\mathrm{d}\,\text{ELBO}(\lambda,\theta,\boldsymbol{x})}{\mathrm{d}\phi} = \frac{\mathrm{d}\lambda}{\mathrm{d}\phi} \nabla_\lambda \, \text{ELBO}(\lambda, \theta, \boldsymbol{x})$

- Background
- Semi-Amortized Variational Autoencoders
- Experiments

Semi-Amortized Variational Autoencoders

However, requiring all the variational parameters $\boldsymbol{\lambda} = \{\lambda_1, \dots, \lambda_n\}$ to be a parametric function of the input may be too strict of a restriction and can lead to an amortization gap.

Semi-Amortized Variational Autoencoders

- sample $\boldsymbol{x} \sim p(\boldsymbol{x})$
- set $\lambda_0 = \text{encoder}(\boldsymbol{x}; \phi)$
- for $k = 0, \ldots, K-1$, set $\lambda_{k+1} = \lambda_k + \alpha \nabla_\lambda \, \text{ELBO}(\lambda_k, \theta, \boldsymbol{x})$
- update θ based on $\frac{\mathrm{d}\,\text{ELBO}(\lambda_K, \theta, \boldsymbol{x})}{\mathrm{d}\theta}$
- update ϕ based on $\frac{\mathrm{d}\,\text{ELBO}(\lambda_K, \theta, \boldsymbol{x})}{\mathrm{d}\phi}$

Semi-Amortized Variational Autoencoders

Algorithm 1 Semi-Amortized Variational Autoencoders

```
Input: inference network $\phi$, generative model $\theta$,
       inference steps $K$, learning rate $\alpha$, momentum $\gamma$,
       loss function $f(\lambda, \theta, \mathbf{x}) = -\text{ELBO}(\lambda, \theta, \mathbf{x})$
Sample $\mathbf{x} \sim p_{\mathcal{D}}(\mathbf{x})$
$\lambda_0 \leftarrow \text{enc}(\mathbf{x}; \phi)$
$v_0 \leftarrow 0$
for $k = 0$ to $K - 1$ do
    $v_{k+1} \leftarrow \gamma v_k - \nabla_\lambda f(\lambda_k, \theta, \mathbf{x})$
    $\lambda_{k+1} \leftarrow \lambda_k + \alpha v_{k+1}$
end for
$\mathcal{L} \leftarrow f(\lambda_K, \theta, \mathbf{x})$
$\overline{\lambda}_K \leftarrow \nabla_\lambda f(\lambda_K, \theta, \mathbf{x})$
$\overline{\theta} \leftarrow \nabla_\theta f(\lambda_K, \theta, \mathbf{x})$
$\overline{v}_K \leftarrow 0$
for $k = K - 1$ to $0$ do
    $\overline{v}_{k+1} \leftarrow \overline{v}_{k+1} + \alpha \overline{\lambda}_{k+1}$
    $\overline{\lambda}_k \leftarrow \overline{\lambda}_{k+1} - \text{H}_{\lambda,\lambda} f(\lambda_k, \theta, \mathbf{x}) \overline{v}_{k+1}$
    $\overline{\theta} \leftarrow \overline{\theta} - \text{H}_{\theta,\lambda} f(\lambda_k, \theta, \mathbf{x}) \overline{v}_{k+1}$
    $\overline{v}_k \leftarrow \gamma \overline{v}_{k+1}$
end for
$\frac{d\mathcal{L}}{d\theta} \leftarrow \overline{\theta}$
$\frac{d\mathcal{L}}{d\phi} \leftarrow \frac{d\lambda_0}{d\phi} \overline{\lambda}_0$
Update $\theta, \phi$ based on $\frac{d\mathcal{L}}{d\theta}, \frac{d\mathcal{L}}{d\phi}$
```

Semi-Amortized Variational Autoencoders

This paper calculated Hessian-vector products with finite differences, which was found to be more memory-efficient than automatic differentiation.

$$H_{\boldsymbol{u}_i,\boldsymbol{u}_j} f(\hat{u})\boldsymbol{v} \approx \frac{1}{\epsilon}(\nabla_{\boldsymbol{u}_i} f(\hat{\boldsymbol{u}}_0, \ldots, \hat{\boldsymbol{u}}_j + \epsilon\boldsymbol{v}, \ldots, \hat{\boldsymbol{u}}_m) - \nabla_{\boldsymbol{u}_i} f(\hat{\boldsymbol{u}}_0, \ldots, \hat{\boldsymbol{u}}_j, \ldots, \hat{\boldsymbol{u}}_m))$$

where ϵ is some small number (we use $\epsilon = 10^{-5}$).

Experiments

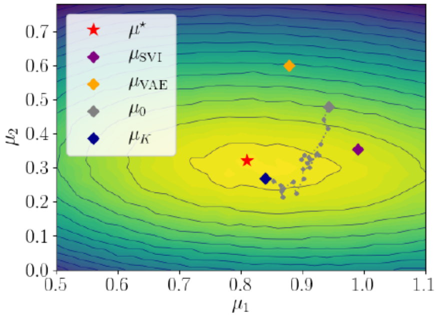
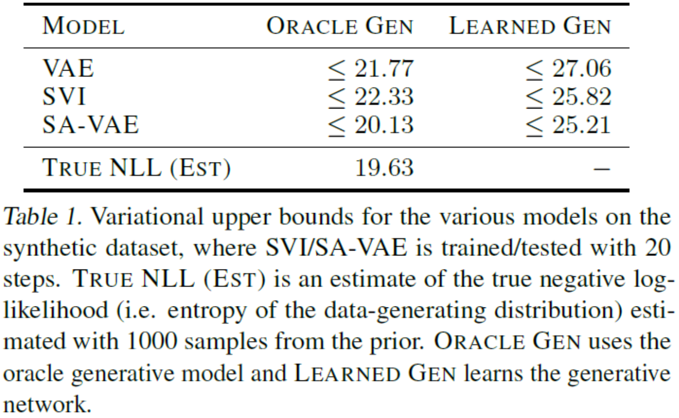


Figure 1. ELBO landscape with the oracle generative model as a function of the variational posterior means μ_1, μ_2 for a randomly chosen test point. Variational parameters obtained from VAE, SVI are shown as $\mu_{\text{VAE}}, \mu_{\text{SVI}}$ and the initial/final parameters from SA-VAE are shown as μ_0 and μ_K (along with the intermediate points). SVI/SA-VAE are run for 20 iterations. The optimal point, found from grid search, is shown as $\mu^\star$.

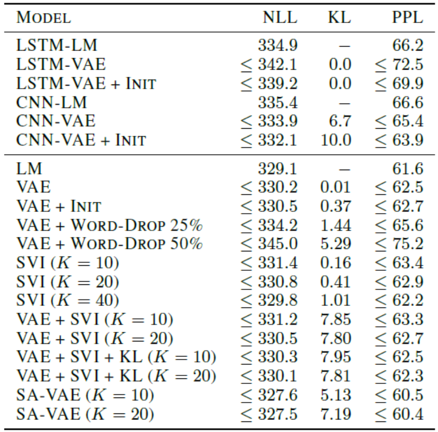
Experiments



MODEL	ORACLE GEN	LEARNED GEN
VAE	≤ 21.77	≤ 27.06
SVI	≤ 22.33	≤ 25.82
SA-VAE	≤ 20.13	≤ 25.21
TRUE NLL (EST)	19.63	–

Table 1. Variational upper bounds for the various models on the synthetic dataset, where SVI/SA-VAE is trained/tested with 20 steps. TRUE NLL (EST) is an estimate of the true negative log-likelihood (i.e. entropy of the data-generating distribution) estimated with 1000 samples from the prior. ORACLE GEN uses the oracle generative model and LEARNED GEN learns the generative network.

Experiments



MODEL	NLL	KL	PPL
LSTM-LM	334.9	–	66.2
LSTM-VAE	$\leq$ 342.1	0.0	$\leq$ 72.5
LSTM-VAE + INIT	$\leq$ 339.2	0.0	$\leq$ 69.9
CNN-LM	335.4	–	66.6
CNN-VAE	$\leq$ 333.9	6.7	$\leq$ 65.4
CNN-VAE + INIT	$\leq$ 332.1	10.0	$\leq$ 63.9
LM	329.1	–	61.6
VAE	$\leq$ 330.2	0.01	$\leq$ 62.5
VAE + INIT	$\leq$ 330.5	0.37	$\leq$ 62.7
VAE + WORD-DROP 25%	$\leq$ 334.2	1.44	$\leq$ 65.6
VAE + WORD-DROP 50%	$\leq$ 345.0	5.29	$\leq$ 75.2
SVI ($K = 10$)	$\leq$ 331.4	0.16	$\leq$ 63.4
SVI ($K = 20$)	$\leq$ 330.8	0.41	$\leq$ 62.9
SVI ($K = 40$)	$\leq$ 329.8	1.01	$\leq$ 62.2
VAE + SVI ($K = 10$)	$\leq$ 331.2	7.85	$\leq$ 63.3
VAE + SVI ($K = 20$)	$\leq$ 330.5	7.80	$\leq$ 62.7
VAE + SVI + KL ($K = 10$)	$\leq$ 330.3	7.95	$\leq$ 62.5
VAE + SVI + KL ($K = 20$)	$\leq$ 330.1	7.81	$\leq$ 62.3
SA-VAE ($K = 10$)	$\leq$ 327.6	5.13	$\leq$ 60.5
SA-VAE ($K = 20$)	$\leq$ 327.5	7.19	$\leq$ 60.4

Table 2. Results on text modeling on the Yahoo dataset. Top results are from Yang et al. (2017), while the bottom results are from this work (+ INIT means the encoder is initialized with a pretrained language model, while models with + WORD-DROP are trained with word-dropout). NLL/KL numbers are averaged across examples, and PPL refers to perplexity. K refers to the number of inference steps used for training/testing.

Experiments

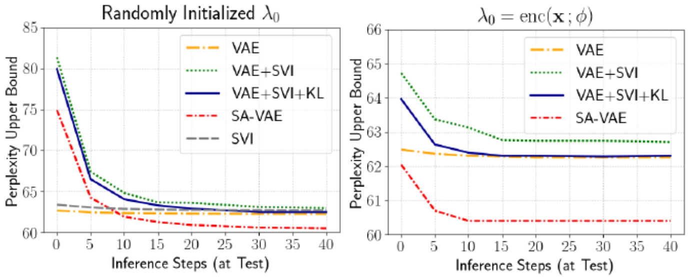


Figure 2. (Left) Perplexity upper bound of various models when trained with 20 steps (except for VAE) and tested with varying number of SVI steps from random initialization. (Right) Same as the left except that SVI is initialized with variational parameters obtained from the inference network.

Iterative Amortized Inference

Joseph Marino, Yisong Yue and Stephan Mandt
presenter: Shen Yuan

October 21, 2021

Iterative Amortized Inference

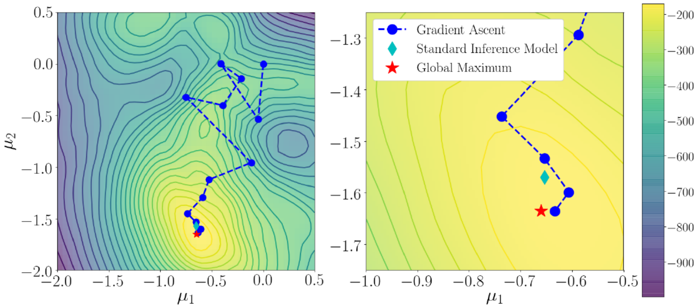


Figure 1. **Visualizing the amortization gap.** Optimization surface of $\mathcal{L}$ (in *nats*) for a 2-D latent Gaussian model and an MNIST data example. Shown on the plots are the optimal estimate (MAP), the output of a standard inference model, and an optimization trajectory of gradient ascent. The plot on the right shows an enlarged view near the optimum. Conventional optimization outperforms the standard inference model, exhibiting an amortization gap. With additional latent dimensions or more complex data, this gap could become larger.

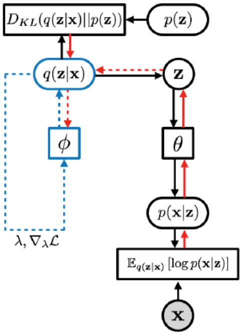
Iterative Amortized Inference

We denote the iterative inference model as f with parameters ϕ.

$$\boldsymbol{\lambda}_{t+1}^{(i)} \leftarrow f_t(\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t^{(i)}, \boldsymbol{\lambda}_t^{(i)}; \phi)$$

where $\boldsymbol{\lambda}$ is the distribution parameters of $q(\boldsymbol{z}|\boldsymbol{x})$, $\mathcal{L}_t^{(i)} \equiv \mathcal{L}(\boldsymbol{x}^{(i)}, \boldsymbol{\lambda}_t^{(i)}; \phi)$ represents the ELBO for data example $\boldsymbol{x}^{(i)}$ at inference iteration t.

Iterative Amortized Inference



$D_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$
$p(\mathbf{z})$
$q(\mathbf{z}|\mathbf{x})$
$\mathbf{z}$
ϕ
θ
$p(\mathbf{x}|\mathbf{z})$
$\lambda, \nabla_\lambda \mathcal{L}$
$\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})]$
$\mathbf{x}$

Figure 2. **Computation graph** for a single-level latent variable model with an iterative inference model. Black components evaluate the ELBO. Blue components are used during variational inference. Red corresponds to gradients. Solid arrows denote deterministic values. Dashed arrows denote stochastic values. During inference, $\boldsymbol{\lambda}$, the distribution parameters of $q(\mathbf{z}|\mathbf{x})$, are first initialized. $\mathbf{z}$ is sampled from $q(\mathbf{z}|\mathbf{x})$ to evaluate the ELBO. Stochastic gradients are then backpropagated to $\boldsymbol{\lambda}$. The iterative inference model uses these gradients to update the current estimate of $\boldsymbol{\lambda}$. The process is repeated iteratively. The inference model parameters, ϕ, are trained through accumulated estimates of $\nabla_\phi \mathcal{L}$.

Algorithm 1 Iterative Amortized Inference

```
**Input:** data $\mathbf{x}$, generative model $p_\theta(\mathbf{x},\mathbf{z})$, inference model $f$
Initialize $t = 0$
Initialize $\nabla_\phi = 0$
Initialize $q(\mathbf{z}|\mathbf{x})$ with $\boldsymbol{\lambda}_0$
**repeat**
    Sample $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})$
    Evaluate $\mathcal{L}_t = \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}_t; \theta)$
    Calculate $\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t$ and $\nabla_\phi \mathcal{L}_t$
    Update $\boldsymbol{\lambda}_{t+1} = f_t(\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t, \boldsymbol{\lambda}_t; \phi)$
    $t = t + 1$
    $\nabla_\phi = \nabla_\phi + \nabla_\phi \mathcal{L}_t$
**until** $\mathcal{L}$ converges
$\theta = \theta + \alpha_\theta \nabla_\theta \mathcal{L}$
$\phi = \phi + \alpha_\phi \nabla_\phi$
```

Semi-Amortized Variational Autoencoders

Algorithm 1 Semi-Amortized Variational Autoencoders

```
Input: inference network $\phi$, generative model $\theta$,
       inference steps $K$, learning rate $\alpha$, momentum $\gamma$,
       loss function $f(\lambda, \theta, \mathbf{x}) = -\text{ELBO}(\lambda, \theta, \mathbf{x})$
Sample $\mathbf{x} \sim p_{\mathcal{D}}(\mathbf{x})$
$\lambda_0 \leftarrow \text{enc}(\mathbf{x}; \phi)$
$v_0 \leftarrow 0$
for $k = 0$ to $K - 1$ do
    $v_{k+1} \leftarrow \gamma v_k - \nabla_\lambda f(\lambda_k, \theta, \mathbf{x})$
    $\lambda_{k+1} \leftarrow \lambda_k + \alpha v_{k+1}$
end for
$\mathcal{L} \leftarrow f(\lambda_K, \theta, \mathbf{x})$
$\overline{\lambda}_K \leftarrow \nabla_\lambda f(\lambda_K, \theta, \mathbf{x})$
$\overline{\theta} \leftarrow \nabla_\theta f(\lambda_K, \theta, \mathbf{x})$
$\overline{v}_K \leftarrow 0$
for $k = K - 1$ to $0$ do
    $\overline{v}_{k+1} \leftarrow \overline{v}_{k+1} + \alpha \overline{\lambda}_{k+1}$
    $\overline{\lambda}_k \leftarrow \overline{\lambda}_{k+1} - \text{H}_{\lambda,\lambda} f(\lambda_k, \theta, \mathbf{x}) \overline{v}_{k+1}$
    $\overline{\theta} \leftarrow \overline{\theta} - \text{H}_{\theta,\lambda} f(\lambda_k, \theta, \mathbf{x}) \overline{v}_{k+1}$
    $\overline{v}_k \leftarrow \gamma \overline{v}_{k+1}$
end for
$\frac{d\mathcal{L}}{d\theta} \leftarrow \overline{\theta}$
$\frac{d\mathcal{L}}{d\phi} \leftarrow \frac{d\lambda_0}{d\phi} \overline{\lambda}_0$
Update $\theta, \phi$ based on $\frac{d\mathcal{L}}{d\theta}, \frac{d\mathcal{L}}{d\phi}$
```

- Iterative Amortized Inference
- Iterative Inference in Latent Gaussian Models
- Experiments

Iterative Inference in Latent Gaussian Models

We now describe an instantiation of iterative inference models for (single-level) latent Gaussian models, which have a Gaussian prior density over latent variables: $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_p, \operatorname{diag} \boldsymbol{\sigma}_p^2)$

While the approximate posterior is also chosen as Gaussian:$q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_q, \operatorname{diag} \boldsymbol{\sigma}_q^2)$

$$\begin{aligned} \boldsymbol{\mu}_{q,t+1} &= f_t^{\boldsymbol{\mu}_q}(\nabla_{\boldsymbol{\mu}_q} \mathcal{L}_t, \boldsymbol{\mu}_{q,t}; \phi), \\ \boldsymbol{\sigma}^2_{q,t+1} &= f_t^{\boldsymbol{\sigma}^2_q}(\nabla_{\boldsymbol{\sigma}^2_q} \mathcal{L}_t, \boldsymbol{\sigma}^2_{q,t}; \phi) \end{aligned}$$

where $f_t^{\boldsymbol{\mu}_q}$ and $f_t^{\boldsymbol{\sigma}^2_q}$ are the iterative inference models for updating $\boldsymbol{\mu}_q$ and $\boldsymbol{\sigma}^2_q$ respectively.

Iterative Inference in Latent Gaussian Models

For instance, assuming a Gaussian output density $q(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu_x}, \operatorname{diag} \boldsymbol{\sigma}_{\boldsymbol{x}}^2)$, the gradient for $\boldsymbol{\mu}_q$ is

$$\nabla_{\boldsymbol{\mu}_q} \mathcal{L} = \boldsymbol{J}^\top \varepsilon_{\boldsymbol{x}} - \varepsilon_{\boldsymbol{z}}$$

where the Jacobian ($\boldsymbol{J}$), bottom-up errors ($\varepsilon_{\boldsymbol{x}}$), and top-down errors ($\varepsilon_{\boldsymbol{z}}$) are defined as

$$\begin{aligned}
\boldsymbol{J} &\equiv \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x})}[\frac{\partial \boldsymbol{\mu_x}}{\partial \boldsymbol{\mu_q}}] \\
\varepsilon_{\boldsymbol{x}} &\equiv \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x})}[(\boldsymbol{x} - \boldsymbol{\mu_x})/\boldsymbol{\sigma}_{\boldsymbol{x}}^2] \\
\varepsilon_{\boldsymbol{z}} &\equiv \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x})}[(\boldsymbol{z} - \boldsymbol{\mu_p})/\boldsymbol{\sigma}_{\boldsymbol{p}}^2]
\end{aligned}$$

Iterative Inference in Latent Gaussian Models

$$\begin{aligned}\varepsilon_{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \\ \boldsymbol{A} &\equiv \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[(\operatorname{diag} \boldsymbol{\sigma}_{\boldsymbol{x}}^2)^{-1}] \\ \boldsymbol{b} &\equiv -\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\frac{\boldsymbol{\mu}_{\boldsymbol{x}}}{\boldsymbol{\sigma}_{\boldsymbol{x}}^2}]\end{aligned}$$

Assuming that the initial approximate posterior and prior are both constant, then in expectation, $\boldsymbol{A}$, $\boldsymbol{b}$, and $\varepsilon_{\boldsymbol{z}}$ are constant across all data examples at the first inference iteration.

Using proper weight initialization and input normalization, it is equivalent to input $\boldsymbol{x}$ or an affine transformation of $\boldsymbol{x}$ into a fully-connected neural network.

Therefore, standard inference models are equivalent to the special case of a one-step iterative inference model.

- Iterative Amortized Inference
- Iterative Inference in Latent Gaussian Models
- Experiments

Experiments

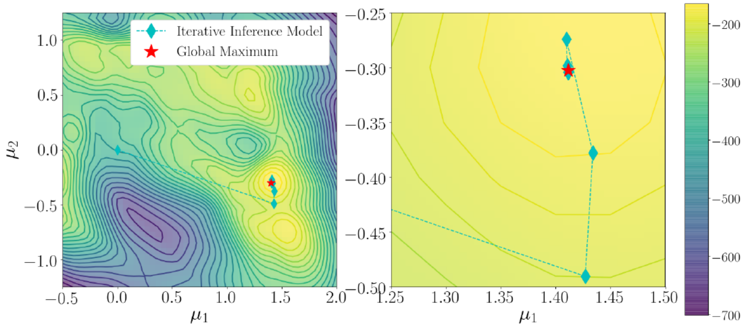


Figure 3. **Direct visualization of iterative amortized inference optimization.** Optimization trajectory on $\mathcal{L}$ (in *nats*) for an iterative inference model with a 2D latent Gaussian model for a particular MNIST example. The iterative inference model adaptively adjusts inference update step sizes to iteratively refine the approximate posterior estimate.

Experiments

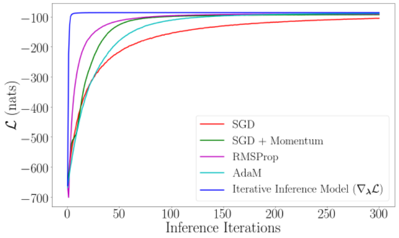


Figure 4. **Comparison of inference optimization performance between iterative inference models and conventional optimization techniques.** Plot shows ELBO, averaged over MNIST validation set. On average, the iterative inference model converges faster than conventional optimizers to better estimates. Note that the iterative inference model remains stable over hundreds of iterations, despite only being trained with 16 inference iterations.

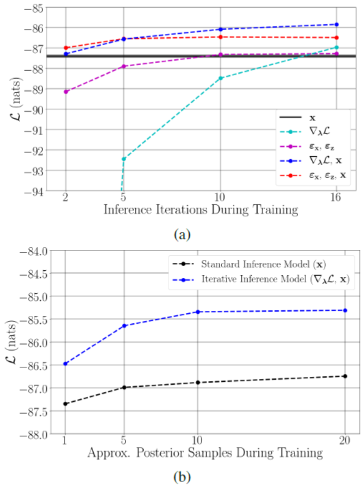


Figure 7. ELBO for standard and iterative inference models on MNIST for **(a)** additional inference iterations during training and **(b)** additional samples. Iterative inference models improve significantly with both quantities. Lines do not imply interpolation.