Semi-Amortized Variational Autoencoders

Yoon Kim    Sam Wiseman    Andrew Miller
David Sontag    Alexander Rush

Code: https://github.com/harvardnlp/sa-vae
Background: Variational Autoencoders (VAE) (Kingma et al. 2013)

Generative model:

- Draw $z$ from a simple prior: $z \sim p(z) = \mathcal{N}(0, I)$
- Likelihood parameterized with a deep model $\theta$, i.e. $x \sim p_\theta(x \mid z)$

Training:

- Introduce variational family $q_\lambda(z)$ with parameters $\lambda$
- Maximize the evidence lower bound (ELBO)

$$
\log p_\theta(x) \geq \mathbb{E}_{q_\lambda(z)} \left[ \log \frac{p_\theta(x, z)}{q_\lambda(z)} \right]
$$

- VAE: $\lambda$ output from an inference network $\phi$

$$
\lambda = \text{enc}_\phi(x)
$$
Background: Variational Autoencoders (VAE) (Kingma et al. 2013)

Generative model:

- Draw $z$ from a simple prior: $p(z) = \mathcal{N}(0, I)$
- Likelihood parameterized with a deep model $\theta$, i.e. $x \sim p_\theta(x | z)$

Training:

- Introduce variational family $q_\lambda(z)$ with parameters $\lambda$
- Maximize the evidence lower bound (ELBO)

$$
\log p_\theta(x) \geq \mathbb{E}_{q_\lambda(z)} \left[ \log \frac{p_\theta(x, z)}{q_\lambda(z)} \right]
$$

- VAE: $\lambda$ output from an inference network $\phi$

$$
\lambda = \text{enc}_\phi(x)
$$
Background: Variational Autoencoders (VAE) (Kingma et al. 2013)

- **Amortized Inference**: *local* per-instance variational parameters \( \lambda^{(i)} = \text{enc}_\phi(x^{(i)}) \) predicted from a *global* inference network (cf. per-instance optimization for traditional VI)

- **End-to-end**: generative model \( \theta \) and inference network \( \phi \) trained together (cf. coordinate ascent-style training for traditional VI)
Background: Variational Autoencoders (VAE) (Kingma et al. 2013)

- **Amortized Inference**: *local* per-instance variational parameters $\lambda^{(i)} = \text{enc}_\phi(x^{(i)})$ predicted from a *global* inference network (cf. per-instance optimization for traditional VI)

- **End-to-end**: generative model $\theta$ and inference network $\phi$ trained together (cf. coordinate ascent-style training for traditional VI)
Background: Variational Autoencoders (VAE) (Kingma et al. 2013)

- Generative model: $\int p_\theta(x|z)p(z)dz$ gives good likelihoods/samples
- Representation learning: $z$ captures high-level features
(1) Posterior collapse

- If generative model $p_\theta(x | z)$ is too flexible (e.g. PixelCNN, LSTM), model learns to ignore latent representation, i.e. $\text{KL}(q(z) \| p(z)) \approx 0$.

- Want to use powerful $p_\theta(x | z)$ to model the underlying data well, but also want to learn interesting representations $z$. 

VAE Issues: Posterior Collapse (Bowman al. 2016)
(1) Posterior collapse

- If generative model $p_\theta(x | z)$ is too flexible (e.g. PixelCNN, LSTM), model learns to ignore latent representation, i.e. $KL(q(z) || p(z)) \approx 0$.

- Want to use powerful $p_\theta(x | z)$ to model the underlying data well, but also want to learn interesting representations $z$.
VAE Issues: Posterior Collapse (Bowman al. 2016)

(1) Posterior collapse

- If generative model $p_\theta(x|z)$ is too flexible (e.g. PixelCNN, LSTM), model learns to ignore latent representation, i.e.
  \[ KL(q(z) \parallel p(z)) \approx 0. \]

- Want to use powerful $p_\theta(x|z)$ to model the underlying data well, but also want to learn interesting representations $z$. 
Example: Text Modeling on Yahoo corpus (Yang et al. 2017)

Inference Network: LSTM + MLP
Generative Model: LSTM, $z$ fed at each time step

<table>
<thead>
<tr>
<th>Model</th>
<th>KL</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LANGUAGE MODEL</td>
<td>–</td>
<td>61.6</td>
</tr>
<tr>
<td>VAE</td>
<td>0.01</td>
<td>$\leq 62.5$</td>
</tr>
<tr>
<td>VAE + Word-Drop 25%</td>
<td>1.44</td>
<td>$\leq 65.6$</td>
</tr>
<tr>
<td>VAE + Word-Drop 50%</td>
<td>5.29</td>
<td>$\leq 75.2$</td>
</tr>
<tr>
<td>ConvNetVAE (Yang et al. 2017)</td>
<td>10.0</td>
<td>$\leq 63.9$</td>
</tr>
</tbody>
</table>
Example: Text Modeling on Yahoo corpus (Yang et al. 2017)

Inference Network: LSTM + MLP

Generative Model: LSTM, \( z \) fed at each time step

<table>
<thead>
<tr>
<th>Model</th>
<th>KL</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language Model</td>
<td>–</td>
<td>61.6</td>
</tr>
<tr>
<td>VAE</td>
<td>0.01</td>
<td>( \leq 62.5 )</td>
</tr>
<tr>
<td>VAE + Word-Drop 25%</td>
<td>1.44</td>
<td>( \leq 65.6 )</td>
</tr>
<tr>
<td>VAE + Word-Drop 50%</td>
<td>5.29</td>
<td>( \leq 75.2 )</td>
</tr>
<tr>
<td>ConvNetVAE (Yang et al. 2017)</td>
<td>10.0</td>
<td>( \leq 63.9 )</td>
</tr>
</tbody>
</table>
VAE Issues: Inference Gap (Cremer et al. 2018)

(2) Inference Gap

Ideally, \( q_{\text{enc}}(x)(z) \approx p_\theta(z \mid x) \)

\[
\text{Inference gap} = \text{KL}(q_{\text{enc}}(x)(z) \mid\mid p_\theta(z \mid x)) = \text{KL}(q_{\lambda^*}(z) \mid\mid p_\theta(z \mid x)) + \text{Approximation gap} + \text{Amortization gap}.
\]

- **Approximation gap**: Gap between true posterior and the best possible variational posterior \( \lambda^* \) within \( Q \)
- **Amortization gap**: Gap between the inference network posterior and best possible posterior
VAE Issues: Inference Gap (Cremer et al. 2018)

(2) Inference Gap

Ideally, \( q_{\text{enc}\phi}(x)(z) \approx p_\theta(z \mid x) \)

\[
\begin{align*}
\text{Inference gap} & = \text{Approximation gap} + \text{Amortization gap} \\
& = \text{Approximation gap} + \text{Amortization gap}
\end{align*}
\]

- **Approximation gap**: Gap between true posterior and the best possible variational posterior \( \lambda^* \) cwithin \( Q \)
- **Amortization gap**: Gap between the inference network posterior and best possible posterior
VAE Issues (Cremer et al. 2018)

- These gaps affect the learned generative model.
  - **Approximation gap**: use more flexible variational families, e.g. Normalizing/IA Flows (Rezende et al. 2015, Kingma et al. 2016) ➞ Has not been shown to fix posterior collapse on text.
  - **Amortization gap**: better optimize $\lambda$ for each data point, e.g. with iterative inference (Hjelm et al. 2016, Krishnan et al. 2018) ➞ Focus of this work.

- Does reducing the amortization gap allow us to employ powerful likelihood models while avoiding posterior collapse?
VAE Issues (Cremer et al. 2018)

- These gaps affect the learned generative model.
- **Approximation gap**: use more flexible variational families, e.g. Normalizing/IA Flows (Rezende et al. 2015, Kingma et al. 2016) ⇒ Has not been shown to fix posterior collapse on text.
- **Amortization gap**: better optimize $\lambda$ for each data point, e.g. with iterative inference (Hjelm et al. 2016, Krishnan et al. 2018) ⇒ Focus of this work.
- Does reducing the amortization gap allow us to employ powerful likelihood models while avoiding posterior collapse?
VAE Issues (Cremer et al. 2018)

- These gaps affect the learned generative model.

- **Approximation gap**: use more flexible variational families, e.g. Normalizing/IA Flows (Rezende et al. 2015, Kingma et al. 2016) ⇒ Has not been show to fix posterior collapse on text.

- **Amortization gap**: better optimize $\lambda$ for each data point, e.g. with iterative inference (Hjelm et al. 2016, Krishnan et al. 2018) ⇒ Focus of this work.

- Does reducing the amortization gap allow us to employ powerful likelihood models while avoiding posterior collapse?
VAE Issues (Cremer et al. 2018)

- These gaps affect the learned generative model.
- **Approximation gap**: use more flexible variational families, e.g. Normalizing/IA Flows (Rezende et al. 2015, Kingma et al. 2016) ⇒ Has not been show to fix posterior collapse on text.
- **Amortization gap**: better optimize λ for each data point, e.g. with iterative inference (Hjelm et al. 2016, Krishnan et al. 2018) ⇒ Focus of this work.
- Does reducing the amortization gap allow us to employ powerful likelihood models while avoiding posterior collapse?
Stochastic Variational Inference (SVI) (Hoffman et al. 2013)

- Amortization gap is mostly specific to VAE

- Stochastic Variational Inference (SVI):
  1. Randomly initialize $\lambda^{(i)}_0$ for each data point
  2. Perform iterative inference, e.g. for $k = 1, \ldots, K$
     \[
     \lambda^{(i)}_k \leftarrow \lambda^{(i)}_{k-1} - \alpha \nabla_{\lambda} \mathcal{L}(\lambda^{(i)}_k, \theta, x^{(i)})
     \]
     where $\mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_{\lambda}(z)}[-\log p_{\theta}(x | z)] + \text{KL}(q_{\lambda}(z) \parallel p(z)]$
  3. Update $\theta$ based on final $\lambda^{(i)}_K$, i.e.
     \[
     \theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}(\lambda^{(i)}_K, \theta, x^{(i)})
     \]
     (Can reduce amortization gap by increasing $K$)
Stochastic Variational Inference (SVI) (Hoffman et al. 2013)

- Amortization gap is mostly specific to VAE
- Stochastic Variational Inference (SVI):
  1. Randomly initialize $\lambda_0^{(i)}$ for each data point
  2. Perform iterative inference, e.g. for $k = 1, \ldots, K$
     \[
     \lambda_k^{(i)} \leftarrow \lambda_{k-1}^{(i)} - \alpha \nabla_\lambda \mathcal{L}(\lambda_{k}^{(i)}, \theta, x^{(i)})
     \]
     where $\mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_\lambda(z)}[-\log p_\theta(x | z)] + \text{KL}(q_\lambda(z) || p(z))$
  3. Update $\theta$ based on final $\lambda_K^{(i)}$, i.e.
     \[
     \theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\lambda_K^{(i)}, \theta, x^{(i)})
     \]
     (Can reduce amortization gap by increasing $K$)
Stochastic Variational Inference (SVI) (Hoffman et al. 2013)

- Amortization gap is mostly specific to VAE
- Stochastic Variational Inference (SVI):
  1. Randomly initialize $\lambda_{0}^{(i)}$ for each data point
  2. Perform iterative inference, e.g. for $k = 1, \ldots, K$
     
     $$
     \lambda_{k}^{(i)} \leftarrow \lambda_{k-1}^{(i)} - \alpha \nabla_{\lambda} \mathcal{L}(\lambda_{k}^{(i)}, \theta, x^{(i)})
     $$
     
     where $\mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_{\lambda}(z)}[-\log p_{\theta}(x \mid z)] + \text{KL}(q_{\lambda}(z) \mid \mid p(z))$
  3. Update $\theta$ based on final $\lambda_{K}^{(i)}$, i.e.
     
     $$
     \theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}(\lambda_{K}^{(i)}, \theta, x^{(i)})
     $$

(Can reduce amortization gap by increasing $K$)
Stochastic Variational Inference (SVI) (Hoffman et al. 2013)

- Amortization gap is mostly specific to VAE
- Stochastic Variational Inference (SVI):
  1. Randomly initialize $\lambda_0^{(i)}$ for each data point
  2. Perform iterative inference, e.g. for $k = 1, \ldots, K$
     \[
     \lambda_k^{(i)} \leftarrow \lambda_{k-1}^{(i)} - \alpha \nabla_{\lambda} \mathcal{L}(\lambda_k^{(i)}, \theta, x^{(i)})
     \]
     where $\mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_{\lambda}(z)}[- \log p_{\theta}(x | z)] + \text{KL}(q_{\lambda}(z) || p(z))$
  3. Update $\theta$ based on final $\lambda_K^{(i)}$, i.e.
     \[
     \theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}(\lambda_K^{(i)}, \theta, x^{(i)})
     \]
     (Can reduce amortization gap by increasing $K$)
Example: Text Modeling on Yahoo corpus (Yang et al. 2017)

Inference Network: LSTM + MLP
Generative Model: LSTM, \( z \) fed at each time step

<table>
<thead>
<tr>
<th>Model</th>
<th>KL</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language Model</td>
<td>−</td>
<td>61.6</td>
</tr>
<tr>
<td>VAE</td>
<td>0.01</td>
<td>( \leq 62.5 )</td>
</tr>
<tr>
<td>SVI ((K = 20))</td>
<td>0.41</td>
<td>( \leq 62.9 )</td>
</tr>
<tr>
<td>SVI ((K = 40))</td>
<td>1.01</td>
<td>( \leq 62.2 )</td>
</tr>
</tbody>
</table>
## Comparing the Amortized/Stochastic Variational Inference

<table>
<thead>
<tr>
<th></th>
<th>AVI</th>
<th>SVI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approximation Gap</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Amortization Gap</td>
<td>Yes</td>
<td>Minimal</td>
</tr>
<tr>
<td>Training/Inference</td>
<td>Fast</td>
<td>Slow</td>
</tr>
<tr>
<td>End-to-End Training</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

SVI: Trade-off between amortization gap vs speed
This Work: Semi-Amortized Variational Autoencoders

- Reduce amortization gap in VAEs by combining AVI/SVI
- Use inference network to initialize variational parameters, run SVI to refine them
- Maintain end-to-end training of VAEs by backpropagating through SVI to train the inference network/generative model
This Work: Semi-Amortized Variational Autoencoders

- Reduce amortization gap in VAEs by combining AVI/SVI
- Use inference network to initialize variational parameters, run SVI to refine them
- Maintain end-to-end training of VAEs by backpropagating through SVI to train the inference network/generative model
This Work: Semi-Amortized Variational Autoencoders

- Reduce amortization gap in VAEs by combining AVI/SVI
- Use inference network to initialize variational parameters, run SVI to refine them
- Maintain end-to-end training of VAEs by backpropagating through SVI to train the inference network/generative model
Semi-Amortized Variational Autoencoders (SA-VAE)

Forward step

1. \[ \lambda_0 = \text{enc}_\phi(x) \]

2. For \( k = 1, \ldots, K \)

   \[ \lambda_k \leftarrow \lambda_{k-1} - \alpha \nabla_\lambda \mathcal{L}(\lambda_k, \theta, x) \]

   where \( \mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_\lambda(z)}[-\log p_\theta(x | z)] + \text{KL}(q_\lambda(z) \| p(z)) \)

3. Final loss given by

   \[ L_K = \mathcal{L}(\lambda_K, \theta, x) \]
Semi-Amortized Variational Autoencoders (SA-VAE)

Forward step

1. \( \lambda_0 = \text{enc}_\phi(x) \)

2. For \( k = 1, \ldots, K \)

\[
\lambda_k \leftarrow \lambda_{k-1} - \alpha \nabla_\lambda \mathcal{L}(\lambda_k, \theta, x)
\]

where \( \mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_\lambda(z)}[- \log p_\theta(x | z)] + \text{KL}(q_\lambda(z) \| p(z)) \)

3. Final loss given by

\[
L_K = \mathcal{L}(\lambda_K, \theta, x)
\]
Semi-Amortized Variational Autoencoders (SA-VAE)

Forward step

1. $\lambda_0 = \text{enc}_\phi(x)$

2. For $k = 1, \ldots, K$

   $\lambda_k \leftarrow \lambda_{k-1} - \alpha \nabla \lambda \mathcal{L}(\lambda_k, \theta, x)$

   where $\mathcal{L}(\lambda, \theta, x) = \mathbb{E}_{q_\lambda(z)}[-\log p_\theta(x | z)] + \text{KL}(q_\lambda(z) \parallel p(z))$

3. Final loss given by

   $L_K = \mathcal{L}(\lambda_K, \theta, x)$
Semi-Amortized Variational Autoencoders (SA-VAE)

Backward step

- Need to calculate derivative of $L_K$ with respect to $\theta, \phi$
- But $\lambda_1, \ldots, \lambda_K$ are all functions of $\theta, \phi$

\[
\lambda_K = \lambda_{K-1} - \alpha \nabla_\lambda L(\lambda_{K-1}, \theta, x)
\]
\[
= \lambda_{K-2} - \alpha \nabla_\lambda L(\lambda_{K-2}, \theta, x)
\]
\[
- \alpha \nabla_\lambda L(\lambda_{K-2} - \alpha \nabla_\lambda L(\lambda_{K-2}, \theta, x), \theta, x)
\]
\[
= \lambda_{K-3} - \ldots
\]

- Calculating the total derivative requires “unrolling optimization” and backpropagating through gradient descent (Domke 2012, Maclaurin et al. 2015, Belanger et al. 2017).
Semi-Amortized Variational Autoencoders (SA-VAE)

Backward step

- Need to calculate derivative of $L_K$ with respect to $\theta, \phi$
- But $\lambda_1, \ldots \lambda_K$ are all functions of $\theta, \phi$

\[
\lambda_K = \lambda_{K-1} - \alpha \nabla_{\lambda} \mathcal{L}(\lambda_{K-1}, \theta, x) \\
= \lambda_{K-2} - \alpha \nabla_{\lambda} \mathcal{L}(\lambda_{K-2}, \theta, x) \\
- \alpha \nabla_{\lambda} \mathcal{L}(\lambda_{K-2} - \alpha \nabla_{\lambda} \mathcal{L}(\lambda_{K-2}, \theta, x), \theta, x) \\
= \lambda_{K-3} - \ldots
\]

- Calculating the total derivative requires “unrolling optimization” and backpropagating through gradient descent (Domke 2012, Maclaurin et al. 2015, Belanger et al. 2017).
Backpropagating through SVI

Simple example: consider just one step of SVI

1. $\lambda_0 = \text{enc}_\phi(x)$
2. $\lambda_1 = \lambda_0 - \alpha \nabla \lambda \mathcal{L}(\lambda_0, \theta, x)$
3. $L = \mathcal{L}(\lambda_1, \theta, x)$
Backpropagating through SVI

Backward step

1. Calculate \( \frac{dL}{d\lambda_1} \)

2. Chain rule:

\[
\frac{dL}{d\lambda_0} = \frac{d\lambda_1}{d\lambda_0} \frac{dL}{d\lambda_1} = \frac{d}{d\lambda_0} \left( \lambda_0 - \alpha \nabla_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

\[
= \left( I - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

Hessian matrix

\[
= \frac{dL}{d\lambda_1} - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \frac{dL}{d\lambda_1}
\]

Hessian-vector product

3. Backprop \( \frac{dL}{d\lambda_0} \) to obtain

\[
\frac{dL}{d\phi} = \frac{d\lambda_0}{d\phi} \frac{dL}{d\lambda_0} \quad \text{(Similar rules for } \frac{dL}{d\theta})
\]
Backpropagating through SVI

Backward step

1. Calculate \( \frac{dL}{d\lambda_1} \)

2. Chain rule:

\[
\frac{dL}{d\lambda_0} = \frac{d\lambda_1}{d\lambda_0} \frac{dL}{d\lambda_1} = \frac{d}{d\lambda_0} \left( \lambda_0 - \alpha \nabla_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

\[
= \left( I - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

3. Backprop \( \frac{dL}{d\lambda_0} \) to obtain

\[
\frac{dL}{d\phi} = \frac{d\lambda_0}{d\phi} \frac{dL}{d\lambda_0} \quad \text{(Similar rules for } \frac{dL}{d\theta})
\]
Backpropagating through SVI

Backward step

1. Calculate \( \frac{dL}{d\lambda_1} \)

2. Chain rule:

\[
\frac{dL}{d\lambda_0} = \frac{d\lambda_1}{d\lambda_0} \frac{dL}{d\lambda_1} = \frac{d}{d\lambda_0} \left( \lambda_0 - \alpha \nabla_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

\[
= \left( I - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

Hessian matrix

\[
= \frac{dL}{d\lambda_1} - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \frac{dL}{d\lambda_1}
\]

Hessian-vector product

3. Backprop \( \frac{dL}{d\lambda_0} \) to obtain \( \frac{dL}{d\phi} = \frac{d\lambda_0}{d\phi} \frac{dL}{d\lambda_0} \) (Similar rules for \( \frac{dL}{d\theta} \))
Backpropagating through SVI

Backward step

1. Calculate \( \frac{dL}{d\lambda_1} \)

2. Chain rule:

\[
\frac{dL}{d\lambda_0} = \frac{d\lambda_1}{d\lambda_0} \frac{dL}{d\lambda_1} = \frac{d}{d\lambda_0} \left( \lambda_0 - \alpha \nabla_{\lambda} \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

\[
= \left( \mathbf{I} - \alpha \nabla^2_{\lambda} \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}
\]

- **Hessian matrix**

\[
= \frac{dL}{d\lambda_1} - \alpha \nabla^2_{\lambda} \mathcal{L}(\lambda_0, \theta, x) \frac{dL}{d\lambda_1}
\]

- **Hessian-vector product**

3. Backprop \( \frac{dL}{d\lambda_0} \) to obtain \( \frac{dL}{d\phi} = \frac{d\lambda_0}{d\phi} \frac{dL}{d\lambda_0} \) (Similar rules for \( \frac{dL}{d\theta} \))
Backpropagating through SVI

Backward step

1. Calculate $\frac{dL}{d\lambda_1}$

2. Chain rule:

$$\frac{dL}{d\lambda_0} = \frac{d\lambda_1}{d\lambda_0} \frac{dL}{d\lambda_1} = \frac{d}{d\lambda_0} \left( \lambda_0 - \alpha \nabla_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}$$

$$= \left( I - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \right) \frac{dL}{d\lambda_1}$$

Hessian matrix

$$= \frac{dL}{d\lambda_1} - \alpha \nabla^2_\lambda \mathcal{L}(\lambda_0, \theta, x) \frac{dL}{d\lambda_1}$$

Hessian-vector product

3. Backprop $\frac{dL}{d\lambda_0}$ to obtain $\frac{dL}{d\phi} = \frac{d\lambda_0}{d\phi} \frac{dL}{d\lambda_0}$ (Similar rules for $\frac{dL}{d\theta}$)
Backpropagating through SVI

In practice:

- Estimate Hessian-vector products with finite differences (LeCun et al. 1993), which was more memory efficient.
- Clip gradients at various points (see paper).
## Summary

<table>
<thead>
<tr>
<th></th>
<th>AVI</th>
<th>SVI</th>
<th>SA-VAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approximation Gap</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Amortization Gap</td>
<td>Yes</td>
<td>Minimal</td>
<td>Minimal</td>
</tr>
<tr>
<td>Training/Inference</td>
<td>Fast</td>
<td>Slow</td>
<td>Medium</td>
</tr>
<tr>
<td>End-to-End Training</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Experiments: Synthetic data

Generate sequential data from a randomly initialized LSTM oracle

1. \( z_1, z_2 \sim \mathcal{N}(0, 1) \)
2. \( h_t = \text{LSTM}([x_t, z_1, z_2], h_{t-1}) \)
3. \( p(x_{t+1} \mid x_{\leq t}, z) \propto \exp(W h_t) \)

Inference network

- \( q(z_1), q(z_2) \) are Gaussians with learned means \( \mu_1, \mu_2 = \text{enc}_\phi(x) \)
- \( \text{enc}_\phi(\cdot) \): LSTM with MLP on final hidden state
Experiments: Synthetic data

Generate sequential data from a randomly initialized LSTM oracle

1. $z_1, z_2 \sim \mathcal{N}(0, 1)$
2. $h_t = \text{LSTM}([x_t, z_1, z_2], h_{t-1})$
3. $p(x_{t+1} | x_{\leq t}, z) \propto \exp(W h_t)$

Inference network

- $q(z_1), q(z_2)$ are Gaussians with learned means $\mu_1, \mu_2 = \text{enc}_\phi(x)$
- $\text{enc}_\phi(\cdot)$: LSTM with MLP on final hidden state
Experiments: Synthetic data

Oracle generative model (randomly-initialized LSTM)

(ELBO landscape for a random test point)
## Results: Synthetic Data

<table>
<thead>
<tr>
<th>Model</th>
<th>Oracle Gen</th>
<th>Learned Gen</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAE</td>
<td>≤ 21.77</td>
<td>≤ 27.06</td>
</tr>
<tr>
<td>SVI (K=20)</td>
<td>≤ 22.33</td>
<td>≤ 25.82</td>
</tr>
<tr>
<td>SA-VAE (K=20)</td>
<td>≤ 20.13</td>
<td>≤ 25.21</td>
</tr>
<tr>
<td>True NLL (Est)</td>
<td>19.63</td>
<td>–</td>
</tr>
</tbody>
</table>
Generative model:

1. $z \sim \mathcal{N}(0, I)$
2. $h_t = \text{LSTM}([x_t, z], h_{t-1})$
3. $x_{t+1} \sim p(x_{t+1} | x_{\leq t}, x) \propto \exp(W h_t)$

Inference network:

- $q(z)$ diagonal Gaussian with parameters $\mu, \sigma^2$
- $\mu, \sigma^2 = \text{enc}_\phi(x)$
- $\text{enc}_\phi(\cdot)$: LSTM followed by MLP
Two other baselines that combine AVI/SVI (but not end-to-end):

- **VAE+SVI 1 (Krishnan et al. 2018):**
  1. Update generative model based on $\lambda_K$
  2. Update inference network based on $\lambda_0$

- **VAE+SVI 2 (Hjelm et al. 2016):**
  1. Update generative model based on $\lambda_K$
  2. Update inference network to minimize $\text{KL}(q_{\lambda_0}(z) \parallel q_{\lambda_K}(z))$, treating $\lambda_K$ as a fixed constant.

(Forward pass is the same for both models)
Results: Text

Two other baselines that combine AVI/SVI (but not end-to-end):

- **VAE+SVI 1 (Krishnan et al. 2018):**
  1. Update generative model based on $\lambda_K$
  2. Update inference network based on $\lambda_0$

- **VAE+SVI 2 (Hjelm et al. 2016):**
  1. Update generative model based on $\lambda_K$
  2. Update inference network to minimize $\text{KL}(q_{\lambda_0}(z) \| q_{\lambda_K}(z))$, treating $\lambda_K$ as a fixed constant.

(Forward pass is the same for both models)
Two other baselines that combine AVI/SVI (but not end-to-end):

- **VAE+SVI 1 (Krishnan et al. 2018):**
  1. Update generative model based on $\lambda_K$
  2. Update inference network based on $\lambda_0$

- **VAE+SVI 2 (Hjelm et al. 2016):**
  1. Update generative model based on $\lambda_K$
  2. Update inference network to minimize $\text{KL}(q_{\lambda_0}(z) \parallel q_{\lambda_K}(z))$, treating $\lambda_K$ as a fixed constant.

(Forward pass is the same for both models)
## Results: Text (Yahoo corpus from Yang et al. 2017)

<table>
<thead>
<tr>
<th>Model</th>
<th>KL</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language Model</strong></td>
<td></td>
<td>61.6</td>
</tr>
<tr>
<td>VAE</td>
<td>0.01</td>
<td>≤ 62.5</td>
</tr>
<tr>
<td>VAE + Word-Drop 25%</td>
<td>1.44</td>
<td>≤ 65.6</td>
</tr>
<tr>
<td>VAE + Word-Drop 50%</td>
<td>5.29</td>
<td>≤ 75.2</td>
</tr>
<tr>
<td><strong>ConvNetVAE (Yang et al. 2017)</strong></td>
<td>10.0</td>
<td>≤ 63.9</td>
</tr>
<tr>
<td>SVI ($K = 20$)</td>
<td>0.41</td>
<td>≤ 62.9</td>
</tr>
<tr>
<td>SVI ($K = 40$)</td>
<td>1.01</td>
<td>≤ 62.2</td>
</tr>
<tr>
<td>VAE + SVI 1 ($K = 20$)</td>
<td>7.80</td>
<td>≤ 62.7</td>
</tr>
<tr>
<td>VAE + SVI 2 ($K = 20$)</td>
<td>7.81</td>
<td>≤ 62.3</td>
</tr>
<tr>
<td><strong>SA-VAE ($K = 20$)</strong></td>
<td>7.19</td>
<td>≤ 60.4</td>
</tr>
</tbody>
</table>
## Results: Text (Yahoo corpus from Yang et al. 2017)

<table>
<thead>
<tr>
<th>Model</th>
<th>KL</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language Model</strong></td>
<td>−</td>
<td>61.6</td>
</tr>
<tr>
<td><strong>VAE</strong></td>
<td>0.01</td>
<td>≤ 62.5</td>
</tr>
<tr>
<td><strong>VAE + Word-Drop 25%</strong></td>
<td>1.44</td>
<td>≤ 65.6</td>
</tr>
<tr>
<td><strong>VAE + Word-Drop 50%</strong></td>
<td>5.29</td>
<td>≤ 75.2</td>
</tr>
<tr>
<td><strong>ConvNetVAE (Yang et al. 2017)</strong></td>
<td>10.0</td>
<td>≤ 63.9</td>
</tr>
<tr>
<td><strong>SVI (K = 20)</strong></td>
<td>0.41</td>
<td>≤ 62.9</td>
</tr>
<tr>
<td><strong>SVI (K = 40)</strong></td>
<td>1.01</td>
<td>≤ 62.2</td>
</tr>
<tr>
<td><strong>VAE + SVI 1 (K = 20)</strong></td>
<td>7.80</td>
<td>≤ 62.7</td>
</tr>
<tr>
<td><strong>VAE + SVI 2 (K = 20)</strong></td>
<td>7.81</td>
<td>≤ 62.3</td>
</tr>
<tr>
<td><strong>SA-VAE (K = 20)</strong></td>
<td>7.19</td>
<td>≤ 60.4</td>
</tr>
</tbody>
</table>
## Results: Text (Yahoo corpus from Yang et al. 2017)

<table>
<thead>
<tr>
<th>Model</th>
<th>KL</th>
<th>PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language Model</strong></td>
<td>−</td>
<td>61.6</td>
</tr>
<tr>
<td>VAE</td>
<td>0.01</td>
<td>≤ 62.5</td>
</tr>
<tr>
<td>VAE + Word-Drop 25%</td>
<td>1.44</td>
<td>≤ 65.6</td>
</tr>
<tr>
<td>VAE + Word-Drop 50%</td>
<td>5.29</td>
<td>≤ 75.2</td>
</tr>
<tr>
<td>ConvNetVAE (Yang et al. 2017)</td>
<td>10.0</td>
<td>≤ 63.9</td>
</tr>
<tr>
<td>SVI ($K = 20$)</td>
<td>0.41</td>
<td>≤ 62.9</td>
</tr>
<tr>
<td>SVI ($K = 40$)</td>
<td>1.01</td>
<td>≤ 62.2</td>
</tr>
<tr>
<td>VAE + SVI 1 ($K = 20$)</td>
<td>7.80</td>
<td>≤ 62.7</td>
</tr>
<tr>
<td>VAE + SVI 2 ($K = 20$)</td>
<td>7.81</td>
<td>≤ 62.3</td>
</tr>
<tr>
<td>SA-VAE ($K = 20$)</td>
<td>7.19</td>
<td>≤ 60.4</td>
</tr>
</tbody>
</table>
Application to Image Modeling (OMNIGLOT)

$q_\phi(z \mid x)$: 3-layer ResNet (He et al. 2016)

$p_\theta(x \mid z)$: 12-layer Gated PixelCNN (van den Oord et al. 2016)

<table>
<thead>
<tr>
<th>Model</th>
<th>NLL (KL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gated PixelCNN</td>
<td>90.59</td>
</tr>
<tr>
<td>VAE</td>
<td>$\leq 90.43$ (0.98)</td>
</tr>
<tr>
<td>SVI ($K = 20$)</td>
<td>$\leq 90.51$ (0.06)</td>
</tr>
<tr>
<td>SVI ($K = 40$)</td>
<td>$\leq 90.44$ (0.27)</td>
</tr>
<tr>
<td>SVI ($K = 80$)</td>
<td>$\leq 90.27$ (1.65)</td>
</tr>
<tr>
<td>VAE + SVI 1 ($K = 20$)</td>
<td>$\leq 90.19$ (2.40)</td>
</tr>
<tr>
<td>VAE + SVI 2 ($K = 20$)</td>
<td>$\leq 90.21$ (2.83)</td>
</tr>
<tr>
<td>SA-VAE ($K = 20$)</td>
<td>$\leq 90.05$ (2.78)</td>
</tr>
</tbody>
</table>

(Amortization gap exists even with powerful inference networks)
Application to Image Modeling (OMNIGLOT)

$q_\phi (z | x)$: 3-layer ResNet (He et al. 2016)

$p_\theta (x | z)$: 12-layer Gated PixelCNN (van den Oord et al. 2016)

<table>
<thead>
<tr>
<th>Model</th>
<th>NLL (KL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gated PixelCNN</td>
<td>90.59</td>
</tr>
<tr>
<td>VAE</td>
<td>(\leq 90.43 (0.98))</td>
</tr>
<tr>
<td>SVI ((K = 20))</td>
<td>(\leq 90.51 (0.06))</td>
</tr>
<tr>
<td>SVI ((K = 40))</td>
<td>(\leq 90.44 (0.27))</td>
</tr>
<tr>
<td>SVI ((K = 80))</td>
<td>(\leq 90.27 (1.65))</td>
</tr>
<tr>
<td>VAE + SVI 1 ((K = 20))</td>
<td>(\leq 90.19 (2.40))</td>
</tr>
<tr>
<td>VAE + SVI 2 ((K = 20))</td>
<td>(\leq 90.21 (2.83))</td>
</tr>
<tr>
<td>SA-VAE ((K = 20))</td>
<td>(\leq 90.05 (2.78))</td>
</tr>
</tbody>
</table>

(Amortization gap exists even with powerful inference networks)
Limitations

- Requires $O(K)$ backpropagation steps of the generative model for each training setup: possible to reduce $K$ via
  - Learning to learn approaches
  - Dynamic scheduling
  - Importance sampling
- Still needs optimization hacks
  - Gradient clipping during iterative refinement
Train vs Test Analysis

Randomly Initialized $\lambda_0$

- VAE
- VAE+SVI
- VAE+SVI+KL
- SA-VAE
- SVI

Perplexity Upper Bound vs Inference Steps (at Test)
Train vs Test Analysis

\[ \lambda_0 = \text{enc}(\mathbf{x} ; \phi) \]

![Graph showing Perplexity Upper Bound vs Inference Steps (at Test) for different methods: VAE, VAE+SVI, VAE+SVI+KL, SA-VAE. The graph illustrates the performance improvement as the number of inference steps increases.]
Lessons Learned

- Reducing amortization gap helps learn generative models of text that give good likelihoods and maintains interesting latent representations.

- But certainly not the full story... still very much an open issue.

- So what are the latent variables capturing?
Lessons Learned

- Reducing amortization gap helps learn generative models of text that give good likelihoods and maintains interesting latent representations.
- But certainly not the full story... still very much an open issue.
- So what are the latent variables capturing?
Lessons Learned

- Reducing amortization gap helps learn generative models of text that give good likelihoods and maintains interesting latent representations.
- But certainly not the full story... still very much an open issue.
- So what are the latent variables capturing?
where can i buy an affordable stationary bike? try this place, they have every type imaginable with prices to match. http: UNK </s>
Test sentence in blue, two generations from $q(z \mid x)$ in red

<s> where can i buy an affordable stationary bike? try this place, they have every type imaginable with prices to match. http: UNK </s>

where can i find a good UNK book for my daughter? i am looking for a website that sells christmas gifts for the UNK. thanks! UNK UNK </s>

where can i find a good place to rent a UNK? i have a few UNK in the area, but i'm not sure how to find them. http: UNK </s>
Test sentence in blue, two generations from $q(z|x)$ in red

<s> where can i buy an affordable stationary bike? try this place, they have every type imaginable with prices to match. http: UNK </s>

where can i find a good UNK book for my daughter? i am looking for a website that sells christmas gifts for the UNK. thanks! UNK UNK </s>

where can i find a good place to rent a UNK? i have a few UNK in the area, but i'm not sure how to find them. http: UNK </s>
New sentence in blue, two generations from $q(z|x)$ in red

<s>which country is the best at soccer? brazil or germany.</s>

who is the best soccer player in the world? i think he is the best player in the world. ronaldinho is the best player in the world. he is a great player.</s>

will ghana be able to play the next game in 2010 fifa world cup? yes, they will win it all.</s>
which country is the best at soccer? brazil or germany.

who is the best soccer player in the world? i think he is the best player in the world. ronaldinho is the best player in the world. he is a great player.

will ghana be able to play the next game in 2010 fifa world cup? yes, they will win it all.
Saliency Analysis

Saliency analysis by Part-of-Speech Tag

![Graph showing saliency analysis by Part-of-Speech Tag]
Saliency Analysis

Saliency analysis by Position

![Bar chart showing saliency analysis by position. The x-axis represents position ranges (1-20, 21-40, 41-60, etc.), and the y-axis represents saliency values ranging from 0.0 to 0.6. The chart shows varying saliency across different position ranges.]

Saliency Analysis

Saliency analysis by Frequency
Saliency Analysis

Saliency analysis by PPL
Conclusion

- Reducing amortization gap helps learn generative models that better utilize the latent space.
- Can be combined with methods that reduce the approximation gap.