SurVAE Flows: Surjections to Bridge the Gap between VAEs and Flows

Didrik Nielsen¹, Priyank Jaini², Emiel Hoogeboom², Ole Winther¹, Max Welling²

Technical University of Denmark¹                      UvA-Bosch Delta Lab, University of Amsterdam²

NeurIPS 2020

Presenters: Keiran Paster, Andrew Li
SurVAE: Motivation

Variational Autoencoders (VAEs) [Kingma & Welling, 2013] and Normalizing Flows [Rezende & Mohamed, 2015] are two distinct approaches to generative modelling.

- Transform a simple prior distribution $p(z)$ to a complex data distribution $p(x)$

https://openai.com/blog/generative-models/
SurVAE: Motivation

Main Ideas:

- *Surjective* transformations to “bridge the gap” between VAEs and Normalizing Flows
- A framework in which VAEs, Normalizing Flows, and surjections are composable layers
Variational Autoencoders

- Models a stochastic generative process:
  \[ z \sim p_\theta(z), \quad x \sim p_\theta(x|z) \]
  Prior \hspace{1cm} Decoder

- The posterior \( p_\theta(z|x) \) involves an intractable integral and is approximated via a neural network:
  \[ q_\phi(z|x) \approx p_\theta(z|x) \]
  Encoder

Useful for optimization (to be shown)

[Kingma & Welling, 2019]
VAE Objective

- The goal is to maximize the likelihood of the data $\log p_\theta(x)$ but this is also intractable!

- Instead, a surrogate objective (ELBO) is used:

$$
\log p_\theta(x) = \mathbb{E}_{q_\phi(z|x)} \left[ \log \left( \frac{p_\theta(x, z)}{q_\phi(z|x)} \right) \right] + \mathbb{E}_{q_\phi(z|x)} \left[ \log \left( \frac{q_\phi(z|x)}{p_\theta(z|x)} \right) \right] + D_{KL}[q_\phi(z|x)\|p_\theta(z|x)]
$$
ELBO

- Note that
  \[
  \log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)} \left[ \log \left( \frac{p_\theta(x, z)}{q_\phi(z|x)} \right) \right] \quad \text{(ELBO)}
  \]

  and the better the approximation \( q_\phi(z|x) \approx p_\theta(z|x) \), the tighter the bound!

- **Main issue:** Many desirable quantities are intractable in the VAE framework, e.g. \( p_\theta(x) \), \( p_\theta(z|x) \).
Normalizing Flows

• Transform a simple distribution $p(z)$ into a more complicated distribution by composing deterministic, invertible transformations (bijectsions)

• Obtain the exact log-probability of any $x$:
  ○ Use change-of-variables formula: $p(x) = p(z)|\det \nabla_x f^{-1}(x)|$
  ○ Optimize the model to maximize likelihood of the data

• Flow layers ideally are expressive, invertible, and have an easily computable Jacobian determinant.

Normalizing Flow Layers

- Affine Coupling Layer (RealNVP - Dinh et al. 2017)
  - Input dimensions are split into two parts.
    \[
    y_{1:d} = x_{1:d} \\
    y_{d+1:D} = x_{d+1:D} \odot \exp(s(x_{1:d})) + t(x_{1:d})
    \]
  - Easy to invert and the Jacobian is convenient.

- Invertible 1x1 conv (Glow - Kingma and Dhariwal, 2018)

Issues with Normalizing Flows

- Transformations must be bijective.
  - Difficult to alter dimensionality
  - Issues mapping continuous latents to discrete data.

- Flow models still fall behind in image quality / log likelihood compared to other model types.

A bijective transformation. Figure from Nielsen et al 2020.
Comparison: VAEs and Flows

**VAEs** learn *stochastic* transformations $\mathcal{Z} \rightarrow \mathcal{X}$ and $\mathcal{X} \rightarrow \mathcal{Z}$.

- Intractable likelihood $p_\theta(x)$.

**Flows** learn *deterministic bijections* $\mathcal{Z} \rightarrow \mathcal{X}$ (and through inverting, $\mathcal{X} \rightarrow \mathcal{Z}$).

- Difficult to alter dimensionality
**VAE**
Very Deep VAE (Child, 2020)

**Normalizing Flow**
Glow (Kingma and Dhariwal, 2018)
SurVAE Flows

- Both VAEs and Flow models optimize the log likelihood of the data $\log p(x)$
  - Flow models optimize this likelihood exactly
  - VAEs optimize a lower bound (ELBO)
- Can we frame VAEs as a layer of a Flow model?
A Connection Between VAEs and Flows

**VAE:**

\[
\log p(x) = \mathbb{E}_{q(z|x)} [\log p(z)] + \mathbb{E}_{q(z|x)} \left[ \log \frac{p(x|z)}{q(z|x)} \right] + \mathbb{E}_{q(z|x)} \left[ \log \frac{q(z|x)}{p(z|x)} \right]
\]

Lik. contrib. $\mathcal{V}(x,z)$ 

Bound looseness $\mathcal{E}(x,z)$

**Normalizing Flow:**

\[
\log p(x) = \log p(z) + \log \left| \det \frac{\partial z}{\partial x} \right| , \quad z = f^{-1}(x)
\]

Lik. contrib. $\mathcal{V}(x,z)$ (change-of-variables)

[Nielsen et al 2020]
A Connection Between VAEs and Flows

- VAEs learn stochastic mappings while Flows learn deterministic mappings $\mathcal{Z} \rightarrow \mathcal{X}$.

- Dirac delta-function:
  \[ \delta(x) = \begin{cases} \infty, & x = 0 \\ 0, & x \neq 0 \end{cases} \]

  \[ \int_{-\infty}^{\infty} \delta(x) \, dx = 1 \]

- Can write a deterministic function $x = f(z)$ as a probability distribution with
  \[ p(x|z) = \delta(x - f(z)) \]
A Connection Between VAEs and Flows

• Let \( p(x|z) = \delta(x - f(z)) \)
  \[
p(z|x) = \delta(z - f^{-1}(x))
  \]
  \[
  q(z|x) = p(z|x)
  \]

• Then
  \[
  \log p(x) = \mathbb{E}_{q(z|x)} \left[ \log p(z) + \log \frac{p(x|z)}{q(z|x)} + \log \frac{q(z|x)}{p(z|x)} \right]
  \]
  \[
  = \log p(z) + \log |\det J|, \quad \text{for } z = f^{-1}(x),
  \]
where \( J^{-1} = \left. \frac{\partial f(z)}{\partial z} \right|_{z=f^{-1}(x)} \)
Surjective Transformations

- Next we consider *surjective* transformations with properties of both VAEs and Flows.

  \[ f : \mathcal{Z} \rightarrow \mathcal{X} \text{ is surjective if every } x \in \mathcal{X} \text{ has a pre-image } z \in \mathcal{Z} \text{ such that } f(z) = x \]

- Multiple inputs can map to the same output.

[Nielsen et al 2020]
Surjective Transformations

- Surjective transformations are *deterministic* forwards and can have *stochastic* inverses.

- Can also have surjections in the “inference” direction $\mathcal{X} \rightarrow \mathcal{Z}$. 

**Generative surjection**

**Inference surjection**
SurVAE Flows

- We just saw how Flow models are a special case of VAEs with a certain distribution.
- SurVAE Flows give a general framework for computing likelihood for different choices of forward and inverse transformations.

\[
\log p(x) \approx \log p(z) + \mathcal{V}(x, z) + \mathcal{E}(x, z)
\]

\[
\mathbb{E}_{q(z|x)} \left[ \log \frac{p(x|z)}{q(z|x)} \right] \geq 0
\]

---

**Algorithm 1: log – likelihood(\(x\))**

**Data:** \(x, p(z) & \{f_t\}_{t=1}^T\)  
**Result:** \(\mathcal{L}(x)\)

**for** \(t\) in range(\(T\)), **do**

**if** \(f_t\) is bijective **then**

\[
z = f_t^{-1}(x);
\]

\[
\mathcal{V}_t = \log |\det \frac{\partial z}{\partial x}|;
\]

**else if** \(f_t\) is stochastic **then**

\[
z \sim q_t(z|x);
\]

\[
\mathcal{V}_t = \log \frac{p_t(x|z)}{q_t(z|x)};
\]

\(x = z\);

**end**

**return** \(\log p(z) + \sum_{t=1}^T \mathcal{V}_t\)
Example: Rounding Surjection

- Consider the surjective transformation \( x = \lfloor z \rfloor \)

- The forward transformation is given by \( P(x|z) = \mathbb{I}(x = \lfloor z \rfloor) \)

- The backward transformation \( q(z|x) \) is stochastic with support over \( \{ x + u | u \in [0, 1)^d \} \)
Example: Rounding Surjection

- To find $\mathcal{V}(x, z)$, use $p(x|z) = \delta(x = \lfloor z \rfloor)$

- $\mathbb{E}_{q(z|x)} \left[ \log \frac{p(x|z)}{q(z|x)} \right] = \mathbb{E}_{q(z|x)} [\log p(x|z)] + \mathbb{E}_{q(z|x)} [\log q(z|x)]$

  Expectation of log delta fn = 0

- So, $\mathcal{V}(x, z) = \mathbb{E}_{q(z|x)} [-\log q(z|x)]$
Related Work

Many well-known methods in the literature can be expressed as SurVAE Flows.

- **Dequantization** (Uria et al, 2013; Ho et al, 2019) is used to train continuous flows on discrete data.
  - Can be obtained via the **Rounding Surjection**.

<table>
<thead>
<tr>
<th>Model</th>
<th>SurVAE Flow architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probabilistic PCA (Tipping and Bishop, 1999)</td>
<td>$Z \xrightarrow{\text{stochastic}} X$</td>
</tr>
<tr>
<td>VAE (Kingma and Welling, 2014; Rezende et al., 2014)</td>
<td>Dequantization (Uria et al., 2013; Ho et al., 2019)</td>
</tr>
<tr>
<td>Diffusion Models (Sohl-Dickstein et al., 2015; Ho et al., 2020)</td>
<td>ANFs, VFlow (Huang et al., 2020; Chen et al., 2020)</td>
</tr>
<tr>
<td></td>
<td>Multi-scale Architectures (Dinh et al., 2017)</td>
</tr>
<tr>
<td></td>
<td>CIFs, Discretely Indexed Flows, DeepGMMs (Cornish et al., 2019; Duan, 2019; Oord and Dambre, 2015)</td>
</tr>
<tr>
<td></td>
<td>RAD Flows (Dinh et al., 2019)</td>
</tr>
</tbody>
</table>
SurVAE Layers

Table 6: Summary of some generative surjection layers.

<table>
<thead>
<tr>
<th>Surjection</th>
<th>Forward</th>
<th>Inverse</th>
<th>$\mathcal{V}(x, z)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rounding</td>
<td>$x =</td>
<td>z</td>
<td>$</td>
</tr>
<tr>
<td>Slicing</td>
<td>$x = z_1$</td>
<td>$z_1 = x$, $z_2 \sim q(z_2</td>
<td>x)$</td>
</tr>
<tr>
<td>Abs</td>
<td>$s = \text{sign } x$</td>
<td>$s \sim \text{Bern}(\pi(x))$</td>
<td>$-\log q(s</td>
</tr>
<tr>
<td>Max</td>
<td>$k = \text{arg max } x$</td>
<td>$z_k = x$, $z_{k-1} \sim q(z_{k-1}</td>
<td>x, k)$</td>
</tr>
<tr>
<td>Sort</td>
<td>$\mathcal{I} = \text{argsort } z$</td>
<td>$\mathcal{I} \sim \text{Cat}(\pi(x))$</td>
<td>$-\log p(\mathcal{I}</td>
</tr>
<tr>
<td>ReLU</td>
<td>$x = \text{max}(z, 0)$ if $x = 0 : z \sim q(z)$, else $z = x$</td>
<td>$\mathbb{I}(x = 0)[-\log q(z)]$</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Summary of some inference surjection layers.

<table>
<thead>
<tr>
<th>Surjection</th>
<th>Forward</th>
<th>Inverse</th>
<th>$\mathcal{V}(x, z)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rounding</td>
<td>$x \sim p(x</td>
<td>z)$ where $x \in [z, z + 1]$</td>
<td>$z =</td>
</tr>
<tr>
<td>Slicing</td>
<td>$z_1 = z$, $z_2 \sim p(x_2</td>
<td>z)$</td>
<td>$z = x_1$</td>
</tr>
<tr>
<td>Abs</td>
<td>$s \sim \text{Bern}(\pi(z))$</td>
<td>$s = \text{sign } x$</td>
<td>$\log p(s</td>
</tr>
<tr>
<td>Max</td>
<td>$k \sim \text{Cat}(\pi(z))$</td>
<td>$k = \text{arg max } x$</td>
<td>$\log p(k</td>
</tr>
<tr>
<td>Sort</td>
<td>$\mathcal{I} \sim \text{Cat}(\pi(z))$</td>
<td>$\mathcal{I} = \text{argsort } x$</td>
<td>$\log p(\mathcal{I}</td>
</tr>
<tr>
<td>ReLU</td>
<td>if $z = 0 : x \sim p(x)$, else $z = x$</td>
<td>$z = \text{max}(x, 0)$</td>
<td>$\mathbb{I}(z = 0) \log p(x)$</td>
</tr>
</tbody>
</table>

(b) Surjective (Gen.)

(c) Surjective (Inf.)

Generative Surjection: stochastic $q(z|x)$

Inference Surjection: stochastic $p(x|z)$

Note: inference surjections have $\mathcal{E}(x, z) = 0$ i.e. exact likelihood.
Experiments - Symmetrical Synthetic Data

- Using an *absolute value inference surjection* improves the modeling of data with symmetries.
- Normalizing flows have trouble modelling disconnected structure in the data.
2D Visualization (Code Notebook)

- Normalizing flow combined with an absolute value surjection (last layer).
2D Visualization (Code Notebook)

- Composing a VAE layer and an abs. (inference) surjection.
Experiments - MaxPoolFlow for Image Data

MaxPooling is used to downscale from the image dimension to a smaller latent dimension.

Adding MaxPooling yields worse log-likelihoods but better inception and FID scores.

<table>
<thead>
<tr>
<th>Model</th>
<th>Inception</th>
<th>FID</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCGAN*</td>
<td>6.4</td>
<td>37.1</td>
</tr>
<tr>
<td>WGAN-GP*</td>
<td>6.5</td>
<td>36.4</td>
</tr>
<tr>
<td>PixelCNN*</td>
<td>4.60</td>
<td>65.93</td>
</tr>
<tr>
<td>PixelIQN*</td>
<td>5.29</td>
<td>49.46</td>
</tr>
<tr>
<td>Baseline (Ours)</td>
<td>5.08</td>
<td>49.56</td>
</tr>
<tr>
<td>MaxPoolFlow (Ours)</td>
<td><strong>5.18</strong></td>
<td><strong>49.03</strong></td>
</tr>
</tbody>
</table>

Table 5: Inception score and FID for CIFAR-10.
*Results taken from Ostrovski et al. (2018).

Figure 6: Flow architecture with max pooling. Surjections in green.

Figure 7: Samples from CIFAR-10 models. Top: MaxPoolFlow, Bottom: Baseline.
Summary

- SurVAE Flows is a framework in which VAEs and Normalizing Flows are modular layers.
  - Encompasses several additional works in the literature, e.g. *Dequantization*

- Surjective layers are introduced as a “bridge” between VAEs and Normalizing Flows
  - Can alter dimensionality
  - (In some cases) can compute exact likelihood

- Several novel, practical surjective layers are derived and introduced.
  - e.g. *MaxPool, AbsoluteValue, Sort*
  - Allows greater flexibility in designing Flow architectures.
  - **Limitation:** most of the introduced surjections are targeted to very specific functions and require extensive domain knowledge to be applied.
References