Residual Flows are ...

Highly scalable invertible generative model that allows free-form Jacobian and make use of unbiased log-likelihood.

(a) Det. Identities (Low Rank)  (b) Autoregressive (Lower Triangular)  (c) Coupling (Structured Sparsity)  (d) Unbiased Est. (Free-form)

Background: Invertible Generative Models

Maximum likelihood estimation. To perform maximum likelihood with stochastic gradient descent, we require estimating
\[ \nabla \mathbb{E}_{q_{\mathbf{z} \sim p(x; \theta)}} \log p(x) = \mathbb{E}_{q_{\mathbf{z} \sim p(x; \theta)}} \nabla \mathbb{E}_{q_{\mathbf{z} \sim p(x; \theta)}} \log p(x) \]

Change of Variables. With an invertible transformation \( f \), we can build a generative model
\[ z \sim p(z), \quad x = f^{-1}(z). \]

Then the log-density of \( x \) is given by
\[ \log p(x) = \log p(f(x)) + \log \left| \frac{df(x)}{dx} \right|. \]

Flow-based generative models can be 1. sampled, if (2) can be computed with arbitrary precision. 2. trained using maximum likelihood, if (3) can be unbiasedly estimated.

Background: Invertible Residual Networks

Residual networks are composed of simple transformations
\[ y = f(x) = x + g(x) \]

Behrmann et al. (2019) proved if \( g \) has Lipschitz strictly less than one, then the residual transformation (4) is invertible.

Sampling. The inverse \( f^{-1} \) can be computed by a fixed-point iteration
\[ x^{(i+1)} = y - g(x^{(i)}) \]

which converges superlinearly by the Banach fixed-point theorem.

Log-likelihood. The change of the variables can be applied.
\[ \log p(x) = \log p(f(x)) + \log \left| \sum_{k=1}^{\infty} \frac{(-1)^k+1}{k} \mathbb{P}(N \geq k) \right| \]

The infinite series is tractable to exactly compute. Fixed truncation creates a biased training objective.

Unbiased Log-likelihood via “Russian Roulette”

Russian roulette estimator. Used for estimating infinite series.
\[ \sum_{k=1}^{\infty} \Delta_k = \mathbb{E}_{x \sim p(x)} \left( \sum_{k=1}^{n} \frac{\Delta_k}{\mathbb{P}(N \geq k)} \right) \]

\[ \log p(x) = \log p(f(x)) + \mathbb{E}_{v, \mathbf{v}} \left( \sum_{k=1}^{n} \frac{(-1)^{k+1} v}{\mathbb{P}(N \geq k)} \mathbb{P}(N \geq k) \right), \]

where \( n \sim p(N) \) and \( v \sim \mathcal{N}(0, 1) \). We used a shifted geometric distribution for \( p(N) \) with an expected compute of 4 terms.

Compared to fixed truncation, this
- Allows making use of big networks and high Lipschitz constants.
- Allows training with higher dimensions (from 32 \( \times \) 32 to 256 \( \times \) 256).

Memory-efficient Gradient Estimation

Neumann gradient series. For estimating (1), we can either
(i) Estimate \( \log p(x) \), then take gradient.
(ii) Analytically compute the gradient power series, then estimate it.

The first option uses variable amount of memory as \( n \) is stochastic. The second option, by using a Neumann series we obtain constant memory cost:
\[ \frac{\partial}{\partial \theta} \log \left| \frac{df(x)}{dx} \right| = \mathbb{E}_{v, \mathbf{v}} \left( \sum_{k=0}^{n} \frac{(-1)^k v^k}{\mathbb{P}(N \geq k)} \mathbb{P}(N \geq k) \right) \frac{\partial f(x, \theta)}{\partial \theta}. \]

Now tractable to train with large networks.

Density Modeling Benchmarks

<table>
<thead>
<tr>
<th>Model</th>
<th>MNIST CIFAR-10 CIFAR-10*</th>
<th>Real NVP (Dinh et al., 2017)</th>
<th>Glow (Kingma &amp; Dhariwal, 2018)</th>
<th>FFJORD (Germain et al., 2016)</th>
<th>Flow++ (Ho et al., 2019)</th>
<th>i-ResNet (Behrmann et al., 2018)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3.40</td>
<td>4.09</td>
<td>3.81</td>
<td>3.66</td>
<td>3.39</td>
</tr>
<tr>
<td>Residual Flow</td>
<td></td>
<td>3.29 (3.09)</td>
<td>3.86</td>
<td>3.69</td>
<td>3.66</td>
<td>3.39</td>
</tr>
<tr>
<td>Both Combined</td>
<td></td>
<td>3.12</td>
<td>4.11</td>
<td>3.81</td>
<td>3.66</td>
<td>3.39</td>
</tr>
</tbody>
</table>

We also show residual blocks > coupling blocks for joint classification and generative modeling, ie. hybrid modeling.

Ablation Experiments

Table: Ablation results. *Larger network.

Qualitative Samples

Figure: Real (left) and random samples (right) trained on 5bit 64 \( \times \) 64 CelebA.

References

Kahn. “Use of different monte carlo sampling techniques.” (1955)