TRUMPETS: Injective Flows for Inference and Inverse Problems*

Konik Kothari[1] **AmirEhsan Khorashadizadeh**[2] **Maarten de Hoop**[3] **Ivan Dokmanić**[2]

[1]Coordinated Science Laboratory, University of Illinois at Urbana-Champaign
[2]Department of Mathematics and Computer Science, University of Basel
[3] Computational and Applied Mathematics, Rice University

A NETWORK ARCHITECTURE AND TRAINING DETAILS

We describe the injective portion of our network architecture that was used to train a CelebA dataset in Figure 1. The bijective revnet block has 3 bijective revnet steps in each block while the injective revnet block has just one injective revnet step which is explained in details in Section 2.1. The bijective part of our network is not shown in Figure 1 but it has 32 bijective revenet steps.

For the scale and bias terms of the coupling layer we used the U-Net architecture with 2 downsampling blocks and 2 corresponding upsampling blocks. Each resolution change is preceded by 2 convolution layers with 32 and 64 output channels. We choose the latent space dimension as 64 for MNIST, 256 for Chest X-ray dataset and 192 for all other datasets. We normalize the data to lie in $[-1, 1]$.

The number of training samples for CelebA, Chest X-ray, MNIST and CIFAR10 are 80000, 80000, 60000, and 50000 respectively. We trained all models for about 300 epochs with a batch size of 64.

All models are trained with Adam optimizer [Kingma and Ba, 2014] with learning rate 10^{-4}. $\gamma = 10^{-6}$ was used as the Tikhonov regularizer parameter for computing pseudoinverse of injective convolutional layers.

B DERIVATIONS OF ERROR AND LIKELIHOOD BOUNDS

B.1 BOUNDING LOG-LIKELIHOOD FOR INJECTIVE FUNCTIONS

Claim 1. *For an injective function $f = f_1 \circ f_2 \circ \dots f_k(z)$ that maps $z \in \mathbb{R}^d$ to $x \in \mathbb{R}^D$,*

$$\log|\det J_f^\mathsf{T} J_f| \leq \sum_{i=1}^{K} \log|\det J_{f_k}^\mathsf{T} J_{f_k}|$$

Proof. We demonstrate the claim for 3 layers; the general statement follows by induction. Consider $x = f(z) = f_1 \circ f_2 \circ f_3(z)$, where $x \in \mathbb{R}^D$ and $z \in \mathbb{R}^d$, $d < D$. Assume that $f_1 : \mathbb{R}^D \mapsto \mathbb{R}^D$, $f_3 : \mathbb{R}^d \mapsto \mathbb{R}^d$ are bijective and $f_2 : \mathbb{R}^d \mapsto \mathbb{R}^D$ is injective. Then

$$J_f = \underbrace{\frac{\partial f_1}{\partial f_2}}_{J_1} \underbrace{\frac{\partial f_2}{\partial f_3}}_{J_2} \underbrace{\frac{\partial f_3}{\partial z}}_{J_3}$$

and we have

$$\begin{aligned} \log\det|J_f^\mathsf{T} J_f| &= \log\det|J_3^\mathsf{T} J_2^\mathsf{T} J_1^\mathsf{T} J_1 J_2 J_3| \\ &= 2\log\det|J_3| + \log\det|J_2^\mathsf{T} J_1^\mathsf{T} J_1 J_2|. \end{aligned} \tag{1}$$

Let now $J_1 = U_1\Sigma_1 V_1^\mathsf{T}$ and $J_2 = U_2\Sigma_2 V_2^\mathsf{T}$. We can compute as

$$\begin{aligned} \log|\det J_2^\mathsf{T} J_1^\mathsf{T} J_1 J_2| &= \log|\det V_2\Sigma_2 U_2^\mathsf{T} V_1\Sigma_1 U_1^\mathsf{T} U_1\Sigma_1 V_1^\mathsf{T} U_2\Sigma_2 V_2^\mathsf{T}| \\ &= \log|V_2\Sigma_2 U_2^\mathsf{T} V_1\Sigma_1^2 V_1^\mathsf{T} U_2\Sigma_2 V_2^\mathsf{T}| \\ &= 2\log|\det\Sigma_2| + \log|\det V_2^\mathsf{T} U_1\Sigma_1^2 U_1^\mathsf{T} V_2| \\ &\leq 2\log|\det\Sigma_2| + 2\log|\det\Sigma_1| \\ &= \log|\det J_2^\mathsf{T} J_2| + \log|\det J_1^\mathsf{T} J_1| \end{aligned} \tag{2}$$

where we used that $\Pi_{i=1}^n \lambda_i(UHU^\mathsf{T}) \leq \Pi_{i=1}^n \lambda_i(H)$ for any symmetric matrix H and unitary matrix U (Horn [1950]). Here $\lambda_i(M)$ is the ith eigenvalue of matrix M.

*Code publicly available at `https://github.com/swing-research/trumpets`.

Supplement for the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2021).

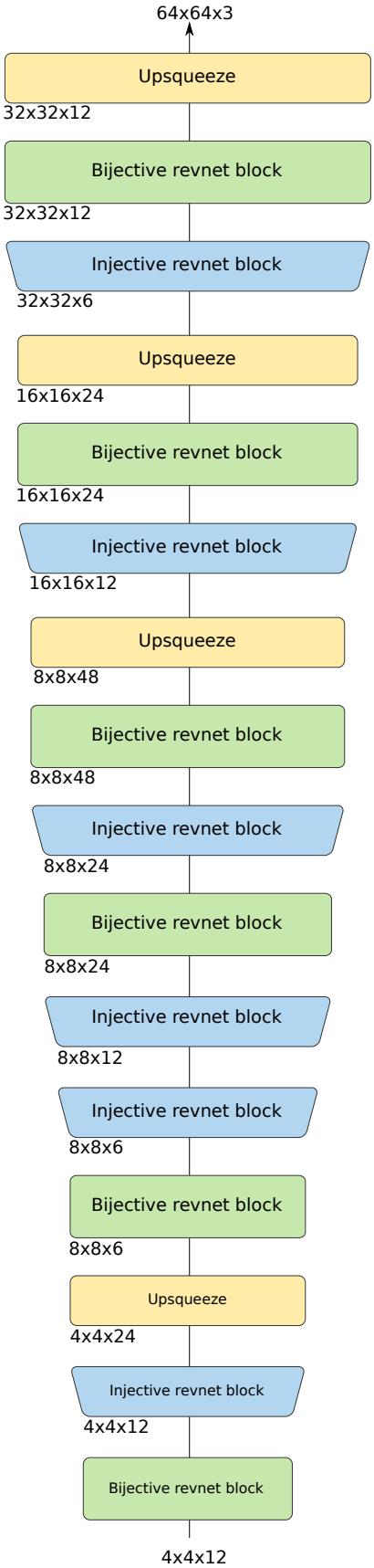


Figure 1: CelebA architecture for the injective portion g of TRUMPET. The input size to each layer is written below it.

Substituting (2) in (1) we obtain,

$$\log|\det J_f^\mathsf{T} J_f| \leq \sum_{i=1}^{3} \log|\det J_{f_k}^\mathsf{T} J_{f_k}|,$$

which establishes the claim. □

B.2 MEASURING ERROR DUE TO DEVIATIONS FROM RANGE

Claim 2. *Consider $y' = y + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 I)$, $y = \ell_w(x)$ and let $E_{\text{Inv}}(y') := \|\ell_w^\dagger(y') - x\|_2^2$ and the re-projection error $E_{\text{Proj}}(y') := \|\ell_w(\ell_w^\dagger(y')) - y'\|_2^2$. Then for both* ReLU *and linear variants of ℓ_w we have*

$$\mathbb{E}_\epsilon E_{\text{Inv}}(y') \propto \sigma_\epsilon^2 \sum_{i=1}^{c} \frac{1}{s_i(w)^2}, \qquad \mathbb{E}_\epsilon E_{\text{Proj}}(y') \propto \sigma_\epsilon^2,$$

where $s_i(w)$'s are the singular values of w and c is the number of input channels in the forward direction.

Proof. Consider $y' = y + \epsilon$, where $y = \ell_w(x)$ and $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 I_{2n})$. We consider a vectorized x and write the 1×1 convolution as a matrix-vector product, Wx say. For a ReLU injective convolution one could write the inverse as

$$x' = W^\dagger \begin{bmatrix} I_n & -I_n \end{bmatrix} y'. \tag{3}$$

We calculate $\mathbb{E}_\epsilon \|x' - x\|_2^2$. Let $M := \begin{bmatrix} I_n & -I_n \end{bmatrix}$ and $B := W^\dagger$, then

$$\begin{aligned} x' &= BM(y + \epsilon) \\ x' - x &= BM\epsilon, \end{aligned}$$

whence

$$\begin{aligned} \|x' - x\|_2^2 &= (BM\epsilon)^\mathsf{T} BM\epsilon \\ \|x' - x\|_2^2 &= \operatorname{Tr}\left(BM\epsilon(BM\epsilon)^\mathsf{T}\right) \\ \|x' - x\|_2^2 &= \operatorname{Tr}\left(BM\epsilon\epsilon^\mathsf{T} M^\mathsf{T} B^\mathsf{T}\right) \\ \|x' - x\|_2^2 &= \operatorname{Tr}\left(M^\mathsf{T} B^\mathsf{T} BM\epsilon\epsilon^\mathsf{T}\right) \end{aligned}$$

so that

$$\begin{aligned} \mathbb{E}_\epsilon \|x' - x\|_2^2 &= \mathbb{E}_\epsilon \operatorname{Tr}\left(M^\mathsf{T} B^\mathsf{T} BM\epsilon\epsilon^\mathsf{T}\right) \\ \mathbb{E}_\epsilon \|x' - x\|_2^2 &= \operatorname{Tr}\left(M^\mathsf{T} B^\mathsf{T} BM\right)\sigma_\epsilon^2 \\ \mathbb{E}_\epsilon \|x' - x\|_2^2 &= 2\operatorname{Tr}\left(B^\mathsf{T} B\right)\sigma_\epsilon^2 \\ \mathbb{E}_\epsilon \|x' - x\|_2^2 &= 2\sum_{i=1}^{c} s_i(w)^{-2}\sigma_\epsilon^2. \end{aligned}$$ □

Similarly for a linear layer the inverse is given as $x' = By'$. Therefore,

$$\begin{aligned} x' &= B(y + \epsilon) \\ x' - x &= B\epsilon \end{aligned}$$

whence

$$\|x' - x\|_2^2 = (B\epsilon)^\mathsf{T} B\epsilon$$
$$\|x' - x\|_2^2 = \mathrm{Tr}\left(B\epsilon(B\epsilon)^\mathsf{T}\right)$$
$$\|x' - x\|_2^2 = \mathrm{Tr}\left(B\epsilon\epsilon^\mathsf{T} B^\mathsf{T}\right)$$
$$\|x' - x\|_2^2 = \mathrm{Tr}\left(B^\mathsf{T} B\epsilon\epsilon^\mathsf{T}\right)$$

so that

$$\mathbb{E}_\epsilon \|x' - x\|_2^2 = \sum_{i=1}^{c} s_i(w)^{-2}\sigma_\epsilon^2. \qquad \square$$

The re-projection error for a ReLU layer is given as

$$\begin{aligned}
\sqrt{E_{\text{Proj}}(y')} &= \left\| \mathrm{ReLU}\left(\begin{bmatrix} W \\ -W \end{bmatrix} x'\right) - y \right\| \\
&= \left\| \mathrm{ReLU}\left(\begin{bmatrix} W \\ -W \end{bmatrix} x'\right) - \mathrm{ReLU}\left(\begin{bmatrix} W \\ -W \end{bmatrix} x\right) - \epsilon \right\| \\
&\leq \left\| \begin{bmatrix} W \\ -W \end{bmatrix} x' - \begin{bmatrix} W \\ -W \end{bmatrix} x \right\| + \|\epsilon\| \\
&= \left\| \begin{bmatrix} W \\ -W \end{bmatrix} (x + BM\epsilon) - \begin{bmatrix} W \\ -W \end{bmatrix} x \right\| + \|\epsilon\| \\
&= \left\| \begin{bmatrix} W \\ -W \end{bmatrix} BM\epsilon \right\| + \|\epsilon\| \\
&\leq (2\|WW^\dagger\|_F + 1)\|\epsilon\|
\end{aligned}$$

Squaring both sides, we get

$$E_{\text{Proj}}(y') = (2\sqrt{c} + 1)^2 \|\epsilon\|^2. \qquad \square$$

Similarly, for a linear layer we have

$$\begin{aligned}
E_{\text{Proj}}(y') &= \|Wx' - Wx - \epsilon\|^2 \\
&= \left\|WW^\dagger\epsilon - \epsilon\right\|^2 \\
&= (c+1)^2 \left\|\epsilon\right\|^2.
\end{aligned}$$

$\square$

B.3 LOG-DETERMINANTS OF JACOBIANS FOR RELU INJECTIVE CONVOLUTIONS

We vectorize x and, again, write the 1×1 convolution as a matrix-vector product Wx. Then, for a ReLU 1×1 convolution, we have

$$y = \mathrm{ReLU}\left(\begin{bmatrix} W \\ -W \end{bmatrix}\right) x.$$

This could be trivially rewritten as $y = W'x$, where the rows of W' are $w'_i = w_i$ if $\langle w_i, x\rangle > 0$ and $w'_i = -w_i$ otherwise. We note that changing the row signs does not change $|\det W|$. Hence, for such a ReLU injective convolutional layer, $\ell_w \log|\det J_{\ell_w}^T J_{\ell_w}| = \sum_{i=1}^{c} s_i^2(w)$, where $s_i(w)$'s are the singular values of w, where w is the 1×1 kernel corresponding to the convolution matrix W.

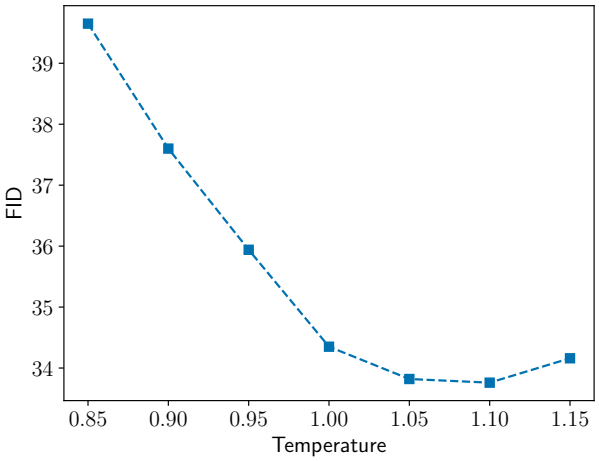


Figure 2: FID score of TRUMPET with sampling temperature.

C SAMPLES

In Figures 3a, 3b and Figures 4a, 4b we compare the performance of TRUMPETs trained with ReLU and linear injective convolutions on the MNIST and 64×64 CelebA datsets. Both variants offer similar performance hence we choose to use linear convolutions for the rest of our results regarding inverse problems and uncertainty quantification. In Figures 6 and 5 we show generated samples from TRUMPET and a few reconstructions of original samples, x given as $f(f^\dagger(x))$ on the CIFAR10 and Chest X-ray datasets respectively. For the CIFAR10 dataset, we do see a low frequency bias in the generated samples. We hope to rectify this as per our discussions in Section 6. For other datasets the low-frequency bias seems to be less of a problem. In fact, on these datasets TRUMPETs outperform previous injective variants of flows [Brehmer and Cranmer, 2020, Kumar et al., 2020].

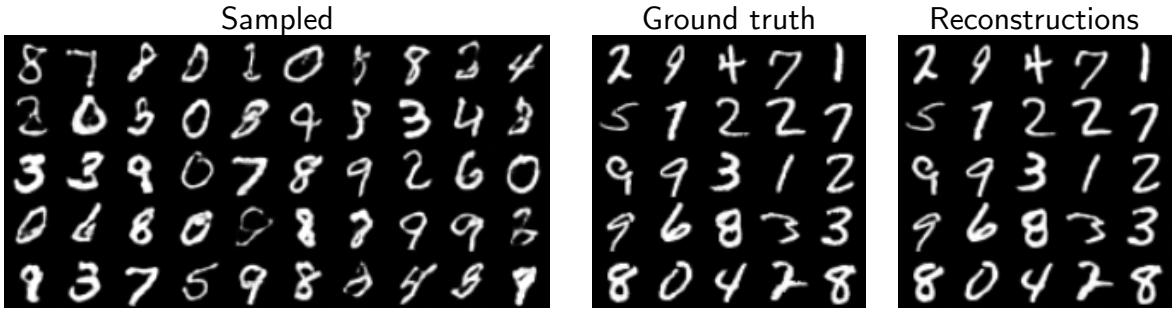
The temperature of sampling has a significant effect on the FID scores as shown in Figure 2. While samples in Figures 4a, 4b are for $T = 1$ we share some samples in Figure 7 for $T = 0.85$.

References

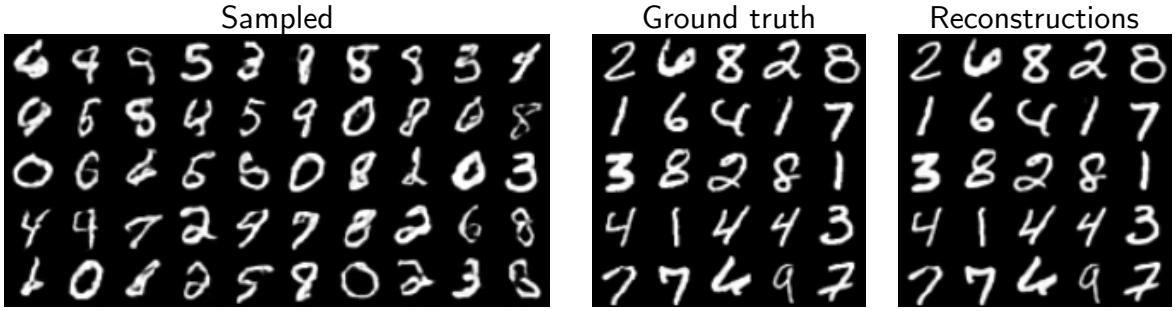
Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. *arXiv preprint arXiv:2003.13913*, 2020.

Alfred Horn. On the singular values of a product of completely continuous operators. *Proceedings of the National Academy of Sciences of the United States of America*, 36 (7):374, 1950.

Diederik P Kingma and Jimmy Ba. Adam: A method for



(a) ReLU 1×1 convolutions



(b) Linear 1×1 convolutions

Figure 3: TRUMPETs trained with (a) ReLU and (b) linear 1×1 convolutions give similar sample quality.

stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Abhishek Kumar, Ben Poole, and Kevin Murphy. Regularized autoencoders via relaxed injective probability flow. *arXiv preprint arXiv:2002.08927*, 2020.

(a) ReLU 1×1 convolutions

(b) Linear 1×1 convolutions

Figure 4: TRUMPETs trained with (a) ReLU and (b) linear 1×1 convolutions give similar sample quality. On the right, we showcase the reconstruction performance—the left column is ground truth and the right is our reconstruction (see Table 2 for quantitative assessment)

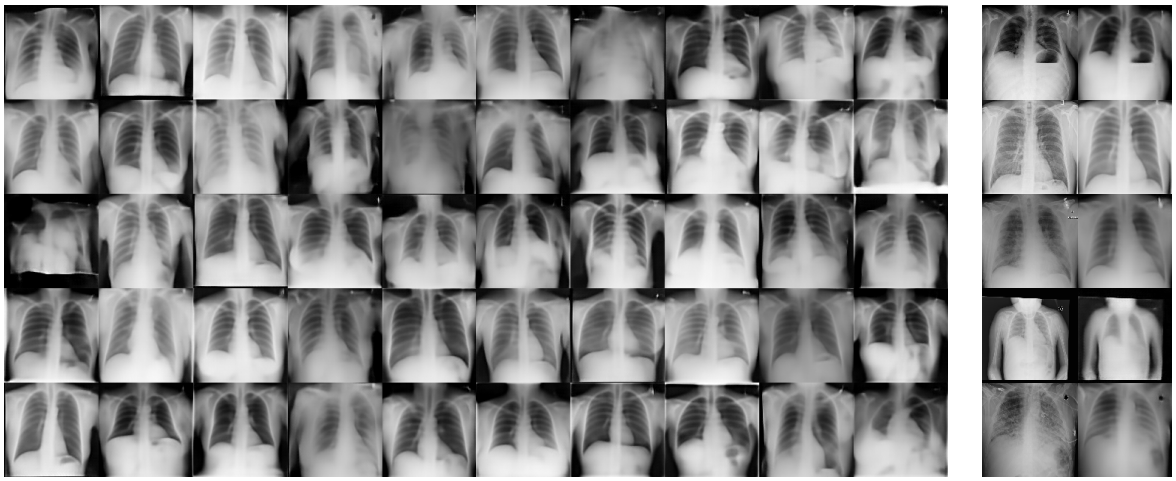


Figure 5: Generated samples on the Chest X-ray. On the right, we showcase the reconstruction performance—the left column is ground truth and the right is our reconstruction (see Table 2 for quantitative assessment)

Figure 6: Generated samples and reconstructions of original data on the CIFAR-10 dataset.

Figure 7: Generated samples on the celeba dataset with linear 1×1 convolution and $T = 0.85$.