Supplementary Materials

Paper ID: 10482

1 Structure of jump-diffusion based classifiers

1.1 Basic structure

Throughout all experiments in this paper, we adapt the network structure from Neural ODE, their code is available at https://github.com/rtqichen/torchdiffeq/blob/master/examples/odenet_mnist.py. Our own code will be publicly available after the reviewing process.

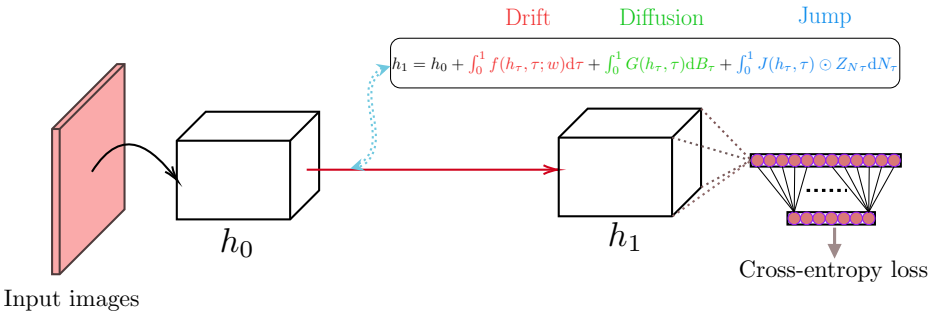

Figure 1: The network architecture. It can be divided to three segments. The first segment (downsampling block) is from input images to h_0; the second segment (jump-diffusion block) works from h_0 to h_1; and the last segment (classification block) process h_1 to the final classification loss.

The whole classifier consists of three parts: the downsampling block; the neural jump-diffusion block; and the classification block. To illustrate the architecture in Fig. 1. The downsampling layer is to preprocess the raw images and downsample it to smaller size. The jump-diffusion block is the most important part, which can be regarded as the continuous version of ResNet with random layers. Finally the classification block is just a fully connected layer combined with softmax nonlinearity. Compared with the original Neural ODE, we make more flexible choices of different blocks to accommodate different datasets:

- We made two different kinds of downsampling blocks, namely "`light`" and "`heavy`". The `light` block has fewer convolutional layers and fewer parameters. For simple dataset such as MNIST (not shown in this paper), using `light` downsampling block saves memory and trains faster.
- For the drift function, we also have three choices to encode the current depth t. Recall the encoding method in Neural ODE is just concatenating t to hidden features. We call this encoding method as `scalar-coding`. Another way of coding depth t is similar to the "time encoding" method in Transformer model [9], specifically we choose the dimension d (typically 64) and calculate

$$x_{2i} = \sin\left(\frac{t}{100^{2i/d}}\right), \text{ and } x_{2i+1} = \cos\left(\frac{t}{100^{2i/d}}\right). \tag{1}$$

  This is done for all $t \in [0, 1]$. The direct consequence of using such `vector-coding` is a much larger hidden dimension, allowing the network to have larger capacity. The last depth encoding

method is `null-coding`, which does not encode the depth information at all. Doing `null-coding` is effectively making the system to be autonomous system.

- For the diffusion function, as mentioned in the main text, we tried four kinds of noises `Additive Gaussian`, `Multiplicative Gaussian`, `Dropout Gaussian`.
- For the jump function, we only tested `Dropout` in this paper, implementing random depth network [4] will be as easy as binding all Bernoulli random variables in `Dropout`.

1.2 Multi-scale flows

The basic architecture shown above has only one jump-diffusion model working on just one resolution of hidden features $\boldsymbol{h}_0$. In order to improve the performance on real data, we borrow the idea of multi-scale architecture from many flow-based generative models (such as RealNVP [2], Glow [5], and FFJORD [3]), as well as autoregressive models such as PixelCNN [8]). We illustrate this architecture in Fig. 2.

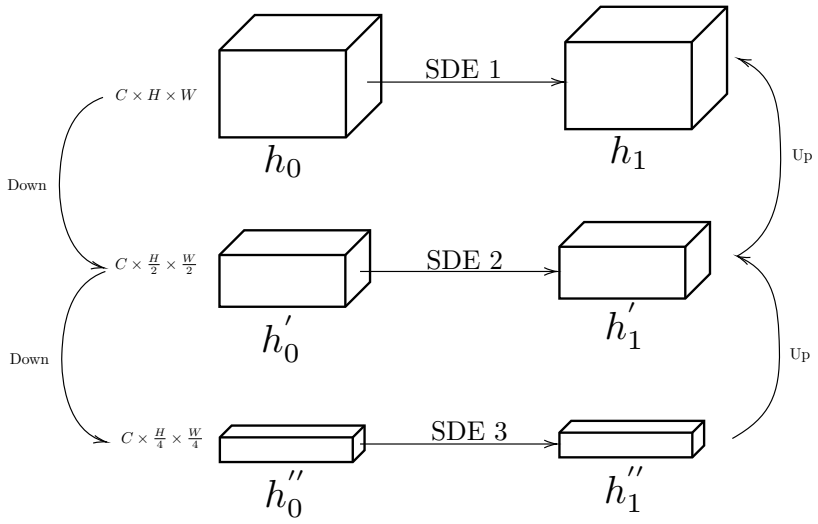

Figure 2: Our multi-scale architecture. In this figure we showcase three resolutions $\{H \times W, \frac{H}{2} \times \frac{W}{2}, \frac{H}{4} \times \frac{W}{4}\}$. The inputs are first downsampled to the desired spatial resolution $\frac{H}{2^k} \times \frac{W}{2^k}$, $k = 0, 1, 2$. For each resolution, there is an independent jump-diffusion model (SDE1, SDE2, SDE3) to process the hidden states, and the final results are then upscaled to the initial resolution $H \times W$.

2 Omitted proofs

We include the omitted proofs, experiment details and performance analysis here.

3 Omitted Theorems and Proofs

Theorem 3.1. [6] *If there exists a non-negative real valued function $V(\boldsymbol{\varepsilon}, t)$ defined on $\mathbb{R}^n \times \mathbb{R}_+$ that has continuous partial derivatives*

$$V_1(\boldsymbol{\varepsilon}, t) := \frac{\partial V(\boldsymbol{\varepsilon}, t)}{\partial \boldsymbol{\varepsilon}}, V_2(\boldsymbol{\varepsilon}, t) := \frac{\partial V(\boldsymbol{\varepsilon}, t)}{\partial t}, V_{1,1}(\boldsymbol{\varepsilon}, t) := \frac{\partial^2 V(\boldsymbol{\varepsilon}, t)}{\partial \boldsymbol{\varepsilon} \partial \boldsymbol{\varepsilon}^\top}$$

and constants $p > 0, c_1 > 0, c_2 \in \mathbb{R}, c_3 \geq 0$ such that the following inequalities hold:

1. $c_1\|\boldsymbol{\varepsilon}\|^p \le V(\boldsymbol{\varepsilon}, t)$
2. $\mathcal{L}V(\boldsymbol{\varepsilon}, t) = V_2(\boldsymbol{\varepsilon}, t) + V_1(\boldsymbol{\varepsilon}, t)\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t) + \frac{1}{2}\mathsf{Tr}[\boldsymbol{G}_\Delta^\top(\boldsymbol{\varepsilon}, t)V_{1,1}(\boldsymbol{\varepsilon}, t)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)] \le c_2 V(\boldsymbol{\varepsilon}, t)$
3. $\|V_1(\boldsymbol{\varepsilon}, t)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\|^2 \ge c_3 V^2(\boldsymbol{\varepsilon}, t)$

for all $\boldsymbol{\varepsilon} \neq \mathbf{0}$ and $t > 0$. Then for all $\boldsymbol{\varepsilon}_0 \in \mathbb{R}^n$,

$$\limsup_{t\to\infty} \frac{1}{t}\log\|\boldsymbol{\varepsilon}_t\| \le -\frac{c_3 - 2c_2}{2p} \quad a.s. \tag{2}$$

In particular, if $c_3 \ge 2c_2$, the solution $\boldsymbol{\varepsilon}_t \equiv \mathbf{0}$ is almost surely exponentially stable.

We present the proofs of theorems on stability of SDE. The proofs are adapted from [6]. We start with two crucial lemmas.

Lemma 3.2. *If $\boldsymbol{f}, \boldsymbol{G}$ satisfy Assumption 2, then $\boldsymbol{f}_\Delta, \boldsymbol{G}_\Delta$ satisfy Assumption 1, 2.*

Proof. By Assumption 2 on $\boldsymbol{f}, \boldsymbol{G}$, we can obtain that for any $\boldsymbol{\varepsilon}, \tilde{\boldsymbol{\varepsilon}} \in \mathbb{R}^n, t \ge 0$

$$\begin{aligned}
&\|\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t)\| + \|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\| \le c_2\|\boldsymbol{\varepsilon}\| \le c_2(1 + \|\boldsymbol{\varepsilon}\|),\\
&\|\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t) - \boldsymbol{f}_\Delta(\tilde{\boldsymbol{\varepsilon}}, t)\| + \|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t) - \boldsymbol{G}_\Delta(\tilde{\boldsymbol{\varepsilon}}, t)\| \le c_2\|\boldsymbol{\varepsilon} - \tilde{\boldsymbol{\varepsilon}}\|.
\end{aligned}$$

This guarantees the uniqueness of the solution of SDE (13) in the main paper. □

Lemma 3.3. *For SDE (13), whenever $\boldsymbol{\varepsilon}_0 \neq \mathbf{0}$, $\Pr\{\boldsymbol{\varepsilon}_t \neq \mathbf{0}$ for all $t \ge 0\} = 1$.*

Proof. We prove it by contradiction. Let $\tau = \inf\{t \ge 0 : \boldsymbol{\varepsilon}_t = 0\}$. Then if it is not true, there exists some $\boldsymbol{\varepsilon}_0 \neq \mathbf{0}$ such that $\Pr\{\tau < \infty\} > 0$. Therefore, we can find sufficiently large constant $T > 0$ and $\theta > 1$ such that $\Pr(A) := \Pr\{\tau < T$ and $|\boldsymbol{\varepsilon}_t| \le \theta - 1,\ \forall\, 0 \le t \le \tau\} > 0$. By Assumption 2 on $\boldsymbol{f}$ and $\boldsymbol{G}$, there exists a positive constant K_θ such that

$$\|\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t)\| + \|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\| \le K_\theta\|\boldsymbol{\varepsilon}\|, \quad \text{for all } \|\boldsymbol{\varepsilon}\| \le \theta \text{ and } 0 \le t \le T. \tag{3}$$

Let $V(\boldsymbol{\varepsilon}, t) = \|\epsilon\|^{-1}$. Then, for any $0 \le \|\boldsymbol{\varepsilon}\| \le \theta$ and $0 \le t \le T$, we have

$$\begin{aligned}
\mathcal{L}V(\boldsymbol{\varepsilon}, t) &= -\|\boldsymbol{\varepsilon}\|^{-3}\boldsymbol{\varepsilon}^\top\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t) + \frac{1}{2}\{-\|\boldsymbol{\varepsilon}\|^{-3}\|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\|^2 + 3\|\boldsymbol{\varepsilon}\|^{-5}\|\boldsymbol{\varepsilon}^\top\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\|^2\}\\
&\le \|\boldsymbol{\varepsilon}\|^{-2}\|\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t)\| + \|\boldsymbol{\varepsilon}\|^{-3}\|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\|^2\\
&\le K_\theta\|\boldsymbol{\varepsilon}\|^{-1} + K_\theta^2\|\boldsymbol{\varepsilon}\|^{-1} = K_\theta(1 + K_\theta)V(\boldsymbol{\varepsilon}, t),
\end{aligned} \tag{4}$$

where the first inequality comes from Cauchy-Schwartz and the last one comes from (3). For any $\delta \in (0, \|\boldsymbol{\varepsilon}_0\|)$, we define the stopping time $\tau_\delta := \inf\{t \ge 0 : \|\boldsymbol{\varepsilon}_t\| \notin (\delta, \theta)\}$. Let $\nu_\delta = \min\{\tau_\delta, T\}$. By Itô's formula, $\mathbb{E}\left[e^{-K_\theta(1+K_\theta)\nu_\delta}V(\boldsymbol{\varepsilon}_{\nu_\delta}, \nu_\delta)\right]$

$$= V(\boldsymbol{\varepsilon}_0, 0) + \mathbb{E}\int_0^{\nu_\delta} e^{-K_\theta(1+K_\theta)s}\Big[-K_\theta(1 + K_\theta)V(\boldsymbol{\varepsilon}_s, s) + \mathcal{L}V(\boldsymbol{\varepsilon}_s, s)\Big]\,\mathrm{d}s \le \|\boldsymbol{\varepsilon}_0\|^{-1}. \tag{5}$$

Since $\tau_\delta \le T$ and $\|\boldsymbol{\varepsilon}_{\tau_\delta}\| = \delta$ for any $\omega \in A$, then (5) implies

$$\mathbb{E}\left[e^{-K_\theta(1+K_\theta)T}\delta^{-1}\mathbf{1}_A\right] = \delta^{-1}e^{-K_\theta(1+K_\theta)T}\Pr(A) \le \|\boldsymbol{\varepsilon}_0\|^{-1}. \tag{6}$$

Thus, $\Pr(A) \le \delta\|\boldsymbol{\varepsilon}_0\|^{-1}e^{K_\theta(1+K_\theta)T}$. Letting $\delta \to 0$, we obtain $\Pr(A) = 0$, which leads to a contradiction. □

Proof of Theorem 3.1

We then prove Theorem 3.1. Clearly, (2) holds for $\boldsymbol{\varepsilon}_0 = \mathbf{0}$ since $\boldsymbol{\varepsilon}_t \equiv \mathbf{0}$. For any $\boldsymbol{\varepsilon}_0 \neq \mathbf{0}$, we have $\boldsymbol{\varepsilon}_t \neq \mathbf{0}$ for all $t \geq 0$ almost surely by Lemma 3.3. Thus, by applying Itô's formula and condition (2), we can show that for $t \geq 0$,

$$\log V(\boldsymbol{\varepsilon}_t, t) \leq \log V(\boldsymbol{\varepsilon}_0, 0) + c_2 t + M(t) - \frac{1}{2}\int_0^t \frac{|V_1(\boldsymbol{\varepsilon}_s, s)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}_s, s)|^2}{V^2(\boldsymbol{\varepsilon}_s, s)}\,\mathrm{d}s. \tag{7}$$

where $M(t) = \int_0^t \frac{V_1(\boldsymbol{\varepsilon}_s,s)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}_s,s)}{V(\boldsymbol{\varepsilon}_s,s)}\,\mathrm{d}\mathbf{B}_s$ is a continuous martingale with initial value $M(0) = 0$. By the exponential martingale inequality, for any arbitrary $\alpha \in (0,1)$ and $n = 1, 2, \cdots$, we have

$$\Pr\left\{\sup_{0\leq t\leq n}\left[M(t) - \frac{\alpha}{2}\int_0^t \frac{|V_1(\boldsymbol{\varepsilon}_s, s)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}_s, s)|^2}{V^2(\boldsymbol{\varepsilon}_s, s)}\,\mathrm{d}s\right] > \frac{2}{\alpha}\log n\right\} \leq \frac{1}{n^2}. \tag{8}$$

Applying Borel-Cantelli lemma, we can get that for almost all $\omega \in \Omega$, there exists an integer $n_0 = n_0(\omega)$ such that if $n \geq n_0$,

$$M(t) \leq \frac{2}{\alpha}\log n + \frac{\alpha}{2}\int_0^t \frac{|V_1(\boldsymbol{\varepsilon}_s, s)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}_s, s)|^2}{V^2(\boldsymbol{\varepsilon}_s, s)}\,\mathrm{d}s, \quad \forall\, 0 \leq t \leq n. \tag{9}$$

Combining (7), (9) and condition (3), we can obtain that

$$\log V(\boldsymbol{\varepsilon}_t, t) \leq \log V(\boldsymbol{\varepsilon}_0, 0) - \frac{1}{2}[(1-\alpha)c_3 - 2c_2]t + \frac{2}{\alpha}\log n. \tag{10}$$

for all $0 \leq t \leq n$ and $n \geq n_0$ almost surely. Therefore, for almost all $\omega \in \Omega$, if $n - 1 \leq t \leq n$ and $n \geq n_0$, we have

$$\frac{1}{t}\log V(\boldsymbol{\varepsilon}_t, t) \leq -\frac{1}{2}[(1-\alpha)c_3 - 2c_2] + \frac{\log V(\boldsymbol{\varepsilon}_0, 0) + \frac{2}{\alpha}\log n}{n-1} \tag{11}$$

which consequently implies

$$\limsup_{t\to\infty}\frac{1}{t}\log V(\boldsymbol{\varepsilon}_t, t) \leq -\frac{1}{2}[(1-\alpha)c_3 - 2c_2]) \quad a.s. \tag{12}$$

With condition (1) and arbitrary choice of $\alpha \in (0,1)$, we can obtain (2).

Proof of Corollary 4.0.1

We apply Theorem 3.1 to establish the theories on stability of SDE (16). Note that $\boldsymbol{f}(\boldsymbol{h}_t, t; \boldsymbol{w})$ is L-Lipschitz continuous w.r.t $\boldsymbol{h}_t$ and $\boldsymbol{G}(\boldsymbol{h}_t, t; \boldsymbol{v}) = \sigma\boldsymbol{h}_t, m = 1$. Then, SDE (16) has a unique solution, with $\boldsymbol{f}_\Delta$ and $\boldsymbol{G}_\Delta$ satisfying Assumption 1, 2,

$$\begin{aligned}
&\|\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}_t, t)\| + \|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}_t, t)\| \leq \max\{L, \sigma\}\|\boldsymbol{\varepsilon}_t\| \leq \max\{L, \sigma\}(1 + \|\boldsymbol{\varepsilon}_t\|),\\
&\|\boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}_t, t) - \boldsymbol{f}_\Delta(\tilde{\boldsymbol{\varepsilon}}_t, t)\| + \|\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}_t, t) - \boldsymbol{G}_\Delta(\tilde{\boldsymbol{\varepsilon}}_t, t)\| \leq \max\{L, \sigma\}\|\boldsymbol{\varepsilon}_t - \tilde{\boldsymbol{\varepsilon}}_t\|.
\end{aligned}$$

To apply Theorem 3.1, let $V(\boldsymbol{\varepsilon}, t) = \|\boldsymbol{\varepsilon}\|^2$. Then,

$$\begin{aligned}
&\mathcal{L}V(\boldsymbol{\varepsilon}, t) = 2\boldsymbol{\varepsilon}^\top \boldsymbol{f}_\Delta(\boldsymbol{\varepsilon}, t) + \sigma^2\|\boldsymbol{\varepsilon}\|^2 \leq (2L + \sigma^2)\|\boldsymbol{\varepsilon}\|^2 = (2L + \sigma^2)V(\boldsymbol{\varepsilon}, t),\\
&\|V_1(\boldsymbol{\varepsilon}, t)\boldsymbol{G}_\Delta(\boldsymbol{\varepsilon}, t)\|^2 = 4\sigma^2 V(\boldsymbol{\varepsilon}, t)^2.
\end{aligned}$$

Let $c_1 = 1, p = 2, c_2 = 2L + \sigma^2, c_3 = 4\sigma^2$. By Theorem 3.1, we finished the proof.

4 Experiment settings

We have experimented several numerical solver for stochastic differential equations, and finally decided to adopt the most straightforward Euler scheme. Although higher order solvers would also work, we find low order solver is fast and precise enough. We follow the idea in Neural ODE [1] and divide the whole classifier into three parts, the first part is to increase the number of channels to a suitable value (which can also be regarded as feature extraction for neural SDE); the following part the the ODE/SDE solver, note that the shape of intermediate states are not changed throughout. The last layer is for classification.

The overview of our model architecture is described in Figure 1. Here we list some key hyper-parameters for each model in Table 1. We can see that the architectures are roughly the same, except that for Tiny-ImageNet, our model is significantly larger due to that fact that this data is significantly harder to train on.

Dataset	First block	SDE block	Last block
MNIST	$\left[\text{Conv2d}(1, 64, 3, 1)\right] \times 1$	$\begin{bmatrix}\text{GroupNorm}(32,\ 64)\\ \text{Conv2d}(64, 64, 3, 1, 1)\\ \text{ReLU}\end{bmatrix} \times 3$	$\begin{bmatrix}\text{GroupNorm}(32, 64)\\ \text{ReLU}\\ \text{GAP}\\ \text{Linear}(64, 10)\end{bmatrix} \times 1$
CIFAR-10	$\begin{bmatrix}\text{Conv2d}(3, 64, 3, 1, 1)\\ \text{GroupNorm}(32,\ 64)\\ \text{ReLU}\\ \text{Conv2d}(64, 128, 4, 2, 1)\\ \text{GroupNorm}(32,\ 128)\\ \text{ReLU}\\ \text{Conv2d}(128, 256, 4, 2, 1)\end{bmatrix} \times 1$	$\begin{bmatrix}\text{GroupNorm}(32,\ 64)\\ \text{Conv2d}(64, 64, 3, 1, 1)\\ \text{ReLU}\end{bmatrix} \times 3$	$\begin{bmatrix}\text{GroupNorm}(32, 64)\\ \text{ReLU}\\ \text{GAP}\\ \text{Linear}(64, 10)\end{bmatrix} \times 1$
Tiny-ImageNet	$\begin{bmatrix}\text{Conv2d}(3, 64, 3, 1, 1)\\ \text{GroupNorm}(32,\ 64)\\ \text{ReLU}\\ \text{Conv2d}(64, 128, 4, 2, 1)\\ \text{GroupNorm}(32,\ 128)\\ \text{ReLU}\\ \text{Conv2d}(128, 256, 4, 2, 1)\end{bmatrix} \times 1$	$\begin{bmatrix}\text{GroupNorm}(32,\ 256)\\ \text{Conv2d}(256, 256, 3, 1, 1)\\ \text{ReLU}\end{bmatrix} \times 3$	$\begin{bmatrix}\text{GroupNorm}(32, 256)\\ \text{ReLU}\\ \text{GAP}\\ \text{Linear}(256, 200)\end{bmatrix} \times 1$

Table 1: Model hyper-parameters. We follow the parameter convention in PyTorch [7]. "GAP" means global average pooling [10].

5 Some empirical analysis

We provide some extra experiments to examine the discretization error due to Euler scheme. Different from traditional weak and strong convergence analysis of SDE solver, here we only need to care about the error in mean values, i.e. $\|\mathbb{E}\boldsymbol{X}_t - \mathbb{E}\bar{\boldsymbol{X}}_t\|$, since in our case only the results will be first be averaged before linear classifier and the accuracy of classification should not be affected as long as the mean values are precise enough. To verify that, we run our neural SDE model under different discretization step, specifically $\Delta t = \{1.0\times10^{-1}, 5.0\times10^{-2}, 1.0\times10^{-2}, 5.0\times10^{-3}, 1.0\times10^{-3}, 5.0\times10^{-4}, 1.0\times10^{-4}\}$ and because we cannot solve the equation in closed form, we choose the result by $\Delta t = 1.0\times10^{-5}$ as the ground truth. For each step size, we solve the SDE 1000 times independently and average the resulting image embedding vectors. The discritization error is measured by the relative error in the sense of Euclidean norm: $\|\boldsymbol{a}-\boldsymbol{b}\|_2/\|\boldsymbol{b}\|_2$. The results are shown in Figure 3. We can observe that although finer step size leads to smaller discretization error, even a coarse step $\Delta t = 0.1$ with relative error $\sim 2^{-4}$ can hardly change the prediction results.

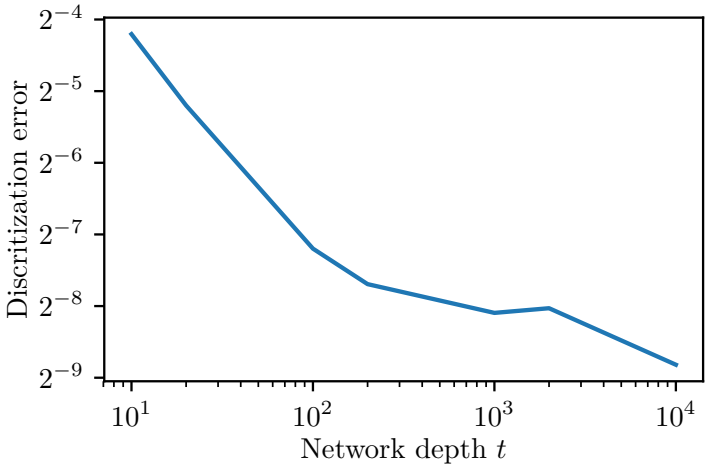

Figure 3: Discretization error under different step size in SDE solver.

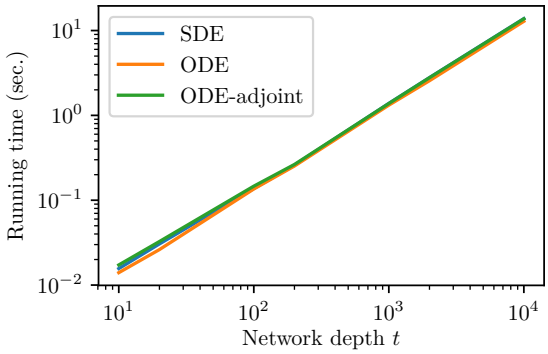

Figure 4: Running time comparison (forward propagation) between Neural SDE, Neural ODE and Neural ODE - adjoint. The curves are largely overlapped, meaning all methods have running time proportional to network depth.

6 Running time comparison: Neural SDE, Neural ODE, and Neural ODE-adjoint

See Figure 4.

References

[1] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pages 6572–6583, 2018. 5

[2] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 2

[3] Will Grathwohl, Ricky TQ Chen, Jesse Betterncourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018. 2

[4] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016. 2

[5] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018. 2

[6] Xuerong Mao. *Stochastic differential equations and applications*. Elsevier, 2007. 2, 3

[7] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[8] Scott Reed, Aäron van den Oord, Nal Kalchbrenner, Sergio Gómez Colmenarejo, Ziyu Wang, Yutian Chen, Dan Belov, and Nando de Freitas. Parallel multiscale autoregressive density estimation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2912–2921. JMLR. org, 2017. 2

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 1

[10] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 5