Continuous Normalizing Flows

Abstract

Continuous Normalizing Flows (CNFs) represent an emerging class of models that transform a simple distribution into a complex one through the integration of ordinary differential equations (ODEs). These flows provide a flexible framework for probabilistic modeling and density estimation in high-dimensional data. In this paper, we explore the theory behind continuous normalizing flows, compare them to traditional normalizing flows.

1 Background

1.1 Normalizing Flows

Let $\phi : \mathbb{R}^d \to \mathbb{R}^d$ be a continuously differentiable function which transforms elements of $\mathbb{R}^d$, with a continously differentiable inverse $\phi^{-1} : \mathbb{R}^d \to \mathbb{R}^d$. Let $q_0(x)$ be a density on $\mathbb{R}^d$ and let $p_1(\cdot)$ be the density induced by the following sampling procedure

$$\begin{aligned} &x \sim q_0 \\ &y = \phi(x), \end{aligned}$$

which corresponds to transforming the samples of q_0 by the mapping ϕ . Using the change-of-variable rule we can compute the density of p_1 as

$$p_1(y) = q_0(\phi^{-1}(y))\det\left[\frac{\partial \phi^{-1}}{\partial y}(y)\right] \tag{1}$$

$$= \frac{q_0(x)}{\det\left[\frac{\partial \phi}{\partial x}(x)\right]} \quad \text{with } x = \phi^{-1}(y) \tag{2}$$

where the last equality can be seen from the fact that $\phi \circ \phi^{-1} = \mathrm{Id}$ and a simple application of the chain rule. The quantity $\frac{\partial \phi^{-1}}{\partial y}$ is the Jacobian of the inverse map. It is a matrix of size $d \times d$ containing $J_{ij} = \frac{d\phi_i^{-1}}{dx_j}$.

1.2 Learning Flow Parameters by Maximum Likelihood

Let's denote the induced parametric density by the flow ϕ_θ as $p_1 \triangleq [\phi_\theta]_{\#}p_0$. A natural optimisation objective for learning the parameters $\theta \in \Theta$ is to consider maximising the probability of the data under the model:

$$\text{argmax}_\theta \ \ \mathbb{E}_{x\sim\mathcal{D}}[\log p_1(x)].$$

Designing flows ϕ therefore requires trading-off expressivity (of the flow and thus of the probabilistic model) with the above mentioned considerations so that the flow can be trained efficiently.

1.3 Full-rank Residual

Expressive flows relying on a residual connection have been proposed as an interesting middle-ground between expressivity and efficient determinant estimation. They take the form:

$$\phi_k(x) = x + \delta \ u_k(x), \tag{3}$$

where unbiased estimate of the log likelihood can be obtained.

One can also compose such flows to get a new flow:

$$\phi = \phi_K \circ \ldots \circ \phi_2 \circ \phi_1.$$

This can be a useful way to construct move expressive flows. The model's log-likelihood is then given by summing each flow's contribution

$$\log q(y) = \log p(\phi^{-1}(y)) + \sum_{k=1}^{K} \log\det\left[\frac{\partial \phi_k^{-1}}{\partial x_{k+1}}(x_{k+1})\right]$$

with $x_k = \phi_K^{-1} \circ \ldots \circ \phi_k^{-1}(y)$.

2 Continuous Time Limit

As mentioned previously, residual flows are transformations of the form $\phi(x) = x+\delta \ u(x)$ for some $\delta gt; 0$ and Lipschitz residual connection u. We can re-arrange this to get

$$\frac{\phi(x) - x}{\delta} = u(x)$$

which is looking awfully similar to u being a derivative. In fact, letting $\delta = 1/K$ and taking the limit $K \to \infty$ under certain conditions, a composition of residual flows $\phi_K \circ \cdots \circ \phi_2 \circ \phi_1$ is given by an ordinary differential equation (ODE):

$$\frac{x_t}{t} = \lim_{\delta\to 0} \frac{x_{t+\delta} - x_t}{\delta} = \frac{\phi_t(x_t) - x_t}{\delta} = u_t(x_t)$$

where the flow of the ODE $\phi_t : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ is defined such that

$$\frac{d\phi_t}{dt} = u_t(\phi_t(x_0)).$$

That is, ϕ_t maps initial condition x_0 to the ODE solution at time t:

$$x_t \triangleq \phi_t(x_0) = x_0 + \int_0^t u_s(x_s)s.$$

3 Continuous change-in-variables

Of course, this only defines the map $\phi_t(x)$; for this to be a useful normalising flow, we still need to compute the log-abs-determinant of the Jacobian. As it turns out, the density induced by ϕ_t (or equivalently u_t) can be computed via the following equation

$$\frac{\partial}{\partial_t} p_t(x_t) = -(\nabla \cdot (u_t p_t))(x_t).$$

This statement on the time-evolution of p_t is generally known as the Transport Equation. We refer to p_t as the probability path induced by u_t. Computing the total derivative (as x_t also depends on t) in log-space yields

$$\frac{d}{dt} \log p_t(x_t) = -(\nabla \cdot u_t)(x_t)$$

resulting in the log density

$$\log p_t(x) = \log p_0(x_0) - \int_0^t (\nabla \cdot u_s)(x_s)s.$$

Parameterising a vector field neural network $u_\theta : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}$ therefore induces a parametric log-density

$$\log p_\theta(x) \triangleq \log p_1(x) = \log p_0(x_0) - \int_0^1 (\nabla \cdot u_\theta)(x_t)dt.$$

In practice, to compute $\log p_t$ one can either solve both the time evolution of x_t and its log density $\log p_t$ jointly

$$\frac{d}{dt} \begin{pmatrix} x_t \\ \log p_t(x_t) \end{pmatrix} = \begin{pmatrix} u_\theta(t, x_t) \\ -\text{div } u_\theta(t, x_t) \end{pmatrix},$$

or solve only for x_t and then use quadrature methods to estimate $\log p_t(x_t)$.

4 Training CNFs

Similarly to any flows, CNFs can be trained by maximum log-likelihood

$$\mathcal{L}(\theta) = \mathbb{E}_{x \sim q_1}[\log p_1(x)],$$

where the expectation is taken over the data distribution and p_1 is the parameteric distribution. This involves integrating the time-evolution of samples x_t and log-likelihood $\log p_t$, both terms being a function of the parametric vector field $u_\theta(t, x)$.

References

[1] Tor Fjelde, Emile Mathieu, and Vincent Dutordoir. *An Introduction to Flow Matching*. 2024. URL: `https://mlg.eng.cam.ac.uk/blog/2024/01/20/flow-matching.html`.

[2] Yaron Lipman et al. *Flow Matching for Generative Modeling*. 2023. arXiv: `2210.02747 [cs.LG]`. URL: `https://arxiv.org/abs/2210.02747`.