

High-Dimensional Probability and the Neural Tangent Kernel

Your Name*

January 22, 2025

Abstract

We analyze the convergence and generalization of a coupling based normalizing flow model.

1 Preliminaries

1.1 Hypothesis set

Define $\mathcal{F} = \{f : \mathbb{R}^d \rightarrow \mathbb{R}^d\}$ as the set of all *Non-Volume Preserving Coupling-Based Normalizing Flows*. And define the *Realization Function* $F : \mathbb{R}^P \rightarrow \mathcal{F}$. Now we have for an $f_\theta := F(\theta)$ the following structure:

Denote the permutation by a matrix $Q \in \mathbb{R}^{d \times d}$ (an orthonormal permutation matrix), so that

$$\tilde{x} = Qx.$$

We then split \tilde{x} into two parts,

$$\tilde{x} = (\tilde{x}_1, \tilde{x}_2) \quad \text{with} \quad \tilde{x}_1 \in \mathbb{R}^{d'}, \tilde{x}_2 \in \mathbb{R}^{d-d'}.$$

An *affine coupling* transform $(\tilde{x}_1, \tilde{x}_2) \mapsto (\tilde{y}_1, \tilde{y}_2)$ is defined by

$$\tilde{y}_1 = \tilde{x}_1, \quad \tilde{y}_2 = \tilde{x}_2 \odot \exp[s_{\theta_1}(\tilde{x}_1)] + t_{\theta_2}(\tilde{x}_1),$$

where

$$s_{\theta_1} : \mathbb{R}^{d'} \rightarrow \mathbb{R}^{d-d'}, \quad t_{\theta_2} : \mathbb{R}^{d'} \rightarrow \mathbb{R}^{d-d'}.$$

are fully connected neural networks parameterized each by a subspace of \mathbb{R}^P . Thus, the overall transformation is

$$T_{\theta_1, \theta_2}(x) = \left(\tilde{x}_1, \tilde{x}_2 \odot \exp[s_{\theta_1}(\tilde{x}_1)] + t_{\theta_2}(\tilde{x}_1) \right)^\top \quad \text{where} \quad \tilde{x} = Qx.$$

The Tangent Kernel of f_θ

A *coupling based normalizing flow model* of depth L is therefore comprised by:

$$f_\theta(x) = T^{(L)} \circ T^{(L-1)} \circ \dots \circ T^{(1)}(x)$$

, and we denote $\theta = (\theta_1^s, \theta_1^t, \theta_2^s, \theta_2^t, \dots, \theta_L^s, \theta_L^t) \in \mathbb{R}^P$

*Your Affiliation, email@example.com

Jacobian Determinant. Because Q is a permutation matrix with $|\det Q| = 1$, its contribution to the Jacobian determinant is 1. The remaining (affine coupling) block is

$$\frac{\partial(\tilde{y}_1, \tilde{y}_2)}{\partial(\tilde{x}_1, \tilde{x}_2)} = \begin{pmatrix} I_d & 0 \\ * & \text{diag}(\exp[s_\theta(\tilde{x}_1)]) \end{pmatrix},$$

so the absolute Jacobian determinant is

$$|\det \nabla_x f_\theta(x)| = \prod_{j=1}^{d-d'} \exp[s_{\theta,j}(\tilde{x}_1)] = \exp\left(\sum_{j=1}^{d-d'} s_{\theta,j}(\tilde{x}_1)\right).$$

1.2 Loss Criterion

For a CPNF $z = f_\theta(x)$, with the base density $p_Z(z)$. By the change-of-variables formula, the model density is given by

$$\hat{p}_X(\mathbf{x}) = p_Z(f_\theta(\mathbf{x})) |\det J(f_\theta, \mathbf{x})|.$$

and minimizing the KL-divergence between $\mu = p_X$ and \hat{p}_X , will be equivalent to minimizing the expectation of the negative log-likelihood of \hat{p}_X with respect to p_X and is approximated by the empirical risk as below:

$$\mathcal{L}(\mathbf{x}; f) = -\log \hat{p}_X(\mathbf{x}) = -\log p_Z(f(\mathbf{x})) - \log |\det J(f, \mathbf{x})|$$

Negative Log-Likelihood. We choose a standard normal base distribution $p_z(z) = \mathcal{N}(z \mid 0, I)$ on \mathbb{R}^D . Then for $y = f_\theta(x)$, the model's density is

$$p_\theta(x) = p_Z(y) |\det \nabla_x f_\theta(x)|.$$

Hence the negative log-likelihood is

$$-\log p_\theta(x) = -\log p_Z(y) - \log |\det \nabla_x f_\theta(x)|.$$

If p_z is standard Gaussian,

$$-\log p_Z(y) = \frac{1}{2} \|y\|^2 + \frac{D}{2} \log(2\pi) \quad (\text{ignoring constants in } \theta).$$

Using the above determinant,

$$-\log |\det \nabla_x f_\theta(x)| = -\sum_{j=1}^{D-d} s_{\theta,j}(\tilde{x}_1).$$

Thus the single-sample loss becomes

$$\ell(\theta; x) = -\log p_\theta(x) = \frac{1}{2} \|f_\theta(x)\|^2 - \sum_{j=1}^{D-d} s_{\theta,j}(\tilde{x}_1) + \text{const.},$$

and the *training objective* for a dataset $\{x_i\}_{i=1}^N$ is

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N \left[-\log p_Z(f_\theta(x_i)) - \log |\det \nabla f_\theta(x_i)| \right] = \frac{1}{N} \sum_{i=1}^N \ell(\theta; x_i).$$

Remark 1.1. We might be able to show that NLL is equivalent to some supervised loss for some unknown optimal f^* , that $Z = f^*(X)$

1.3 Probability Space

Denote the probability space $(\Omega, \Sigma, \mathbb{P})$, where $X : \Omega \rightarrow \mathbb{R}^d$ has the unknown distribution $\mu \in \mathcal{M}(\mathbb{R}^d)$ and $Z \sim N(0, I_d)$. Also define the empirical distribution μ_N for our dataset $S = \{X^{(i)}\}_{i=1}^N$, as $\mu_N(A) = \frac{1}{N} \sum_{i=1}^N \mathbf{1}_{X_i \in A}$, where X_i are independent copies of X .

The ultimate learning objective is to minimize the *statistical risk*:

$$R(f) := \mathbb{E}_\mu[\ell(\theta; X)]$$

, and the *empirical risk* is:

$$\hat{R}_S(f) := \frac{1}{|S|} \sum_{x \in S} \ell(\theta; x)$$

2 The Neural Tangent Kernel

H: Maybe we should see the whole p_θ as the network ?

First we compute gradient of a transform $T^{(i)} = T_{\theta_1, \theta_2}$ as:

$$\nabla_{(\theta_1, \theta_2)} T_{\theta_1, \theta_2}(x) = \begin{bmatrix} \mathbf{0}_{\dim(\tilde{x}_1) \times \dim(\theta_1)} & \mathbf{0}_{\dim(\tilde{x}_1) \times \dim(\theta_2)} \\ \tilde{x}_2 \odot \exp[s_{\theta_1}(\tilde{x}_1)] \odot \nabla_{\theta_1} s_{\theta_1}(\tilde{x}_1) & \nabla_{\theta_2} t_{\theta_2}(\tilde{x}_1) \end{bmatrix} := \partial T$$

and so we have $\nabla_\theta T_{\theta_1, \theta_2}(x) = (0, \dots, \partial T, \dots, 0) \in \mathbb{R}^{P \times d}$.

We can define the *Tangent Kernel* of a CBNF f_θ as $H(t)$, which is an $N \times N$ positive semi-definite matrix whose (i, j) -th entry is $\langle \frac{\partial f(\theta(t), x_i)}{\partial \theta}, \frac{\partial f(\theta(t), x_j)}{\partial \theta} \rangle$.

For our **Training Cost Functional** $\mathcal{L}(\theta)$, we have:

$$\nabla_\theta \mathcal{L}(\theta) = \nabla_\theta f$$

$$C^{\mu_N}(f) = \left\| -\log p_Z(f) - \log \left| \det J(f) \right| \right\|_{\mu_N} = \frac{1}{N} \sum_{i=1}^N -\log p_Z(f(X_i)) - \log \left| \det J(f, X_i) \right|$$

, which is \hat{R}_S , for our S . Now we take the (functional) derivative of this with respect to a function f_θ , denoted by:

$$\partial_f^{\mu_N} C|_{f_\theta} = \|f - [\text{Tr} \left(J(f)^{-1} \frac{\partial J(f)}{\partial f_1} \right) \cdots \text{Tr} \left(J(f)^{-1} \frac{\partial J(f)}{\partial f_d} \right)]^\top\|_{\mu_n}$$

This is a point in \mathcal{F}^* and so is a function of $f \in \mathcal{F}$ to \mathbb{R} , and we can denote it by $\phi_\theta : \mathcal{F} \rightarrow \mathbb{R}$.

2.1 Deriving the Kernel

So denoting $f_\theta := F(\theta)$ the parameters change as below in gradient descent:

$$\frac{d\theta_p}{dt} = -\eta \partial_{\theta_p} C \circ F(\theta) = -\eta \partial_f^{\mu_N} C|_{f_\theta}(\partial_{\theta_p} F(\theta)) = -\eta \phi_\theta(\partial_{\theta_p} F(\theta))$$

, giving us the gradient form:

$$\nabla_\theta C \circ F(\theta) = [\phi_\theta(\partial_{\theta_1} F(\theta)) \cdots \phi_\theta(\partial_{\theta_P} F(\theta))]^\top, \quad \frac{d\theta}{dt} = -\eta \nabla_\theta C \circ F(\theta)$$

Defining the Neural Tangent Kernel $\mathbf{H}(\theta)$ We want to solve the differential equation below:

$$\partial_t f_{\theta(t)} = -\frac{1}{N} \sum_{i=1}^N$$

From the blog:

$$\frac{df(\mathbf{x}'; \theta)}{dt} = \frac{df(\mathbf{x}'; \theta)}{d\theta} \frac{d\theta}{dt} = -\frac{1}{N} \sum_{i=1}^N \underbrace{\nabla_{\theta} f(\mathbf{x}; \theta)^{\top} \nabla_{\theta} f(\mathbf{x}^{(i)}; \theta)}_{\text{Neural Tangent Kernel}} \nabla_f \ell(f, y^{(i)})$$

$$\frac{df(\mathbf{x}'; \theta)}{dt} = \frac{df(\mathbf{x}'; \theta)}{d\theta} \frac{d\theta}{dt} = -\frac{\eta}{N} \sum_{i=1}^N \nabla_{\theta} f(\mathbf{x}'; \theta)^{\top} \nabla_{\theta} f(\mathbf{x}^{(i)}; \theta) \nabla_f \ell(f, y^{(i)})$$

3 Random Process View

It is well-known that Neural Networks in the infinite-width output a Gaussian process at their Gaussian initialization.

But what happens in training to a Neural Network. **In training** the cost function follows a Gaussian process, indexed by the space \mathbb{R}^P of parameters. we can create some empirical processes according to the empirical risk and statistical risk indexed by the hypothesis \mathcal{F} or \mathbb{R}^P .

We have three probability measures so far: $N(0, I), \mu, \mu_N$.

Definition 3.1. Learning Processes

Take the functions $C : \mathcal{F} \rightarrow \mathbb{R}$ and $R := C \circ F : \mathbb{R}^P \rightarrow \mathbb{R}$, we can use the laws μ and μ_N such that:

- **Function Processes:** The random variables $f(X)$ or $F(\theta)(X)$, respectively indexed on \mathcal{F} and \mathbb{R}^P , with $X \sim \mu$ or μ_N .
- **Empirical Processes:** The functions $\mathbb{G}_N(f) := \sqrt{N}(\mu_N - \mu)f$ and $\mathbb{G}_N^{\theta}(\theta) := \sqrt{N}(\mu_N - \mu)F(\theta)$, respectively indexed on \mathcal{F} and \mathbb{R}^P , with $X \sim \mu$ or μ_N .
- **Training Processes:** The functions $C(f)$ and $C \circ F(f)$, when $S = \{X_i\}_{i=1}^N \sim \mu^N$ or maybe even $S \sim \mu_N^N$ can be useful.

We shall analyze properties of this random process. For example the ultimate goad of machine learning would be characterized as $\mathbb{E}[\sup_{\theta} R(\theta)]$, which is the statistical risk for the objective learning concept. Our empirical risk is also characterized by $\mathbb{E} \sup_{\theta} \hat{R}(\theta)$

What is the covariance function of these processes ? is it NTK ? Can we apply the results of the books on Random Processes to these ?

4 Sub-Gaussian Investigation

References

A Additional Proofs

Additional technical proofs, auxiliary lemmas, or numerical experiments may be included here.