Journal of Machine Learning Research 27 (2026) 1-52 Submitted 7/24; Revised 11/25; Published 1/26

Finite Neural Networks as Mixtures of Gaussian Processes: From Provable Error Bounds to Prior Selection

Steven Adams S.J.L.ADAMS@TUDELFT.NL
Delft Center for Systems and Control (DCSC), Delft University of Technology, The Netherlands

Andrea Patanè APATANE@TCD.IE
School of Computer Science and Statistics, Trinity College Dublin, Ireland

Morteza Lahijanian MORTEZA.LAHIJANIAN@COLORADO.EDU
Departments of Aerospace Engineering Sciences and Computer Science, University of Colorado Boulder, USA

Luca Laurenti L.LAURENTI@TUDELFT.NL
Delft Center for Systems and Control (DCSC), Delft University of Technology, The Netherlands
The Italian Institute of Artificial Intelligence (AI4I), Italy

Editor: Christophe Giraud

Abstract

Infinitely wide or deep neural networks (NNs) with independent and identically distributed (i.i.d.) parameters have been shown to be equivalent to Gaussian processes. Because of the favorable properties of Gaussian processes, this equivalence is commonly employed to analyze neural networks and has led to various breakthroughs over the years. However, neural networks and Gaussian processes are equivalent only in the limit; in the finite case there are currently no methods available to approximate a trained neural network with a Gaussian model with bounds on the approximation error. In this work, we present an algorithmic framework to approximate a neural network of finite width and depth, and with not necessarily i.i.d. parameters, with a mixture of Gaussian processes with bounds on the approximation error. In particular, we consider the Wasserstein distance to quantify the closeness between probabilistic models and, by relying on tools from optimal transport and Gaussian processes, we iteratively approximate the output distribution of each layer of the neural network as a mixture of Gaussian processes. Crucially, for any NN and $\epsilon > 0$ our approach is able to return a mixture of Gaussian processes that is ϵ-close to the NN at a finite set of input points. Furthermore, we rely on the differentiability of the resulting error bound to show how our approach can be employed to tune the parameters of a NN to mimic the functional behavior of a given Gaussian process, e.g., for prior selection in the context of Bayesian inference. We empirically investigate the effectiveness of our results on both regression and classification problems with various neural network architectures. Our experiments highlight how our results can represent an important step towards understanding neural network predictions and formally quantifying their uncertainty.

Keywords: Neural Networks, Gaussian Processes, Bayesian inference, Wasserstein distance

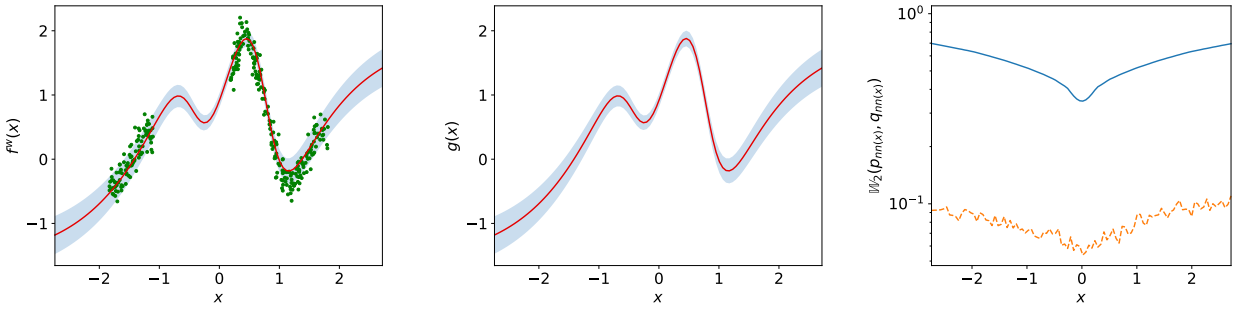
1. Introduction

Deep neural networks have achieved state-of-the-art performance in a wide variety of tasks, ranging from image classification (Krizhevsky et al., 2012) to robotics and reinforcement learning (Mnih et al., 2013). In parallel with these empirical successes, there has been a significant effort in trying to understand the theoretical properties of neural networks (Goodfellow et al., 2016) and to guarantee their robustness (Szegedy et al., 2013). In this context, an important area of research is that of stochastic neural networks (SNNs), where some of the parameters of the neural network (weights and biases) are not fixed, but follow a distribution. SNNs include many machine learning models commonly used in practice, such as dropout neural networks (Gal and Ghahramani, 2016), Bayesian neural networks (Neal, 2012), neural networks with only a subset of stochastic layers (Favaro et al., 2025), and neural networks with randomized smoothing (Cohen et al., 2019). Among these, because of their convergence to Gaussian processes, particular theoretical attention has been given to infinite neural networks with independent and identically distributed (i.i.d.) parameters (Neal, 2012; Lee et al., 2018).

Gaussian processes (GPs) are a class of stochastic processes that are widely used as non-parametric machine learning models (Rasmussen, 2003). Because of their many favorable analytic properties (Adler and Taylor, 2009), the convergence of infinite SNNs to GPs has enabled many breakthroughs in the understanding of neural networks, including their modeling capabilities (Schoenholz et al., 2016), their learning dynamics (Jacot et al., 2018), and their adversarial robustness (Bortolussi et al., 2024). Unfortunately, existing results to approximate a SNN with a GP are either limited to untrained networks with i.i.d. parameters (Neal, 2012) or lack guarantees of correctness (Khan et al., 2019). In fact, the input-output distribution of a SNN of finite depth and width is generally non-Gaussian, even if the distribution over its parameters is Gaussian (Lee et al., 2020). This leads to the main question of this work: *Can we develop an algorithmic framework to approximate a finite SNN (trained or untrained and not necessarily with i.i.d. parameters) with Gaussian models while providing formal guarantees of correctness (i.e., provable bounds on the approximation error and that can be made arbitrarily small)?*

In this paper, we propose an algorithmic framework to approximate the input-output distribution of an arbitrary SNN over a finite set of input points with a Gaussian Mixture Model (GMM), that is, a mixture of M Gaussian distributions (McLachlan and Peel, 2000). Critically, the GMM approximation resulting from our approach comes with error bounds on its distance (in terms of the 2-Wasserstein distance,[1] Villani et al., 2009) to the input-output distribution of the SNN. An illustrative example of our framework is shown in Figure 1a, where, given a SNN trained on a 1D regression task (Figure 1a), our framework outputs a GMM approximation (Figure 1b) with error bounds on its closeness to the SNN (Figure 1c). Our approach is based on iteratively approximating the output distribution of each layer of the SNN with a mixture of Gaussian distributions and propagating this distribution through the next layer. In order to propagate a distribution through a non-linear activation function, we first approximate it with a discrete distribution, which we call a *signature approximation*.[2] The resulting discrete distribution can then be propagated exactly through a layer of the neural network (activation function and affine combination with weights and biases), which in the case of jointly Gaussian weights and biases, leads to a new Gaussian mixture distribution. To quantify the error between the SNN and the resulting GMM, we use techniques

1. Note that, as we will emphasize in Section 3, the choice of the 2-Wasserstein distance to quantify the distance between a SNN and a GMM guarantees that a bound on the 2-Wasserstein distance also implies a bounds in how distant are their mean and variance.
2. A discrete approximation of a continuous distribution is also called a codebook in the field of constructive quantization (Graf and Luschgy, 2000) or particle approximation in Bayesian statistics (Liu and Wang, 2016).



(a) SNN (b) GMM (c) Error

Figure 1: Visualization of (a) a Monte Carlo approximation of a SNN trained on samples from a 1D mixture of sines with additive noise, and (b) the GMM approximation of the same SNN composed of a mixture of size 10, as obtained with our approach. The SNN has one hidden layer with 64 neurons and is trained using VOGN (Khan et al., 2018). The red line and blue-shaded region in (a) and (b) illustrate the point-wise mean and standard deviation at a uniform grid of 100 points in the input domain. In (c), the dashed and solid lines represent, respectively, the empirical estimates of the 2-Wasserstein distance between the GMM and SNN distributions at each input point, and formal bounds of the same quantity, as obtained by the results derived in this paper.

from optimal transport, probability theory, and interval arithmetic. In particular, for each of the approximation steps described above, we bound the error introduced in terms of the 2-Wasserstein distance. Then, we combine these bounds using interval arithmetic to bound the 2-Wasserstein distance between the SNN and the approximating GMM. Furthermore, by relying on the fact that GMMs can approximate any distribution arbitrarily well (Delon and Desolneux, 2020), we prove that by appropriately picking the parameters of the GMM, the bound on the approximation error converges uniformly to zero by increasing the size of the GMM. Additionally, we show that the resulting error bound is piecewise differentiable in the hyper-parameters of the neural network, allowing gradient-based optimization of the behavior of a NN to mimic that of a given GMM.

We empirically validate our framework on various SNN architectures, including fully-connected and convolutional layers, trained for both regression and classification tasks on various data sets including MNIST, CIFAR-10, and a selection of UCI data sets. Our experiments confirm that our approach can successfully approximate a SNN with a GMM with arbitrarily small error, albeit with increasing computational costs when the network's depth increases. Furthermore, perhaps surprisingly, our results show that even a mixture with a relatively small number of components generally suffices to empirically approximate a SNN accurately. To showcase the importance of our results, we then consider two applications: (i) uncertainty quantification in SNNs, and (ii) prior selection for SNNs. In the former case we show how the GMM approximation resulting from our framework can be used to study and quantify the uncertainty of the SNN predictions in classification tasks on the MNIST and CIFAR-10 data sets. In the latter case, we consider prior selection for neural networks, which is arguably one of the most important problems in performing Bayesian inference with neural networks (Fortuin, 2022), and show that our framework allows one to precisely encode functional information in the prior of SNNs expressed as Gaussian processes (GPs), thereby enhancing SNNs' posterior performance and outperforming state-of-the-art methods for prior selection of neural networks.

In summary, the main contributions of our paper are:

- We introduce a framework based on discrete approximations of continuous distributions to approximate SNNs of arbitrary depth and width as GMMs, with formal error bounds in terms of the 2-Wasserstein distance.
- We prove the uniform convergence of our framework by showing that for any finite set of input points our approach can return a GMM of finite size such that the 2-Wasserstein distance between this distribution and the joint input-output distribution of the SNN on these points can be made arbitrarily small.
- We perform a large-scale empirical evaluation to demonstrate the efficacy of our algorithm in approximating SNNs with GMMs and to show how our results could have a profound impact in various areas of research for neural networks, including uncertainty quantification and prior selection.

1.1 Related Works

The convergence of infinitely wide SNNs with i.i.d. parameters to GPs was first studied by Neal (2012) by relying on the central limit theorem. The corresponding GP kernel for the one hidden layer case was then analytically derived by Williams (1996). These results were later generalized to deep NNs (Hazan and Jaakkola, 2015; Lee et al., 2018; Matthews et al., 2018), convolutional layers (Garriga-Alonso et al., 2019; Novak et al., 2018; Garriga-Alonso and van der Wilk, 2021), and general non-linear activation functions (Hanin, 2023). However, these results only hold for infinitely wide or deep neural networks with i.i.d. parameters; in practice, NNs have finite size and depth and for trained NNs their parameters are generally not i.i.d.. To partially address these issues, recent works have started to investigate how these results apply in the finite case (Dyer and Gur-Ari, 2020; Antognini, 2019; Yaida, 2020; Bracale et al., 2021; Klukowski, 2022; Balasubramanian et al., 2024). In particular, Eldan et al. (2021) were the first to provide upper bounds on the rates at which a single layer SNN with a specific parameter distribution converges to the infinite width GP in terms of the Wasserstein distance. This result was later generalized to isotropic Gaussian parameter distributions (Cammarota et al., 2023), deep Random NN (Basteri and Trevisan, 2024), and to other metrics, such as the total variation, sup norm, and Kolmogorov distances (Apollonio et al., 2025; Bordino et al., 2023; Favaro et al., 2025). However, to the best of our knowledge, all existing works are limited to untrained SNNs with i.i.d. weights and biases, which may be taken as priors in a Bayesian setting (Bishop and Nasrabadi, 2006) or may represent the initialization of gradient flows in an empirical risk minimization framework (Jacot et al., 2018). In contrast, critically, in our paper we allow for correlated and non-identical distributed weights and biases, thus also including trained posterior distributions of SNNs learned via Variational Inference. In this context, Khan et al. (2019) showed that for a subset of Gaussian Bayesian inference techniques, the approximate posterior parameter distributions are equivalent to the posterior distributions of GP regression models, implying a linearization of the approximate SNN posterior in function space (Immer et al., 2021). Instead, we approximate the SNN with guarantees in function space and our approach generalizes to any approximate inference technique.

A fundamentally different approach to approximating general distributions is to use empirical measures obtained from independent samples of the target distribution. Such methods provide only statistical rather than deterministic guarantees: their accuracy holds with high probability and relies on assumptions such as bounded support or low dimensionality (Fournier and Guillin, 2015), which typically do not hold for SNN output distributions.

While the former set of works focuses on the convergence of SNNs to GPs, various works have considered the complementary problem of finding the distribution of the parameters of a SNN that mimic a given GP (Flam-Shepherd et al., 2017, 2018; Tran et al., 2022; Matsubara et al., 2021). This is motivated by the fact that GPs offer an excellent framework to encode functional prior knowledge; consequently, this problem has attracted significant interest in encoding functional prior knowledge into the prior of SNNs. In particular, closely related to our setting are the works of Flam-Shepherd et al. (2017) and Tran et al. (2022), which optimize the parametrization of the parameter-space distribution of the SNN to minimize the KL divergence and 1-Wasserstein distance, respectively, with a desired GP prior. They then utilize the optimized parameter prior to perform Bayesian inference, showing improved posterior performance. However, these methods lack formal guarantees on the closeness of the optimized SNN and the GP that the error bounds we derive in this paper provide.

2. Notation

For a vector $x \in \mathbb{R}^n$, we denote by $\|x\|$ the Euclidean norm on $\mathbb{R}^n$, and use $x^{(i)}$ to denote the i-th element of x. Similarly, for a matrix $M \in \mathbb{R}^{n\times m}$ we denote by $\|M\|$ the spectral (matrix) norm of M and use $M^{(i,j)}$ to denote the (i,j)-th element of M. In addition, $\mathrm{diag}(x)$ denotes a matrix with the elements of x on its diagonal. For a matrix $T \in \mathbb{R}^{m\times n}$, the post image of a region $\mathcal{X} \subset \mathbb{R}^n$ under T is defined as $\mathrm{Post}(\mathcal{X}, T) = \{Tx \mid x \in \mathcal{X}\}$. If $\mathcal{X}$ is finite, $|\mathcal{X}|$ denotes its cardinality.

Given a measurable space $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ with $\mathcal{B}(\mathcal{X})$ being the $\sigma-$algebra, we denote with $\mathcal{P}(\mathcal{X})$ the set of probability distributions on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$. In this paper, for a metric space $\mathcal{X}$, $\mathcal{B}(\mathcal{X})$ is assumed to be the Borel σ-algebra of $\mathcal{X}$. Considering two measurable spaces $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ and $(\mathcal{Y}, \mathcal{B}(\mathcal{Y}))$, a probability distribution $p \in \mathcal{P}(\mathcal{X})$, and a measurable mapping $h : \mathcal{X} \to \mathcal{Y}$, we use $h\#p$ to denote the pushforward measure of p by h, i.e., the measure on $\mathcal{Y}$ such that $\forall \mathcal{A} \in \mathcal{B}(\mathcal{Y})$, $(h\#p)(\mathcal{A}) = p(h^{-1}(\mathcal{A}))$. For $N \in \mathbb{N}$, $\Pi_N = \{\pi \in \mathbb{R}^N_{\geq 0} : \sum_{i=1}^N \pi^{(i)} = 1\}$ is the N-simplex. A discrete probability distribution $d \in \mathcal{P}(\mathcal{X})$ is defined as $d = \sum_{i=1}^N \pi^{(i)} \delta_{c_i}$, where δ_c is the Dirac delta function centered at location $c \in \mathcal{X}$ and $\pi \in \Pi_N$. Lastly, the set of discrete probability distributions on $\mathcal{X}$ with at most N locations is denoted as $\mathcal{D}_N(\mathcal{X}) \subset \mathcal{P}(\mathcal{X})$.

3. Preliminaries

In this section, we give the necessary preliminary information on Gaussian models and on the Wasserstein distance between probability distributions.

3.1 Gaussian Processes and Gaussian Mixture Models

A Gaussian process (GP) g is a stochastic process such that for any finite collection of input points $\mathcal{X} = \{x_1, \ldots, x_D\}$ the joint distribution of $g(\mathcal{X}) \coloneqq \{g(x_1), \ldots, g(x_D)\}$ follows a multivariate Guassian distribution with mean function m and covariance function k, i.e., $g(\mathcal{X}) \sim \mathcal{N}(m(\mathcal{X}), k(\mathcal{X}, \mathcal{X}))$ (Adler and Taylor, 2009). A Gaussian Mixture Model (GMM) with $M \in \mathbb{N}$ components, is a set of M GPs, also called components, averaged w.r.t. a probability vector $\pi \in \Pi_M$ (Tresp, 2000). Therefore, the probability distribution of a GMM follows a Gaussian mixture distribution.

Definition 1 (Gaussian Mixture Distribution) *A probability distribution $q \in \mathcal{P}(\mathbb{R}^d)$ is called a Gaussian Mixture distribution of size $M \in \mathbb{N}$ if $q = \sum_{i=1}^M \pi^{(i)} \mathcal{N}(m_i, \Sigma_i)$, where $\pi \in \Pi_M$ and $m_i \in \mathbb{R}^d$ and $\Sigma_i \in \mathbb{R}^{d\times d}$ are the mean and covariance matrix of the i-th Gaussian distribution in the mixture. The set of all Gaussian mixture distributions with M or less components is denoted by $\mathcal{G}_M(\mathbb{R}^d) \subset \mathcal{P}(\mathbb{R}^d)$.*

One of the key properties of GMMs, which motivates their use in this paper to approximate the probability distribution induced by a neural network, is that for large enough M they can approximate any continuous probability distribution arbitrarily well (Delon and Desolneux, 2020). Furthermore, being a Gaussian mixture distribution a weighted sum of Gaussian distributions, it inherits the favorable analytic properties of Gaussian distributions (Bishop and Nasrabadi, 2006).

3.2 Wasserstein Distance

To approximate SNNs to GMMs and vice-versa, and quantify the quality of the approximation, we need a notion of distance between probability distributions. While various distance measures are available in the literature, in this work we consider the Wasserstein distance (Gibbs and Su, 2002). To define the Wasserstein distance, for $\rho \geq 1$ we define the $\rho-$Wasserstein space of distributions $\mathcal{P}_\rho(\mathbb{R}^d)$ as the set of probability distributions with finite moments of order ρ, i.e., any $p \in \mathcal{P}_\rho(\mathbb{R}^d)$ is such that $\int_{\mathbb{R}^d} \|x\|^\rho dp(x) < \infty$. For $p, q \in \mathcal{P}_\rho(\mathbb{R}^d)$, the ρ-Wasserstein distance $\mathbb{W}_\rho$ between p and q is defined as

$$\mathbb{W}_\rho(p,q) \coloneqq \left(\inf_{\gamma \in \Gamma(p,q)} \mathbb{E}_{(x,z)\sim\gamma}[\|x-z\|^\rho] \right)^{\frac{1}{\rho}} = \left(\inf_{\gamma \in \Gamma(p,q)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x-z\|^\rho \gamma(dx,dz) \right)^{\frac{1}{\rho}}, \tag{1}$$

where $\Gamma(p,q) \subset \mathcal{P}_\rho(\mathbb{R}^d \times \mathbb{R}^d)$ represents the set of probability distributions with marginal distributions p and q. It can be shown that $\mathbb{W}_\rho$ is a metric, which is given by the minimum cost, according to the $\rho-$power of the Euclidean norm, required to transform one probability distribution into another (Villani et al., 2009). Furthermore, another attractive property of the ρ-Wasserstein distance, which distinguishes it from other divergence measures such as the KL divergence (Hershey and Olsen, 2007), is that closeness in the ρ-Wasserstein distance implies closeness in the first ρ moments.[3]

In what follows, we focus on the 2-Wasserstein distance.[4] While closed-form expressions exist for the 2-Wasserstein distance for Gaussian distributions (Givens and Shortt, 1984),[5] this is not the case for Gaussian mixture distributions. However, a tractable upper bound for the $\mathbb{W}_2$ is provided by the $\mathbb{MW}_2$ distance, formally defined below following (Delon and Desolneux, 2020).

Definition 2 ($\mathbb{MW}_2$ Distance) *Let $p = \sum_{i=1}^{N_p} \pi_p^{(i)} p_i \in \mathcal{G}_{N_1}(\mathbb{R}^n)$ and $q = \sum_{j=1}^{N_q} \pi_q^{(j)} q_j \in \mathcal{G}_{N_1}(\mathbb{R}^n)$ be two Gaussian Mixture distributions. Then, the $\mathbb{MW}_2$-distance is defined as*

$$\mathbb{MW}_2(p,q) \coloneqq \left(\inf_{\gamma \in \Gamma(p,q) \cap \mathcal{G}_{<\infty}(\mathbb{R}^{2n})} \mathbb{E}_{(x,z)\sim\gamma}[\|x-z\|^2] \right)^{\frac{1}{2}} = \left(\min_{\pi \in \Gamma(\pi_p, \pi_q)} \sum_{i,j} \pi^{(i,j)} \mathbb{W}_2^2(p_i, q_j) \right)^{\frac{1}{2}}, \tag{2}$$

where $\mathcal{G}_{<\infty}(\mathbb{R}^n \times \mathbb{R}^n)$ is the set of Gaussian mixture distributions with finitely many components.

The $\mathbb{MW}_2$ distance is a Wasserstein-type distance that restricts the set of possible couplings to Gaussian mixtures. This guarantees that $\mathbb{MW}_2$ computation reduces to a finite discrete linear program with $N_p \cdot N_q$ optimization variables and coefficients (Delon and Desolneux, 2020). The coefficients are given by the Wasserstein distance between the mixture's Gaussian components, which have closed-form expressions. Furthermore, as Definitions 1 and 2 only differ for the intersection on $\mathcal{G}_{<\infty}$ in the couplings, it is easy to see that $\mathbb{MW}_2(p,q) \geq \mathbb{W}_2(p,q)$.

3. See Lemma 28 in Appendix A.
4. Since $\mathbb{W}_1 \leq \mathbb{W}_2$, the methods presented in this work naturally extend to the 1-Wasserstein distance. Detailed comparisons and potential improvements when utilizing the 1-Wasserstein distance are reported in the Appendix.
5. For $p_1 = \mathcal{N}(m_1, \Sigma_1) \in \mathcal{G}(\mathbb{R}^n)$ and $p_2 = \mathcal{N}(m_2, \Sigma_2) \in \mathcal{G}(\mathbb{R}^n)$, it holds that $\mathbb{W}_2^2(p_1,p_2) = \|m_1 - m_2\|^2 + \operatorname{trace}\left(\Sigma_1 + \Sigma_2 - 2\left(\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2}\right)^{1/2}\right)$.

4. Problem Formulation

In this section, we first introduce the class of neural networks considered in this paper, and then we formally state our problem.

4.1 Stochastic Neural Networks (SNNs)

For an input vector $x \in \mathbb{R}^{n_0}$, we consider a feedforward neural network of K hidden layers $f^w(x)$ defined iteratively as:

$$z_1 = L^{w_0}(x) \tag{3a}$$
$$z_{k+1} = L^{w_k}(\sigma(z_k)), \qquad k \in \{1, \dots, K\} \tag{3b}$$
$$f^w(x) = z_{K+1} \tag{3c}$$

where, for n_k being the number of neurons of layer k, we have that $\sigma_k : \mathbb{R}^{n_k} \to \mathbb{R}^{n_k}$ is a the vector of piecewise continuous activation functions, and $L^{w_k} : \mathbb{R}^{n_k} \to \mathbb{R}^{n_{k+1}}$ denotes an affine transformation, such as a fully-connected (dense) layer or a convolutional layer. Each L^{w_k} is parametrized by weights W_k, e.g., a transformation matrix or convolutional kernel, and bias b_k, which are collectively stored in $w_k = \{W_k, b_k\}$. We denote by $w = \{w_k\}_{k=0}^K$ the union of all the neural network parameters. z_{K+1} is the final output of the network, possibly corresponding to the vector of logits in case of classification problems.

In this work, we assume that the parameters w_k of each layer k, rather than being fixed, are distributed according to a probability distribution p_{w_k}.[6] For any $x \in \mathbb{R}^{n_0}$, placing a distribution over the parameters leads to a distribution over the outputs of the NN. That is, $f^w(x)$ is a random variable and $\{f^w(x)\}_{x \in \mathbb{R}^{n_0}}$ is a stochastic process, which we call a *stochastic neural network* (SNN) to emphasize its randomness (Yu et al., 2021). In particular, $f^w(x)$ follows a probability distribution $p_{nn(x)}$, which, as shown in Adams et al. (2023), can be iteratively defined over the layers of the SNN. Specifically, as shown in Eqn (4) below, $p_{nn(x)}$ is obtained by recursively propagating through the layers of the NN architecture the distribution induced by the random parameters at each layer. The propagation is obtained by marginalization of the output distribution at each layer with the distribution of the previous layer:

$$p_{nn(x),1} = \mathbb{E}_{z_0 \sim \delta_x}[L^{w_0}(z_0) \# p_{w_0}] \tag{4a}$$
$$p_{nn(x),k+1} = \mathbb{E}_{z_k \sim p_{nn(x),k}}[L^{w_k}(\sigma(z_k)) \# p_{w_k}], \qquad k \in \{1, \dots, K\} \tag{4b}$$
$$p_{nn(x)} = p_{nn(x),K+1} \tag{4c}$$

where δ_x is the Delta Dirac function centered at x.[7] For any finite subset $\mathcal{X} \subset \mathbb{R}^{n_0}$ of input points, we use $p_{nn(\mathcal{X})}$ to denote the joint distribution of the output of f^w evaluated at the points in $\mathcal{X}$.

In what follows, because of its practical importance and the availability of closed-form solutions, we will introduce our methodological framework under the assumption that p_{w_k} is a multivariate Gaussian distributions for all k. We stress that this does not imply that the output distribution of the SNN, $p_{nn}(x)$, is also Gaussian; in fact, in general, it is not. Furthermore, we should also already remark that, as we will explain in Remark 16 and illustrate in the experimental results, the methods we present in this paper can be extended to more general (non-necessarily Gaussian) $p_{w_k} \in \mathcal{P}_2(\cdot)$.

6. This includes architectures with both deterministic and stochastic parameters, where the distribution over deterministic parameters can be modeled as a Dirac delta function.

7. Equivalently, for all k, $p_{nn(x),k+1}$ is a mixture distribution with infinitely many components of the form $L^{w_k}(\sigma(z_k)) \# p_{w_k}$ for $z_k \in \mathbb{R}^{n_k}$, weighted by $p_{nn(x),k}$, or, equivalently, the pushforward of the joint distribution of p_{w_k} and $p_{nn(x),k}$ by $L^{w_k}(\sigma(z_k))$.

Remark 3 *The above definition of SNNs encompasses several stochastic models of practical importance, including Bayesian Neural Networks (BNNs) trained with Gaussian Variational Inference methods such as, e.g., Bayes by Backprop (Blundell et al., 2015) or Noisy Adam (Zhang et al., 2018), NNs with only a subset of stochastic layers (Tang and Salakhutdinov, 2013), Gaussian Dropout NNs (Srivastava et al., 2014), and NNs with randomized smoothing and/or stochastic inputs (Cohen et al., 2019). The methods proposed in this paper apply to all such models that assume the weight distributions of different layers are independent. In particular, in the case of BNNs, in Section 8, we will show how our approach can be used to investigate both the prior and posterior behavior, in which case, depending on the context, p_w could represent either the prior or the approximate posterior distribution of the weights and biases.*

4.2 Problem Statement

Given an error threshold $\epsilon > 0$ and a SNN, the main problem we consider, as formalized in Problem 1 below, is that of finding a GMM that is $\epsilon-$close according to the 2-Wasserstein distance to the SNN.

Problem 1 *Let $\mathcal{X} \subset \mathbb{R}^{n_0}$ be a finite set of input points, and $\epsilon > 0$ be an error threshold. Then, for a SNN with distribution $p_{nn(\mathcal{X})}$ find a GMM with distribution $q_{nn(\mathcal{X})}$ such that:*

$$\mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right) \leq \epsilon. \tag{5}$$

In Problem 1 we aim to approximate a SNN with a GMM, thus extending existing results (Neal, 2012; Matthews et al., 2018) that approximate an untrained SNN with i.i.d. parameters with a GP under some limit (e.g., infinite width, Neal, 2012; Matthews et al., 2018, or infinitely many convolutional filters, Garriga-Alonso et al., 2019). In contrast, in Problem 1 we consider both trained and untrained SNNs of finite width and depth and, crucially, we aim to provide formal error bounds on the approximation error. Note that in Problem 1 we consider a set of input points and compute the $2-$Wasserstein distance of the joint distribution of a SNN and a GMM on these points. Thus, also accounting for the correlations between these input points.

Remark 4 *While in Problem 1 we seek for a Guassian approximation of a SNN, one could also consider the complementary problem of finding the parameters of a SNN that best approximate a given GMM. Such a problem also has wide application. In fact, a solution to this problem would allow one to encode informative functional priors represented via Gaussian processes to SNNs, addressing one of the main challenges in performing Bayesian inference with neural networks (Flam-Shepherd et al., 2017; Tran et al., 2022). As the Wasserstein distance satisfies the triangle inequality and because there exist closed-form expressions for an upper bound on the Wasserstein distance between Gaussian Mixture distributions (Delon and Desolneux, 2020), a solution to Problem 1 can be readily used to address the above mentioned complementary problem. This will be detailed in Section 7.2 and empirically demonstrated in Section 8.3.*

Remark 5 *Note that in Problem 1 we consider a GMM of arbitrary finite size. This is because such a model can approximate any continuous distribution arbitrarily well, and, consequently, guarantees of convergence to satisfy Problem 1 can be obtained, as we will prove in Section 6. However, Problem 1 could be restricted to the single Gaussian case, in which case Problem 1 reduces to finding the GP that best approximates a SNN. However, in this case, because $p_{nn(\mathcal{X})}$ is not necessarily Gaussian, the resulting distance bound between the GP and the SNN may not always be made arbitrarily small.*

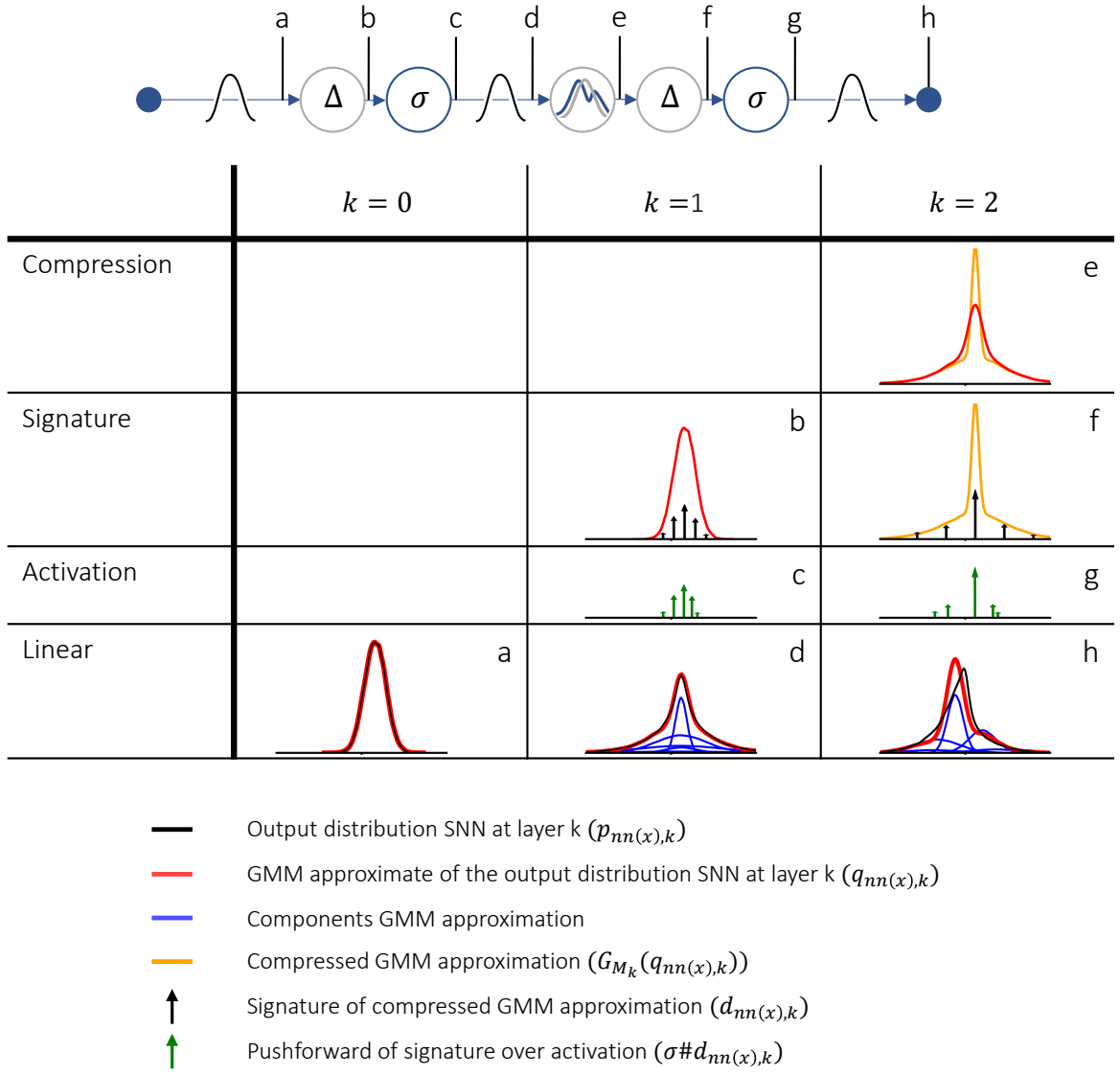


Figure 2: Illustration of the iterative approximation procedure of the output distribution of a single-input single-output SNN with two hidden layers ($K = 2$) and one neuron per layer ($n_1, n_2 = 1$) at a single input point by a Gaussian Mixture distribution.

4.2.1 Approach Outline

Our approach to solving Problem 1 is based on iteratively approximating the output distribution of each layer of a SNN as a mixture of Gaussian distributions and quantifying the approximation error introduced by each operation. As illustrated in Figure 2, following the definition of $p_{nn(x)}$ in Eqn (4), the first step is to perform an affine combination of the input point x with the parameters of the first layer (step a). Because the SNN parameters are assumed to be jointly Gaussian and jointly Gaussian random variables are closed under affine combination, this affine operation leads to a Gaussian distribution. Then, before propagating this distribution through an activation function, a signature operation on the resulting distribution is performed; that is, the continuous

distribution is approximated into a discrete distribution (step b). After the signature approximation operation, the resulting discrete distribution is passed through the activation function (step c) and an affine combination with the parameters of the next layer is performed (step d). Under the assumption that the parameters are jointly Gaussian, this affine operation results into a Gaussian mixture distribution of size equal to the support of the discrete distribution resulting from the signature operation. To limit the computational burden of propagating a Gaussian mixture with a large number of components, a compression operation is performed that compresses this distribution into another mixture of Gaussian distributions with at most M components for a given M (step e). After this, a signature operation is performed again on the resulting Gaussian mixture distribution (step f), and the process is repeated until the last layer. Consequently, to construct $q_{nn(x)}$, the Gaussian mixture approximation of $p_{nn(x)}$, our approach iteratively performs four operations: affine transformation, compression, signature approximation, and propagation through an activation function. To quantify the error in our approach, we derive formal error bounds in terms of the Wasserstein distance for each of these operations and show how the resulting bounds can be combined via interval arithmetic to an upper bound of $\mathbb{W}_2\left(p_{nn(x)}, q_{nn(x)}\right)$.

In what follows, first, as it is one of the key elements of our approach, in Section 5 we introduce the concept of signature of a probability distribution and derive error bounds on the 2-Wasserstein distance between a Gaussian mixture distribution and its signature approximation. Then, in Section 6 we formalize the approach described in Figure 2 to approximate a SNN with a GMM and derive bounds for the resulting approximation error. Furthermore, in Subsection 6.4 we prove that $q_{nn(x)}$, the Gaussian mixture approximation resulting from our approach, converges uniformly to $p_{nn(x)}$, the distribution of the SNN, that is, at the cost of increasing the support of the discrete approximating distributions and the number of components of $q_{nn(x)}$, the 1- and 2-Wasserstein distance between $q_{nn(x)}$ and $p_{nn(x)}$ can be made arbitrarily small. Then, in Section 7 we present a detailed algorithm of our approach. Finally, we conclude our paper with Section 8, where an empirical analysis illustrating the efficacy of our approach is performed.

5. Approximating Gaussian Mixture Distributions by Discrete Distributions

One of the key steps of our approach is to perform a signature operation on a Gaussian Mixture distribution, that is, a Gaussian Mixture distribution is approximated with a discrete distribution, called a signature (step b and f in Figure 2). In Subsection 5.1 we formally introduce the notion of the signature of a probability distribution. Then, in Subsection 5.2 we show how for a Gaussian Mixture distribution a signature can be efficiently computed with guarantees on the closeness of the signature and the original Gaussian mixture distribution in the $\mathbb{W}_2$-distance.

5.1 Signatures: the Discretization of a Continuous Distribution

For a set of points $\mathcal{C} = \{c_i\}_{i=1}^N \subset \mathbb{R}^d$ called *locations*, we define $\Delta_{\mathcal{C}} : \mathbb{R}^d \to \mathcal{C}$ as the function that assigns any $z \in \mathbb{R}^d$ to the closest point in $\mathcal{C}$, i.e.,

$$\Delta_{\mathcal{C}}(z) \coloneqq \underset{c \in \mathcal{C}}{\operatorname{argmin}} \|z - c\|.$$

The pushforward operation induced by $\Delta_{\mathcal{C}}$ is a mapping from $\mathcal{P}(\mathbb{R}^d)$ to $\mathcal{D}_N(\mathbb{R}^d)$ and defines the *signature* of a probability distribution. That is, as formalized in Definition 6 below, a signature induced by $\Delta_{\mathcal{C}}$ is an approximation of a continuous distribution with a discrete one of support with a cardinality N.

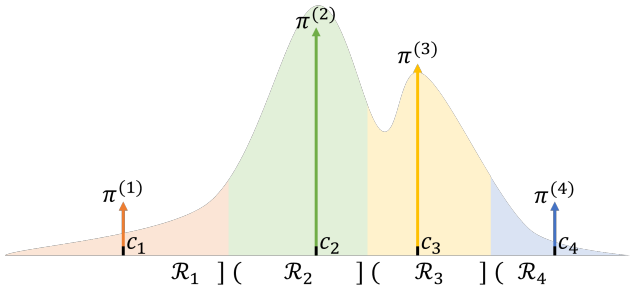


Figure 3: Signature of a continuous probability distribution. The signature is defined by locations $\{c_i\}_{i=1}^4$ which imply Voronoi partition $\{\mathcal{R}_i\}_{i=1}^4$. The continuous distribution is discretized w.r.t. the partition by storing all the probability mass $\pi^{(i)}$ covered by $\mathcal{R}_i$ at c_i.

Definition 6 (Signature of a Probability Distribution) *The signature of a probability distribution $p \in \mathcal{P}(\mathbb{R}^d)$ w.r.t. points $\mathcal{C} = \{c_i\}_{i=1}^N \subset \mathbb{R}^d$ is the discrete distribution $\Delta_{\mathcal{C}} \# p = \sum_{i=1}^N \pi^{(i)} \delta_{c_i} \in \mathcal{D}_N(\mathbb{R}^d)$, where $\pi^{(i)} = \mathbb{P}_{z \sim p}[z \in \mathcal{R}_i]$ with*

$$\mathcal{R}_i \coloneqq \left\{ z \in \mathbb{R}^d \colon \|z - c_i\| \leq \|z - c_j\|, \forall j \in \{1, \ldots, N\}, j \neq i \right\}. \tag{6}$$

The intuition behind a signature is illustrated in Figure 3: the signature of distribution p is a discretization of p that assigns to each location $c_i \in \mathcal{C}$ a probability given by the probability mass of $\mathcal{R}_i$ according to p. Note that partition $\{\mathcal{R}_i\}_{i=1}^N$, as defined in Eqn (6), is the Voronoi partition of $\mathbb{R}^d$ w.r.t. the Euclidean distance. Hence, the signature of a probability distribution can be interpreted as a discretization of the probability distribution w.r.t. a Voronoi partition of its support. In the remainder, we refer to $\mathcal{C}$ as the locations of the signature, $|\mathcal{C}|$ as the signature size, $\{\mathcal{R}_i\}_{i=1}^N$ as the partition of a signature, and we call Δ the signature operation.

Remark 7 *The signature of a probability distribution admits a closed-form expression when π in Definition 6 is analytically tractable, i.e., the probability mass over the signature's partition $\{\mathcal{R}_i\}_{i=1}^N$, is analytically tractable. For a Gaussian distribution, this is the case when the regions $\mathcal{R}_i$ are aligned with the geometry induced by the covariance matrix. In particular, for $\mathcal{N}(\mu, \Sigma)$, axis-aligned hyperrectangles in the eigenbasis of Σ, ellipsoids of the form $\{x \colon (x-\mu)^T \Sigma^{-1} (x-\mu) \leq c\}$, and unions or intersections of such sets all admit a closed-form expressions for their probability under $\mathcal{N}(\mu, \Sigma)$ (Genz and Bretz, 2009).*

As a consequence of Remark 7, for a Gaussian mixture distribution with components with different eigenbases of their covariance, we apply the signature operation separately to each Gaussian distribution in the mixture. Specifically, for a Gaussian mixture distribution $p = \sum_{i=1}^N \pi^{(i)} p_i$ and a set of signature locations $\boldsymbol{\mathcal{C}} = \{\mathcal{C}_i\}_{i=1}^N$, we define the *component-wise signature* of p as[8]

$$\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}} \# p \coloneqq \sum_{i=1}^N \pi^{(i)} \Delta_{\mathcal{C}_i} \# p_i. \tag{7}$$

In the next subsection, we show how to efficiently bound $\mathbb{W}_2(p, \boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}} \# p)$.

8. An alternative approach would be to apply the same signature locations to all components in the mixture, and approximate the resulting discrete distribution via numerical integration. However, this becomes intractable when component covariances differ significantly.

Remark 8 *The approximation of a continuous distribution by a discrete distribution, also called a particle- or quantization approximation, is a well-known concept in the literature (Graf and Luschgy, 2000; Pages and Wilbertz, 2012). The notion of the signature of a probability distribution introduced here is unique in that the discrete approximation is fully defined by the Voronoi partition of the support of the continuous distribution, thereby connecting the concept of signatures to semi-discrete optimal transport (Peyré et al., 2019).*

5.2 Wasserstein Bounds for the Signature Operation

The computation of $W_2(p, \Delta_{\mathcal{C}}\#p)$ requires solving a semi-discrete optimal transport problem, where we need to find the optimal transport plan from each point z to a specific location $c_i \in \mathcal{C}$ (Peyré et al., 2019). Luckily, as illustrated in the following proposition, which is a direct consequence of Lemma 3.1 in (Canas and Rosasco, 2012), for the case of a signature of a probability distribution, the resulting transport problem can be solved exactly.

Proposition 9 *Let $\rho \geq 1$, then for a probability distribution $p \in \mathcal{P}_\rho(\mathbb{R}^n)$, and signature locations $\mathcal{C} = \{c_i\}_{i=1}^N \subset \mathbb{R}^n$, we have that*

$$\mathbb{W}_\rho^\rho(p, \Delta_{\mathcal{C}}\#p) = \sum_{i=1}^N \pi^{(i)} \mathbb{E}_{z\sim p}[\|z - c_i\|^\rho \mid z \in \mathcal{R}_i].$$

According to Proposition 9, transporting the probability mass at each point z to a discrete location c_i based on the Voronoi partition of $\mathbb{R}^d$ guarantees that the cost $\|z - c_i\|$ is the smallest and leads to the smallest possible transportation cost, i.e., the optimal transportation strategy for the Wasserstein Distance. Proposition 9 is general and guarantees that for any distribution $p \in \mathcal{P}_2(\mathbb{R}^n)$, $\mathbb{W}_2(p, \Delta_{\mathcal{C}}\#p)$ only depends on the probability mass and the conditional 2-moment of p w.r.t. regions $\mathcal{R}_i \in \{\mathcal{R}_1, ..., \mathcal{R}_N\}$. In the following proposition, we show how closed-form expressions for $\mathbb{W}_2^2(p, \Delta_{\mathcal{C}}\#p)$ can be derived for univariate Gaussian distributions. This result is then generalized in Corollary 11 to general multivariate Gaussian distributions.

Proposition 10 *For $p = \mathcal{N}(0, 1)$, signature locations $\mathcal{C} = \{c_i\}_{i=1}^N \subset \mathbb{R}$ and associated partition $\{\mathcal{R}_i\}_{i=1}^n$ with $\mathcal{R}_i = [l_i, v_i] \subseteq \mathbb{R} \cup \{-\infty, \infty\}$ for each i, it holds that*

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}}\#\mathcal{N}(0, 1), \mathcal{N}(0, 1)) = \sum_{i=1}^N \pi^{(i)} \left[\nu_i + (\mu_i - c_i)^2\right] \tag{8}$$

with,

$$\pi^{(i)} = \Phi(u_i) - \Phi(l_i), \quad \mu_i = \frac{1}{\pi^{(i)}}[\phi(l_i) - \phi(u_i)], \quad \nu_i = 1 + \frac{1}{\pi^{(i)}}[l_i\phi(l_i) - u_i\phi(u_i)] - \mu_i^2 \tag{9}$$

where ϕ and Φ are the pdf and cdf of a standard (univariate) Gaussian distribution, respectively, i.e., $\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$ and $\Phi(x) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right]$.

In Proposition 10, μ_i and ν_i are the mean and variance of a standard univariate Gaussian distribution restricted on $[l_i, u_i]$. Note that some of the regions in the partition will be unbounded. However, as the standard Gaussian distribution exponentially decays to zero for large x, i.e., $\lim_{x\to\infty} \exp(x^2)\mathcal{N}(x \mid 0, 1) = 0$, it follows that the bound in Eqn (8) is finite even if some of the regions in $\{\mathcal{R}_i\}_{i=1}^N$ are necessarily unbounded.

A corollary of Proposition 10 is Corollary 11, where we extend Proposition 10 to general multivariate Gaussian distributions under the assumption that locations $\mathcal{C}$ are such that sets $\{\mathcal{R}_i\}_{i=1}^N$

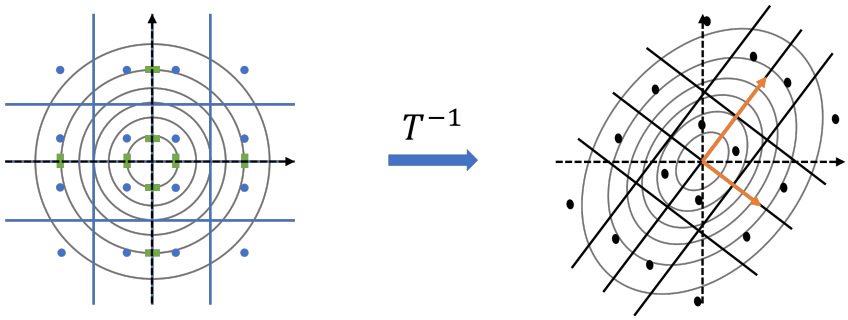


Figure 4: Construction of the signature of a 2D Gaussian distribution (black dots) using the signature of a standard Gaussian distribution (blue dots) as a template. In the space induced by T, that is, the basis of the covariance matrix (orange arrows) of the Gaussian distribution, all dimensions of the Gaussian distribution are independent. Hence, we can take the cross-product of the signatures of each univariate Gaussian marginal in dimension j of the transformed space with signature locations $\mathcal{C}^j$ (green stripes) as a template. The signature in the original space is then obtained by taking the post image of the template under T^{-1}. The blue and black lines represent the edges of the Voronoi partition associated with the signature locations in the transformed and original space, respectively.

define a grid in the transformed space induced by the basis of the covariance matrix that we denote as T. As illustrated in Figure 4, it is always possible to satisfy this assumption by taking $\mathcal{C}$ as the image of an axis-aligned grid of points under transformation T^{-1}, where T^{-1} can be computed via an eigendecomposition (Kasim, 2020) or Cholesky decomposition (Davis et al., 2016) of the covariance matrix.[9]

Corollary 11 (of Proposition 10) *For $\mathcal{N}(m, \Sigma) \in \mathcal{G}(\mathbb{R}^n)$, let matrix $T \in \mathbb{R}^{n\times n}$ be such that $T = \mathrm{diag}(\lambda)^{-\frac{1}{2}} V^T$ where $\mathrm{diag}(\lambda) = V^T \Sigma V$ is a diagonal matrix whose entries are the eigenvalues of Σ, and V is the corresponding orthogonal (eigenvector) matrix. Further, let $\mathcal{C} = \{c_i\}_{i=1}^N \subset \mathbb{R}^n$ be a set of signature locations on a grid in the transformed space induced by T, i.e.,*

$$\mathrm{Post}(\mathcal{C} - \{m\}, T) = \mathcal{C}^1 \times \mathcal{C}^2 \times \ldots \times \mathcal{C}^n \tag{10}$$

with $\mathcal{C}^j = \{c_i\}_{i=1}^{N_j}$ the set of signature locations in the transformed space for dimension j. Then,

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}} \# \mathcal{N}(m, \Sigma), \mathcal{N}(m, \Sigma)) = \sum_{j=1}^{n} \lambda^{(j)} \mathbb{W}_2^2 \left(\Delta_{\mathcal{C}^j} \# \mathcal{N}(0, 1), \mathcal{N}(0, 1)\right).$$

In the case of Gaussian mixture distributions, i.e., where $p = \sum_{i=1}^M \tilde{\pi}^{(i)} \mathcal{N}(m_i, \Sigma_i)$, and the component-wise signature operation defined in Eqn (7), i.e., $\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}}$ with $\boldsymbol{\mathcal{C}} = \{\mathcal{C}_{i=1}^N\}$, we have that

$$\mathbb{W}_2^2 \left(p, \boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}} \# p\right) \leq \sum_{i=1}^{M} \tilde{\pi}^{(i)} \mathbb{W}_2^2 \left(p_i, \Delta_{\mathcal{C}_i} \# p_i\right) =: \hat{\mathbb{W}}_{2, \boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}}}^2 (p). \tag{11}$$

9. Given the eigenvalues vector λ and eigenvector matrix V of a covariance matrix Σ, we can take $T^{-1} = V\mathrm{diag}(\lambda)$. In the case of a degenerate multivariate Gaussian, where Σ is not full rank, we can take $\mathcal{C}$ as the image of an axis-aligned grid of points in a space of dimension $\mathrm{rank}(\Sigma)$, under transformation $T^{-1} = (V\mathrm{diag}(\lambda)^{\frac{1}{2}})^{(:,1:\mathrm{rank}(\Sigma))}$.

That is, for a mixture of Gaussian distributions, the 2-Wasserstein distance between the mixture and its component-wise signature is bounded by the weighted sum of the 2-Wasserstein distances between each Gaussian component and its signature.[10] Note that, under the assumption that the signature locations of each component satisfy Eqn (10), this upper bound admits a closed form as given by Corollary 11.

Remark 12 *For a multivariate Gaussian $p = \mathcal{N}(m, \Sigma)$, the set $\mathcal{C}$ of $N > 1$ signature locations that minimize $\mathbb{W}_2(\Delta_{\mathcal{C}}\#p, p)$ is generally non-unique, and finding any such set is computationally intractable (Graf and Luschgy, 2000). However, for univariate Gaussians, an optimal signature placement strategy exists as outlined in Pagès and Printems (2003) and will be used in Subsection 7.1 to construct grid-constrained signature locations $\mathcal{C}$ for multivariate Gaussians.*

5.3 Convergence Rates for Signature Approximations of Gaussian Mixtures

In the previous section, we showed how we can obtain a closed form expression for $\mathbb{W}_2(p, \Delta_{\mathcal{C}}\#p)$ when p is a multivariate Gaussian distribution, and that this expression can be used to upper bound the 2-Wasserstein distance error from the component-wise signature $\boldsymbol{\Delta}_{\mathcal{C}}\#p$ when p is a mixture of Gaussian distributions. Here, we show that this error converges uniformly to zero as the number of signature locations increases. We start with the case where p is a univariate Gaussian distribution, and then extend to mixtures of Gaussian distributions. To our knowledge, the resulting uniform, non-asymptotic convergence rates are novel. Indeed, existing works on quantization for Gaussian measures (e.g., Graf and Luschgy, 2000; Pagès and Printems, 2003) typically provide asymptotic rates and existence results for optimal N-point quantizers, rather than the explicit uniform-in-N error bounds derived here.

Proposition 13 *For $p = \mathcal{N}(0, 1)$, consider $N \in \mathbb{N}$ signature locations on a uniform grid centered at zero with spacing $(2\sqrt{\log N})/N$ between consecutive points,*[11] *that is*

$$\mathcal{C} = \left\{ \frac{2\sqrt{\log N}}{N} (2i - N - 1) : i \in \{1, \dots, N\} \right\}. \tag{12}$$

Then, for all $N \in \mathbb{N}$,

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}}\#\mathcal{N}(0,1), \mathcal{N}(0,1)) \leq \frac{4 \log N}{N^2} + r(N), \tag{13}$$

where the remainder term

$$r(N) = \begin{cases} 1, & N = 1, \\ \frac{4}{N^2\sqrt{2\pi}} \left(\sqrt{\log N} + \frac{1}{4\sqrt{\log N}} \right), & N \geq 2. \end{cases} \tag{14}$$

The proof of Proposition 13 (given in the Appendix) follows by applying Proposition 10 to write the $\mathbb{W}_2$ error as a sum over the Voronoi partition induced by $\mathcal{C}$, then bound each interval's contribution by its squared radius and control the unbounded intervals using Gaussian tail bounds (Mills' ratio).

Remark 14 *For $N \geq 2$, the remainder term of the bound of Proposition 13 (Eqn 14) is*

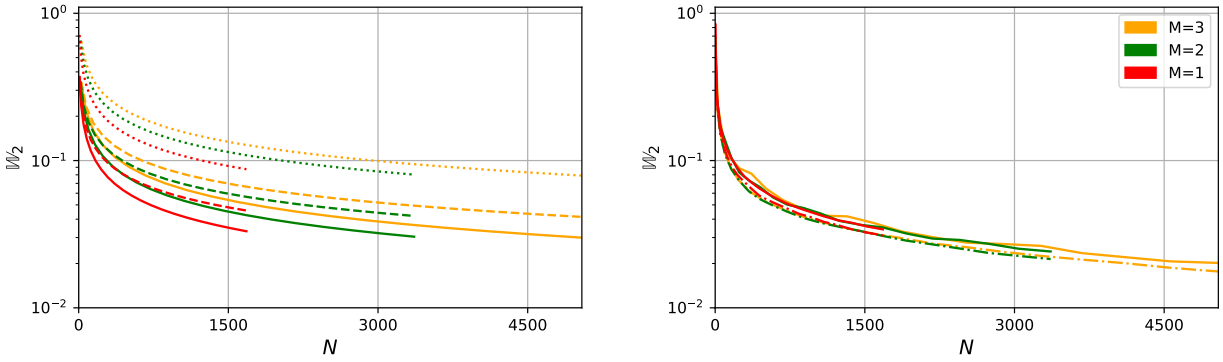
$$r(N) = \frac{4\sqrt{\log N}}{N^2\sqrt{2\pi}} + O\left(\frac{1}{N^2\sqrt{\log N}} \right).$$

10. This trivial result is a special case of Lemma 31 in the Appendix on the Wasserstein distance between mixtures.
11. We choose the spacing $(2\sqrt{\log N})/N$ as it balances the discretization error in the central interval and the Gaussian tail mass. Other spacings trade off these contributions differently: for example, a spacing of $(\log N)/N$ would reduce the remainder term $r(N)$ but increase the leading term to $(\log^2 N)/N^2$.

Hence $r(N) = O\left(\frac{\sqrt{\log N}}{N^2}\right)$ *and* $r(N) = o\left(\frac{\log N}{N^2}\right)$*, so the bound in Eqn* (13) *is dominated by its first term* $\frac{4 \log N}{N^2}$*, and consequently*

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}} \# \mathcal{N}(0,1), \mathcal{N}(0,1)) \leq \frac{4 \log N}{N^2}\big(1 + o(1)\big).$$

That is, for a uniform grid, the squared W_2 *error decays on the order of* $(\log N)/N^2$*. In one dimension, Theorem 6.2 of Graf and Luschgy (2000) implies that the minimal achievable squared* $\mathbb{W}_2$ *error of any* N*-location signature approximating a univariate Gaussian decays as* $1/N^2$ *up to constants. Thus the uniform grid in Proposition 13 attains the optimal polynomial rate* $1/N^2$ *up to a single logarithmic factor. The* $\log N$ *term arises from using a uniform construction that provides an explicit bound for all* N*, with a remainder of strictly lower order than the leading term. In practice, adaptive (non-uniform) placement (Remark 12) can remove this logarithmic factor and improve the constants in the error bound, as illustrated in Figure 5.*



(a) Formal upper bounds on $\mathbb{W}_2$ for uniform and adaptive grid-constrained signatures.

(b) Empirical $\mathbb{W}_2$ comparisons of grid-constrained and approximate optimal signatures.

Figure 5: The 2-Wasserstein distance between a 2D Gaussian mixture distribution with M components and signature approximations with N locations. Solid lines correspond to signatures with non-uniform grid-constrained locations obtained via Algorithm 2, while dashed lines correspond to signatures with uniform grid-constrained locations as in Corollary 15. Dash-dotted lines correspond to signatures with locations given by the cluster centers obtained via N-Means on 10^5 Monte Carlo samples from the GMM, which provide an approximation to the optimal signature locations (Pagès and Printems, 2003). For grid-constrained signatures, formal bounds are computed using Algorithm 2, while empirical estimates are computed from 10^4 samples from each distribution by computing the 2-Wasserstein distance between the resulting empirical distributions. The dashed reference curves show the convergence rate from Corollary 15, which corresponds to the uniform grid. For comparability, each mixture has an expected squared ℓ_2 norm of one, so the reported distance equals the relative 2-Wasserstein distance defined in Eqn (27).

Analogously to how Proposition 10 extends to mixtures of Gaussian distributions, Proposition 13 extends to mixtures of multivariate Gaussian distributions, as shown in the following corollary.

Corollary 15 (of Proposition 13) *Let $p = \sum_{i=1}^{M} \tilde{\pi}^{(i)} \mathcal{N}(m_i, \Sigma_i)$. For each component i of p, define a set of signature locations $\mathcal{C}_i \in \mathbb{R}^n$ as in Eqn* (10)*, using uniform one-dimensional grid of points $\mathcal{C}_i^j$ of size $N_{i,j}$ as in Eqn* (12) *for each dimension j.*

$$\mathbb{W}_2^2\left(p, \mathbf{\Delta}_{\{\mathcal{C}_i\}_{i=1}^M} \# p\right) \leq \sum_{i=1}^{M} \tilde{\pi}^{(i)} \sum_{j=1}^{n} \lambda_i^{(j)} \left(\frac{4 \log N_{i,j}}{N_{i,j}^2} + r(N_{i,j}) \right)$$

where λ is the vector of eigenvalues of Σ_i.

Corollary 15 follows from Eqn (11), which decomposes the mixture error into a mixture-weighted sum of component errors. For each component we diagonalize Σ_i, apply Corollary 11 to separate dimensions, and then invoke Proposition 13 on each one-dimensional uniform grid of points $\mathcal{C}_i^j$, yielding the closed-form expression for the error bound after summing over i and j.

In Figure 5, we examine the convergence rate of the approximation error for 2D Gaussian mixtures decays as the number of signature locations increases, comparing three constructions: uniform Cartesian product grids (Corollary 15), adaptive (non-uniform) grids, and an empirical proxy for the optimal signature locations. Figure 5a confirms that adaptive grids substantially reduce the $\mathbb{W}_2$ error relative to the uniform case (see Remark 14). Figure 5b further shows that adaptive placement already captures most of the attainable improvement, with only modest additional gains from unconstrained (cluster-based) locations for the distinct-mode mixtures considered. Appendix F provides further discussion and additional experiments on the conservatism of the component-wise signature bound in Eqn. (11).

6. Stochastic Neural Networks as Gaussian Mixture Models

In this section, we detail and formalize our approach as illustrated in Figure 2 to iteratively approximate a SNN with a GMM at a finite set of input points $\mathcal{X}$. We first consider the case where $\mathcal{X} = \{x\}$, i.e., we approximate the distribution of a SNN over a single input point. The extension to finite set of input points is considered in Section 6.3. Last, in Section 6.4, we prove that $q_{nn(\mathcal{X})}$, i.e., the Gaussian mixture approximation resulting from our approach, converges to $p_{nn(\mathcal{X})}$ in Wasserstein distance. That is, the error between $p_{nn(\mathcal{X})}$ and $q_{nn(\mathcal{X})}$ can be made arbitrarily small by increasing the number of signature locations and GMM components.

6.1 Gaussian Mixture Model Approximation of a Neural Network

As shown in Eqn (4), $p_{nn(x)}$, the output distribution of a SNN at input point x, can be represented as a composition of K stochastic operations, one for each layer. To find a Gaussian mixture distribution $q_{nn(x)}$ that approximates $p_{nn(x)}$, our approach is based on iteratively approximating each of these operations with a GMM, as illustrated in Figure 2. Our approach can then be formalized as in Eqn (15) below for $k \in \{1, \ldots, K\}$:

$$q_{nn(x),1} = \mathbb{E}_{z_0 \sim \delta_x}[L^{w_0}(z_0) \# p_{w_0}] \qquad \text{(Initialization and Affine operation)} \tag{15a}$$

$$d_{nn(x),k} = \mathbf{\Delta}_{\boldsymbol{\mathcal{C}}_k} \# G_{M_k}(q_{nn(x),k}) \qquad \text{(Compression and Signature operations)} \tag{15b}$$

$$q_{nn(x),k+1} = \mathbb{E}_{z_k \sim d_{nn(x),k}}[L^{w_k}(\sigma(z_k)) \# p_{w_k}] \qquad \text{(Activation and Affine operations)} \tag{15c}$$

$$q_{nn(x)} = q_{nn(x),K+1} \qquad \text{(Output)} \tag{15d}$$

where $\boldsymbol{\mathcal{C}}_k = \{\mathcal{C}_{k,i} \subset \mathbb{R}^{n_k}\}_{i=1}^{M_k}$ denotes the signature locations for layer k. $G_{M_k} : \mathcal{G}_\infty(\mathbb{R}^{n_k}) \to \mathcal{G}_{M_k}(\mathbb{R}^{n_k})$ is a compression operation that compresses the Gaussian mixture distribution $q_{nn(x),k}$ into a Gaussian Mixture distribution with at most M_k components, thus limiting the approximation's complexity. We show how operation G_{M_k} is performed using moment matching in Section 7.1.

Eqn (15) consists of the following steps: $q_{nn(x),1}$ is the distribution resulting from an affine combination of the input x with the parameters at the first layer, $d_{nn(x),k}$ is the result of a compression and signature operation of $q_{nn(x),k}$, the output of the previous layer. The approximate output distribution $q_{nn(x),k+1}$ at each layer k is obtained by marginalizing $L^{w_k}(\sigma(z_k))\#p_{w_k}$ w.r.t. $d_{nn(x),k}$. We recall that for any fixed z_k, and Gaussian distribution p_{w_k}, $L^{w_k}(\sigma(z_k))\#p_{w_k}$ is Gaussian, as Gaussian distributions are closed w.r.t. affine combination. Consequently, as for all k, $d_{nn(x),k}$ is a discrete distribution, $q_{nn(x),k+1}$ is a Gaussian mixture distribution with a size equal to the support of $d_{nn(x),k}$.

Example 1 *Consider the case where L^{w_k} is a fully-connected layer with weight matrix $W \in \mathbb{R}^{n_{k+1}\times n_k}$ and bias vector $b \in \mathbb{R}^{n_{k+1}}$, i.e., $L^{w_k}(z) = Wz + b$. Assume* $\mathrm{vec}(W)$ *and b are distributed according to $\mathcal{N}(\mathrm{vec}(M), \Sigma)$ and $\mathcal{N}(m_b, \Sigma_b)$, respectively, where* $\mathrm{vec}(\cdot)$ *stacks the columns of a matrix into a single vector. Then, the approximate output distribution at the $(k+1)$-th layer, $q_{nn(x),k+1}$, as defined in Eqn* (15c) *for $M_k = 1$, signature locations $\mathcal{C}_{k,1} = \{c_i\}_{i=1}^{N_k}$ and partition $\{\mathcal{R}_i\}_{i=1}^{N_k}$ from Eqn* (6)*, can be written as*

$$q_{nn(x),k+1} = \sum_{i=1}^{N_k} \mathbb{P}_{z\sim G_1(q_{nn(x),k})}[z \in \mathcal{R}_i] \cdot \mathcal{N}\left(m_b + M\sigma(c_i), \Sigma_b + \sum_{j,l=1}^{n_{k+1}} \sigma(c_i)^{(j)}\sigma(c_i)^{(l)}\Sigma_{jl}\right)$$

where Σ_{jl} denotes the (j,l)-th block of the covariance matrix Σ, each block of size $n_k \times n_k$, i.e., the covariance between the j-th and l-th columns of W.

Note that the approximation scheme in Eqn (15) allows for an arbitrary choice of signature locations $\mathcal{C}_{k,i}$ for the i-th component at each layer k. To obtain a closed-form expression for $q_{nn(x)}$, the locations $\boldsymbol{\mathcal{C}}_k$ should be selected as discussed in Remark 7; that is, so that $\mathbb{P}_{z\sim G_1(q_{nn(x),k})}[z \in \mathcal{R}]$ in Example 1 is analytically tractable for all regions $\mathcal{R}$.

Remark 16 *The approach described in Eqn* (15) *leads to a Gaussian mixture distribution $q_{nn(x)}$ under the assumption that the parameter distributions p_{w_k} are Gaussian. In the more general case, where p_{w_k} is non-Gaussian, one can always recursively approximate $L^{w_k}(z)\#p_{w_k}$ for each k by a discrete distribution using the signature operation. That is, $L^{w_k}(z)\#p_{w_k}$ in Eqn* (15c) *is replaced with $L^{w_k}(z)\#\Delta_{\mathcal{C}_{w_k}}\#p_{w_k}$, where $\mathcal{C}_{w_k}$ is the set of signature locations. Applying this additional operation, Eqn* (15) *will lead to a discrete distribution for $q_{nn(x)}$.*

6.2 Wasserstein Distance Guarantees

Since $q_{nn(x)}$ is built as a composition of K iterations of the stochastic operations in Eqn (15), to bound $\mathbb{W}_2\left(p_{nn(x)}, q_{nn(x)}\right)$ we need to formally quantify the error introduced by each of the steps in Eqn (15). A summary of how each of the operations occurring in Eqn (15) modifies the Wasserstein distance between two distributions is provided in Table 1 (details are given in Appendix B.2). These bounds are then composed using interval arithmetic to bound $\mathbb{W}_2\left(p_{nn(x)}, q_{nn(x)}\right)$ in the following theorem.

Theorem 17 *Let $p_{nn(x)}$ be the output distribution of a SNN with K hidden layers for input $x \in \mathbb{R}^{n_0}$ and $q_{nn(x)}$ be a Gaussian mixture distribution built according to Eqn* (15)*. Iteratively define $\hat{\mathbb{W}}_2$ as*

$$\hat{\mathbb{W}}_{2,1} = 0, \tag{16a}$$

$$\hat{\mathbb{W}}_{2,k+1} = \mathcal{L}_k \Big[\hat{\mathbb{W}}_{2,k} + \mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right) + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(x),k})\right)\Big], \qquad \forall k \in \{1,\dots,K\}, \tag{16b}$$

$$\hat{\mathbb{W}}_2 = \hat{\mathbb{W}}_{2,K+1}, \tag{16c}$$

where $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}}}$ is defined in Eqn (11)*, $\mathbb{MW}_2$ as defined in Eqn* (2)*, and*

$$\mathcal{L}_k \coloneqq \mathcal{L}_\sigma \mathbb{E}_{w_k \sim p_{w_k}}[\mathcal{L}_{L^{w_k}}^2]^{\frac{1}{2}}, \tag{17}$$

with $\mathcal{L}_\sigma$ and $\mathcal{L}_{L^{w_k}}$ the Lipschitz constants of the activation function σ and affine operation L^{w_k}, respectively. Then, it holds that

$$\mathbb{W}_2\left(p_{nn(x)}, q_{nn(x)}\right) \leq \hat{\mathbb{W}}_2(p_{nn(x)}, q_{nn(x)}) \coloneqq \hat{\mathbb{W}}_2.$$

The proof of Theorem 17 is reported in Appendix B.2 and is based on the triangle inequality property of the 2-Wasserstein distance, which allows us to bound $\mathbb{W}_2\left(p_{nn(x)}, q_{nn(x)}\right)$ iteratively over the hidden layers. Note that Theorem 17 depends on the quantities $\mathbb{E}_{w_k \sim p_{w_k}}[\mathcal{L}_{L^{w_k}}^2]^{1/2}$, $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(x),k})\right)$, and $\mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)$, which can be computed as follows. First, $\mathbb{E}_{w_k \sim p_{w_k}}[\mathcal{L}_{L^{w_k}}^2]^{1/2}$, is the expectation of any Lipschitz constant $\mathcal{L}_{L^w}$ (any constant satisfying the Lipschitz inequality, not necessarily the minimal one) of the affine operation L^{w_k}, for which closed-form expressions for both feedforward and convolutional layers are given in Appendix C. Second, $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(x),k})\right)$ can be computed using Corollary 11, when the sets of signature locations satisfy the grid-regularity constraint in Eqn (10). Finally, $\mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)$ is obtained by computing the $\mathbb{MW}_2$-distance between Gaussian mixtures distributions (see Definition 2), which requires solving a finite discrete linear program with M_k times the number of components of $q_{nn(\mathcal{X}),k}$ optimization variables and coefficient given by the pairwise Wasserstein distances between the mixture's Gaussian components.

Remark 18 *If p_{w_k} is non-Gaussian, the additional approximation step introduced in Remark 16 is accounted for by an extra application of the triangle inequality, which introduces an additional error term $\mathbb{W}_2(p_{w_k}, \Delta_{\mathcal{C}_{w_k}} \# p_{w_k})$ in Eqn* (16b)*. While the conditional expectations in Proposition 9 may no longer admit closed-form expressions in this case, they can always be conservatively estimated using approximate numerical integration.*

Remark 19 *In Theorem 17, instead of bounding the error from the signature approximation of $G_{M_k}(q_{nn(x),k})$ in Eqn* (16b)*, propagated through the activation function σ, using the global Lipschitz constant $\mathcal{L}_\sigma$ of σ, one can alternatively use a bound based on the local Lipschitz constant $\mathcal{L}_{\sigma|\mathcal{R}_i}$ w.r.t. region $\mathcal{R}_i$, as reported in Table 1. Specifically, one can replace $\mathcal{L}_\sigma \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(x),k})\right)$ with*

$$\left(\sum_{i=1}^{N} \mathcal{L}_{\sigma|\mathcal{R}_i}^2 \pi^{(i)} \mathbb{E}_{z \sim G_{M_k}(q_{nn(x),k})}[\|z - c_i\|^\rho \mid z \in \mathcal{R}_i]\right)^{1/2}$$

in Eqn (16b)*. While generally tighter, this bound requires explicit computation of the local Lipschitz constant and the expectation w.r.t each region $\mathcal{R}_i$, which limits the efficiency of the grid structure $\mathcal{C}_i$ used in Corollary 11.*

Operation	Wasserstein Bounds
Affine	$\mathbb{W}_2^2\left(\mathbb{E}_{\tilde{z}\sim p}\left[L^w(\tilde{z})\#p_{w_k}\right], \mathbb{E}_{z\sim q}\left[L^w(z)\#p_w\right]\right) \leq \mathbb{E}_{w\sim p_w}\left[\mathcal{L}_{L^w}^2\right]\mathbb{W}_2^2(p,q)$
Activation	$\mathbb{W}_2(\sigma\#p, \sigma\#q) \leq \mathcal{L}_\sigma \mathbb{W}_2(p,q)$ $\mathbb{W}_2^2(\sigma\#p, \sigma\#\Delta_{\mathcal{C}}\#p) \leq \sum_{i=1}^N \mathcal{L}_{\sigma\|\mathcal{R}_i}^2 \pi^{(i)} \mathbb{E}_{z\sim p}[\|z-c_i\|^2 \mid z\in\mathcal{R}_i]$
Signature/ Compression	$\mathbb{W}_2(p, h\#q) \leq \mathbb{W}_2(p,q) + \mathbb{W}_2(q, h\#q)$, where $h \in \{\Delta_{\mathcal{C}}, G_M\}$

Table 1: Summary of the bounds on the 2-Wasserstein distance between two distributions $p, q \in \mathcal{P}_2(\mathbb{R}^n)$, obtained by pushing forward p and q through common SNN operation types. Here, $\mathcal{L}_{L^w}$ denotes the global Lipschitz constant of a Lipschitz function L^w, parametrized by $w \in \mathbb{R}^d$ and distributed according to $p_w \in \mathcal{P}(\mathbb{R}^d)$, and $\mathcal{L}_\sigma$ and $\mathcal{L}_{\sigma|\mathcal{R}_i}$ denote, respectively, the global Lipschitz constant and the Lipschitz constant over region $\mathcal{R}_i$ of a Lipschitz function σ. Proofs of these results are provided in Appendix B.2.

6.3 Approximation for Sets of Input Points

We now consider the case where $\mathcal{X} = \{x_1, ..., x_D\}$, that is, a finite set of input points, and extend our approach to a Gaussian mixture approximation of $p_{nn(\mathcal{X})}$. We first note that $p_{nn(\mathcal{X})}$ can be equivalently represented by extending Eqn (4) as follows, where in the equation below $\text{vec}(\mathcal{X}) = (x_1^T, \ldots, x_D^T)^T$ is the vectorization of $\mathcal{X}$,

$$p_{nn(\mathcal{X}),1} = \mathbb{E}_{z_0\sim\delta_{\text{vec}(\mathcal{X})}}\left[\boldsymbol{L}^{w_0}(z_0)\#p_{w_0}\right], \tag{18a}$$

$$p_{nn(\mathcal{X}),k+1} = \mathbb{E}_{z_k\sim p_{nn(\mathcal{X}),k}}\left[\boldsymbol{L}^{w_k}(\sigma(z_k))\#p_{w_k}\right], \qquad k\in\{1,\ldots,K\}, \tag{18b}$$

$$p_{nn(\mathcal{X})} = p_{nn(\mathcal{X}),K+1}, \tag{18c}$$

where $\boldsymbol{L}^{w_k} : \mathbb{R}^{D\cdot n_k} \to \mathbb{R}^{D\cdot n_k}$ is the stacking of D times L^k, i.e.,

$$\boldsymbol{L}^{w_k}\left((z_1^T, \ldots, z_D^T)^T\right) = \left(L^{w_k}(z_1)^T, \ldots, L^{w_k}(z_D)^T\right)^T.$$

That is, $p_{nn(\mathcal{X})}$ is computed by stacking D times the neural network for the inputs in $\mathcal{X}$. Following the same steps as in the previous section, $p_{nn(\mathcal{X})}$ can then be approximated by the Gaussian Mixture distribution $q_{nn(\mathcal{X})}$, for $k \in \{1, \ldots, K\}$ defined as

$$q_{nn(\mathcal{X}),1} = \mathbb{E}_{z_0\sim\delta_{\text{vec}(\mathcal{X})}}\left[\boldsymbol{L}^{w_0}(z_0)\#p_{w_0}\right], \qquad \text{(Initialization and Affine operation)} \tag{19a}$$

$$d_{nn(\mathcal{X}),k} = \boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}\#G_M(q_{nn(\mathcal{X}),k}), \qquad \text{(Compression and Signature operations)} \tag{19b}$$

$$q_{nn(\mathcal{X}),k+1} = \mathbb{E}_{z_k\sim d_{nn(\mathcal{X}),k}}\left[\boldsymbol{L}^{w_k}(\sigma(z_k))\#p_{w_k}\right], \qquad \text{(Activation and Affine operations)} \tag{19c}$$

$$q_{nn(\mathcal{X})} = q_{nn(\mathcal{X}),K+1}, \qquad \text{(Output)} \tag{19d}$$

where $\boldsymbol{\mathcal{C}}_k = \{\mathcal{C}_{k,i} \subset \mathbb{R}^{D\cdot n_k}\}_{i=1}^{M_k}$ are the sets of signature locations at layer k, and $M_k \in \mathbb{N}$ is the compression size. Theorem 17 implies the following corollary, which bounds the approximation error of $q_{nn(\mathcal{X})}$.

Corollary 20 (of Theorem 17) *Let $p_{nn(\mathcal{X})}$ be the joint distribution of a SNN with K hidden layers over a finite set of input points $\mathcal{X} = \{x_i\}_{i=1}^{D} \subset \mathbb{R}^{n_0}$ and $q_{nn(\mathcal{X})}$ be a Gaussian mixture distribution built according to Eqn (19). Iteratively define $\hat{\mathbb{W}}_2$ as*

$$\hat{\mathbb{W}}_{2,1} = 0, \tag{20a}$$

$$\hat{\mathbb{W}}_{2,k+1} = \mathcal{L}_k \Big[\hat{\mathbb{W}}_{2,k} + \mathbb{MW}_2\left(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\right) + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(\mathcal{X}),k})\right)\Big], \qquad \forall k \in \{1,\dots,K\}, \tag{20b}$$

$$\hat{\mathbb{W}}_2 = \hat{\mathbb{W}}_{2,K+1}, \tag{20c}$$

where $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}}}$ is defined in Eqn (11), $\mathbb{MW}_2$ as defined in Eqn (2), and $\mathcal{L}_k$ is as defined Eqn (17). Then, it holds that

$$\mathbb{W}_2(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}) \le \hat{\mathbb{W}}_2(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}) \coloneqq \hat{\mathbb{W}}_2.$$

Note that the effect of multiple input points on the error introduced by $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(\mathcal{X}),k})\right)$ and $\mathbb{MW}_2\left(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\right)$ depends on the specific choice of points in $\mathcal{X}$. For points in $\mathcal{X}$ where $q_{nn(\mathcal{X}),k}$ is highly correlated, the error will be similar to that of a single-input case. However, if there is little correlation, the error bound will increase linearly with the number of points.

6.4 Convergence Analysis

In Theorem 17 and Corollary 20, we derived error bounds on the distance between $q_{nn(\mathcal{X})}$ and $p_{nn(\mathcal{X})}$, which depend on the signature locations and on the size of the compressed mixtures. The following theorem shows that by appropriately selecting these quantities, the error bound can be made arbitrarily small.

Theorem 21 *Let $p_{nn(\mathcal{X})}$ be the output distribution of a SNN with K hidden layers for a finite set of inputs $\mathcal{X} \subset \mathbb{R}^{n_0}$. For any $\epsilon > 0$, for layer $k \in \{1,\dots,K\}$, choose a compression size $M_k \in \mathbb{N}$ such that*

$$\epsilon_k \coloneqq \frac{\epsilon}{K\prod_{i=k}^{K}\mathcal{L}_i} - \mathbb{MW}_2\left(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\right) > 0, \tag{21}$$

where $\mathcal{L}_i$ is defined in Eqn (17). Choose per-component, per-dimension signature grid sizes $N_{k,i,j} \in \mathbb{N}$ satisfying

$$\sum_{i=1}^{M_k} \tilde{\pi_k}^{(i)} \sum_{j=1}^{n_k} \lambda_{k,i}^{(j)} \left(\frac{4\log N_{k,i,j}}{N_{k,i,j}^2} + r(N_{k,i,j})\right) \le \epsilon_k^2, \tag{22}$$

where $\tilde{\pi}_k$ are the mixture weights, $\lambda_{k,i}$ is the vector of eigenvalues of the covariance matrix of the i-th component of $q_{nn(\mathcal{X}),k}$, and $r(\cdot)$ is given in Eqn (14). Construct for each component i of $q_{nn(\mathcal{X}),k}$ a set of signature locations $\mathcal{C}_{k,i} \subset \mathbb{R}^{n_k}$ as in Eqn (10), define $\boldsymbol{\mathcal{C}}_k = \{\mathcal{C}_{k,i}\}_{i=1}^{M_k}$, and let $q_{nn(\mathcal{X}),k}$ be as in Eqn (19). Then, for $\hat{\mathbb{W}}_2$ defined in Eqn (20), it holds that

$$\mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X}),K+1}\right) \le \hat{\mathbb{W}}_2 \le \epsilon. \tag{23}$$

To prove Theorem 21, we use that Corollary 20 implies the layer-wise decomposition

$$\hat{\mathbb{W}}_2 = \sum_{k=1}^{K}\left(\prod_{i=k}^{K}\mathcal{L}_i\right)\Big[\mathbb{MW}_2\big(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\big) + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\big(G_{M_k}(q_{nn(\mathcal{X}),k})\big)\Big].$$

That is, $\hat{\mathbb{W}}_2$ equals the sum of compression and signature errors at layer k, scaled by the product of Lipschitz constants of the remaining layers $\prod_{i=k}^{K} \mathcal{L}_i$. If at each layer k the compression size M_k and grid sizes $N_{k,i,j}$ are such that

$$\mathbb{MW}_2\big(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\big) + \hat{\mathbb{W}}_{2,\mathbf{\Delta}_{\mathcal{C}_k}}\big(G_{M_k}(q_{nn(\mathcal{X}),k})\big) \leq \frac{\epsilon}{K\prod_{i=k}^{K}\mathcal{L}_i},$$

it follows that $\hat{\mathbb{W}}_2 \leq \epsilon$.

For any choice of M_k such that $\epsilon_k > 0$ (Eqn 21), Corollary 15 allows us to select grid sizes $N_{k,i,j}$ so that $\hat{\mathbb{W}}_{2,\mathbf{\Delta}_{\mathcal{C}_k}}\big(G_{M_k}(q_{nn(\mathcal{X}),k})\big) \leq \epsilon_k$, thereby satisfying the per-layer requirement and yielding $\hat{\mathbb{W}}_2 \leq \epsilon$. Applying no compression (i.e., setting M_k equal to the original number of mixture components in $q_{nn(\mathcal{X}),k}$) ensures $\epsilon_k > 0$ but can cause exponential growth in the total number of signature locations across depth, since signatures are taken component-wise. In practice, we therefore compress only redundant or low-weight components, i.e., retaining distinct modes, which preserves most of the signature budget while substantially reducing the number of grids (see Section 8.1).

Remark 22 *The number of signature locations required at layer k in Theorem 21 is essentially inversely proportional to the layer accuracy parameter ϵ_k, and hence to the target accuracy ϵ (up to the distribution of the budget across layers).*[12] *In the worst-case scenario where each of the n_k dimensions of a component requires the same grid size, the total number of signature locations for that component scales as*

$$\prod_{j=1}^{n_k} N_{k,i,j} = O\big(\epsilon_k^{-n_k}(\log(1/\epsilon_k))^{n_k/2}\big).$$

In practice, this curse of dimensionality is strongly mitigated because Condition (22) *weights each dimension by its eigenvalue. Empirically, most variance concentrates in a few neurons per layer, so the effective dimensionality is lower and many low-variance dimensions require only a single point ($N_{k,i,j} = 1$). Additionally, non-uniform and adaptive (e.g., variance-proportional) grids further reduce required sizes while preserving $\mathbb{W}_2$ accuracy (see Remark 12). In Section 8.1, we empirically analyze how this nominal exponential dependence is mitigated in practice.*

Example 2 *Consider the two-hidden layer ($K = 2$) SNN in Figure 2, where the Lipschitz constants (as defined in Eqn 17) of the last two layers are $\mathcal{L}_1 = 0.5$ and $\mathcal{L}_2 = 0.41$. Using Theorem 21, we derive the compression and grid sizes required to ensure that the approximation $q_{nn(x)}$ defined in Eqn* (15) *is within* 0.2 *in 2-Wasserstein distance ($\epsilon = 0.2$) of the true SNN output distribution $p_{nn(x)}$ at the input point considered in the figure.*

The distribution after the first linear layer, $q_{nn(x),1}$, is a univariate Gaussian with variance $\lambda_1 = 0.8$ (step a in the figure), so no compression is required and we set $M_1 = 1$. The maximum allowable signature approximation error at this layer is

$$\epsilon_1 = \frac{\epsilon}{K\prod_{i=1}^{2}\mathcal{L}_i} = \frac{0.2}{2\cdot 0.5\cdot 0.41} = 0.49.$$

Constructing the signature locations $\mathcal{C}_1$ uniformly according to Eqn (12) *yields*

$$\mathbb{W}(\Delta_{\mathcal{C}}\#q_{nn(x),1}, q_{nn(x),1}) \leq 0.49,$$

12. At layer k, the signature contribution behaves like $O\left(\frac{\log N_{k,i,j}}{N_{k,i,j}^2}\right)$ for large $N_{k,i,j}$. Solving $\log N_{k,i,j}/N_{k,i,j}^2 \approx \epsilon_k^2$ gives $N_{k,i,j} = O\big(\epsilon_k^{-1}\sqrt{\log(1/\epsilon_k)}\big)$: a $1/\epsilon_k$ rate with only a mild $\sqrt{\log}$ factor.

that is, Eqn (22) *holds for* $N_1 = 5$ *and* $\epsilon_1 = 0.49$. *Using instead the optimal grid computed via Algorithm 2, we obtain an error of* 0.23 *with the same number of locations. Using these optimal locations, we propagate the signature of* $q_{nn(x),1}$ *through the activation and the second linear layer and obtain* $q_{nn(x),2}$ *as in Eqn* (15c) *(steps b-d). Next, we compress the resulting 5-component mixture* $q_{nn(x),2}$ *to a mixture with* $M_2 = 3$ *components, which yields a compression error*

$$\mathbb{MW}_2\left(q_{nn(x),2}, G_{M_2}(q_{nn(x),2})\right) = 0.02.$$

This leaves a remaining signature approximation budget of

$$\epsilon_2 = \tfrac{0.2}{2\cdot 0.41} - 0.02 = 0.22.$$

The compressed mixture $G_{M_2}(q_{nn(x),2})$ *has three components with weights* $\pi = (0.2, 0.6, 0.2)$ *and variances* $(0.5, 0.2, 0.5)$. *Eqn* (22) *is therefore satisfied with grid sizes* $N_{2,1} = 3$, $N_{2,2} = 4$, *and* $N_{2,3} = 3$. *Taking the signature of* $G_{M_2}(q_{nn(x),2})$ *and passing it through the activation and final linear layer, we obtain an approximation* $q_{nn(x)}$ *with* $M = 10$ *components achieving* $\epsilon = 0.2$. *Using Algorithm 2 again to construct the signature locations, we instead guarantee* $\epsilon_2 \leq 0.22$ *with only* $M = 5$ *and obtain a final approximation of* 5 *components (steps f-h), achieving* $\epsilon = 0.12$.

Algorithm 1: Gaussian Mixture Approximation of a SNN

input : $p_{nn(\mathcal{X})}$
output: $q_{nn(\mathcal{X})}$ and $\hat{\mathbb{W}}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right)$
begin
Initialize $q_{nn(\mathcal{X}),1}$ as in Eqn (19a) and $\hat{\mathbb{W}}_{2,1}$ as in Eqn (16a)
for $k \in \{1, \ldots, K\}$ **do**
Construct $G_{M_k}(q_{nn(\mathcal{X}),k})$ using compression Algorithm 3
Compute $\mathbb{MW}_2\left(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\right)$ as in Eqn (2)
Construct signature locations $\boldsymbol{\mathcal{C}}_k$ and the signature $d_{nn(\mathcal{X}),k}$ of $G_{M_k}(q_{nn(\mathcal{X}),k})$, and compute the signature error $\hat{\mathbb{W}}_{2,\Delta_{\boldsymbol{\mathcal{C}}_k}}(G_{M_k}(q_{nn(\mathcal{X}),k}))$ all via Algorithm 2
Update $q_{nn(\mathcal{X}),k+1}$ as in Eqn (19c)
Compute $\mathbb{E}_{w_k \sim p_{w_k}}[\mathcal{L}^2_{L^{w_k}}]^{\frac{1}{2}}$ as described in Appendix C
Update $\hat{\mathbb{W}}_{2,k+1}$ as in Eqn (16b)
Return $q_{nn(\mathcal{X})} = q_{nn(\mathcal{X}),K+1}$ and $\hat{\mathbb{W}}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right) = \hat{\mathbb{W}}_{2,K+1}$

7. Algorithmic Framework

In this section, the theoretical methodology to solve Problem 1 developed in the previous sections is translated into an algorithmic framework. First, following Eqn (19), we present an algorithm to construct the Gaussian mixture approximation of a SNN with error bounds in the 2-Wasserstein distance, including details on the compression step. Then, we rely on the fact that the error bound resulting from our approach is piecewise differentiable with respect to the parameters of the SNN to address the complementary problem highlighted in Remark 4. This problem involves deriving an algorithm to optimize the parameters of a SNN such that the SNN approximates a given GMM.

Algorithm 2: Signatures of Gaussian Mixtures

input : $q = \sum_{i=1}^{M} \tilde{\pi}^{(i)} \mathcal{N}(m_i, \Sigma_i)$
output: $d = \sum_{j=1}^{N} \pi^{(j)} \delta_{c_j}$ and $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}(q)$
begin
  Construct the signature of the components of the mixture using the (fixed and optimal) signature of $\mathcal{N}(0,1)$ with locations $\mathcal{C}_{1d}$:
  for $i \in \{1, \ldots, M\}$ **do**
1 Compute eigenvalues vector λ_i and eigenvector matrix V_i of Σ_i
2 Find the optimal grid sizes $\{N_1^*, \ldots, N_n^*\}$ for λ_i according to Eqn (24)
3 Construct grid of points $\mathcal{C}^* = \mathcal{C}^*(N_1) \times \ldots \times \mathcal{C}^*(N_n)$, with $\mathcal{C}^*(N_i)$ as in Eqn (25)
4 Transform the grid to the original space: $\mathcal{C}_i = \text{Post}(\mathcal{C}^*, S) + m_i$ where $S = V_i \text{diag}(\lambda_i)^{\frac{1}{2}}$
5 Set $\hat{\mathbb{W}}^2_{2,\Delta_{\mathcal{C}_i}} = \sum_{l=1}^{n} \lambda_i^{(l)} \sum_{j=1}^{N_l} \pi_l^{(j)} \left[\nu_j^{(l)} + (\mu_j^{(l)} - \tilde{c}_j^{(l)})^2\right]$, with $\mathcal{C}^*(N_l) = \{c_j^{(l)}\}_{j=1}^{N_l}$, and $\pi_l^{(j)}$, $\mu_j^{(l)}$, and $\nu_j^{(l)}$ as in Eqn (9)
6 Define $d_i = \sum_{j=1}^{|\mathcal{C}_i|} \pi^{(j)} c_j$ for $\pi^{(j)} = \prod_{l=1}^{n} \pi_l^{(j)}$
7 Return $d = \sum_{i=1}^{M} \tilde{\pi}^{(i)} d_i$ and $\hat{\mathbb{W}}^2_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}(q) = \sum_{i=1}^{M} \tilde{\pi}^{(i)} \hat{\mathbb{W}}^2_{2,\Delta_{\mathcal{C}_i}}$

Algorithm 3: Compress Gaussian Mixture

input : $\tilde{q} = \sum_{i=1}^{\tilde{M}} \tilde{\pi}^{(i)} \mathcal{N}(\tilde{m}_i, \tilde{\Sigma}_i)$
output: $q = \sum_{j=1}^{M} \pi^{(j)} \mathcal{N}(m_j, \Sigma_j)$
begin
1 Collect the components of $\tilde{q}$ in M clusters by applying M-means clustering to $\{\tilde{m}_1, \ldots, \tilde{m}_{\tilde{M}}\}$ using Lloyd's Algorithm
  for $j \in \{1, \ldots, M\}$ **do**
    Compress all components in the j-th cluster with indices in $\mathcal{I}_j$ to component $\mathcal{N}(m_j, \Sigma_j)$ with weight $\pi^{(j)}$ by taking the weighted average of the means and covariances:
2    $\pi^{(j)} = \sum_{i \in \mathcal{I}_j} \tilde{\pi}^{(i)}$; $m_j = \frac{1}{\pi^{(j)}} \sum_{i \in \mathcal{I}_j} \tilde{\pi}^{(i)} \tilde{m}_i$;
    $\Sigma_j = \frac{1}{\pi^{(j)}} \sum_{i \in \mathcal{I}_j} \tilde{\pi}^{(i)} \left[\tilde{\Sigma}_i + (\tilde{m}_i - m_j)(\tilde{m}_i - m_j)^T\right]$
3 Return $q = \sum_{j=1}^{M} \pi^{(j)} \mathcal{N}(m_j, \Sigma_j)$

7.1 Construction of the GMM Approximation of a SNN

The procedure to build a Gaussian mixture approximation $q_{nn(\mathcal{X})}$ of $p_{nn(\mathcal{X})}$, i.e., the output distribution of a SNN at a set of input points $\mathcal{X}$ as in Eqn (19), is summarized in Algorithm 1. The algorithm consists of a forward pass over the layers of the SNN. First, the Gaussian output distribution after the first affine layer is constructed (line 1). Then, for each layer $k > 0$, $q_{nn(\mathcal{X}),k}$ is compressed to a mixture of size M (line 2); the signature of $G_{M_k}(q_{nn(\mathcal{X})})$ is computed (line 4); and the signature is passed through the activation and affine layer to obtain $q_{nn(\mathcal{X}),k+1}$ (line 5). Parallel to this, error bounds on the 2-Wasserstein distance are computed for the compression operation (line 3), signature operation (line 4), and affine operation (line 6), which are then composed into a bound on $\mathbb{W}_2\left(p_{nn(\mathcal{X}),k+1}, q_{nn(\mathcal{X}),k+1}\right)$ that is propagated to the next layer (line 7).

Signature Operation The signature operation on $q_{nn(\mathcal{X}),k}$ in line 4 of Algorithm 1 is performed according to the steps in Algorithm 2. That is, the signature of $q_{nn(\mathcal{X}),k}$ is taken as the mixture of the signatures of the components of $q_{nn(\mathcal{X}),k}$ (lines 1-6). For each Gaussian component of the mixture, to ensure an analytically exact solution for the Wasserstein distance resulting from the signature, we position its signature locations on a grid that aligns with the covariance matrix's basis vectors, i.e., the locations satisfy the constraint as in Eqn (10) of Corollary 11. In particular, the grid of signatures is constructed so that the Wasserstein distance from the signature operation is minimized. According to the following proposition, the problem of finding this optimal grid can be solved in the transformed space, in which the Gaussian is independent over its dimensions.

Proposition 23 *For $\mathcal{N}(m,\Sigma) \in \mathcal{P}(\mathbb{R}^n)$, let matrix $T \in \mathbb{R}^{n\times n}$ be such that $T = \operatorname{diag}(\lambda)^{-\frac{1}{2}}V^T$ where $\operatorname{diag}(\lambda) = V^T\Sigma V$ is a diagonal matrix whose entries are the eigenvalues of Σ, and V is the corresponding orthogonal (eigenvector) matrix. Then,*

$$\underset{\substack{\{\mathcal{C}^1,\dots,\mathcal{C}^n\}\subset\mathbb{R},\\ \sum_{i=1}^n|\mathcal{C}^i|=N}}{\operatorname{argmin}} \mathbb{W}_2\left(\Delta_{\mathrm{Post}(\mathcal{C}^1\times\dots\times\mathcal{C}^n,T^{-1})+\{m\}}\#\mathcal{N}(m,\Sigma),\mathcal{N}(m,\Sigma)\right) = \{\mathcal{C}^*(N_1^*),\dots,\mathcal{C}^*(N_n^*)\},$$

where

$$\mathcal{C}^*(N) = \underset{\mathcal{C}\in\mathbb{R},|\mathcal{C}|=N}{\operatorname{argmin}} \mathbb{W}_2\left(\Delta_{\mathcal{C}}\#\mathcal{N}(0,1),\mathcal{N}(0,1)\right), \tag{24}$$

$$\{N_1^*,\dots,N_n^*\} = \underset{\substack{\{N_1,\dots,N_n\}\in\mathbb{N}^n,\\ \prod_{j=1}^n,N_j=N}}{\operatorname{argmin}} \sum_{j=1}^n \lambda^{(j)}\mathbb{W}_2\left(\Delta_{\mathcal{C}^*(N_i)}\#\mathcal{N}(0,1),\mathcal{N}(0,1)\right). \tag{25}$$

Following Proposition 23, the optimal grid of signature locations of any Gaussian is defined by the optimal signature locations for a univariate Gaussian (Eqn 24) and the optimal size of each dimension of the grid (Eqn 25). While the optimization problem in Eqn (24) is non-convex, it can be solved up to any precision using the fixed-point approach proposed in Kieffer (1982). In particular, we solve the problem once for a range of grid sizes N and store the optimal locations in a lookup table that is called at runtime. To address the optimization problem in Eqn (25), we use that $\mathbb{W}_2\left(\Delta_{\mathcal{C}^*(N)}\#\mathcal{N}(0,1),\mathcal{N}(0,1)\right)$, with optimal locations $\mathcal{C}^*(N)$ as in Eqn (24), is strictly decreasing for increasing N so that even for large N, the number of feasible non-dominated candidates $\{N_1,\dots,N_n\}$ is small.

Remark 24 *Algorithm 2 constructs a set of grid-constrained signatures that satisfy Eqn* (10)*, thereby enabling formal error evaluation via Corollary 11. In settings where formal error bounds are not required, more flexible choices for the signature locations can alternatively be employed, provided they satisfy the conditions in Remark 7 that ensure tractability of the signature operations. One such alternative signature placement strategy is presented in Appendix D. In Section 8.1, we empirically compare the strategy in Algorithm 2 with the strategy in Appendix D by estimating the Wasserstein distance using Monte Carlo methods.*

Compression Operation Our approach to compress a GMM of size N into a GMM of size $M < N$, i.e., operation G_{M_k} in Eqn (15b), is described in Algorithm 3 and is based on the M-means clustering using Lloyd's algorithm (Lloyd, 1982) on the means of the mixture's components (line 1). That is, each cluster is substituted by a Gaussian distribution with mean and covariance equal to those of the mixture in the cluster (line 2).

Remark 25 *For the compression of discrete distributions, such as in the case of Bayesian inference via dropout, we present a computationally more efficient alternative procedure to Algorithm 3 in Section E of the Appendix. Note that in the dropout case, the support of the multivariate Bernoulli distribution representing the dropout masks at each layer's input grows exponentially with the layer width, making compression of paramount importance in practice.*

7.2 Construction of a SNN Approximation of a GMM

The algorithmic framework presented in the previous subsection finds the Gaussian Mixture distribution that best approximates the output distribution of a SNN and returns bounds on their distance. In this subsection, we show how our results can be employed to solve the complementary problem of finding the parameter distribution of a SNN to best match a given GMM.

Let us consider a finite set of input points $\mathcal{X} \subset \mathbb{R}^{n_0}$ and a SNN and GMM, whose distributions at $\mathcal{X}$ are respectively $p_{nn(\mathcal{X})}$ and $q_{gmm(\mathcal{X})}$. Furthermore, in this subsection, we assume that the parameter distribution p_w of the SNN is parametrized by a set of parameters ψ. For instance, in the case where p_w is a multivariate Gaussian distribution, then ψ parametrizes its mean and covariance. Note that, consequently, also $p_{nn(\mathcal{X})}$ depends on ψ through Eqn (18). To make the dependence explicit, in what follows, we will use a superscript to emphasize the dependency on ψ. Then, our goal is to find $\psi^* = \operatorname{argmin}_\psi \mathbb{W}_2(p^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})})$. To do so, we rely on Corollary 26 below, which uses the triangle inequality to extend Corollary 20.

Corollary 26 (of Corollary 20) *Let $\mathcal{X} \subset \mathbb{R}^{n_0}$ be a finite set of input points. Then, for a SNN and a GMM, whose distributions at $\mathcal{X}$ are respectively $p^\psi_{nn(\mathcal{X})}$ and $q_{gmm(\mathcal{X})}$, it holds that*

$$\mathbb{W}_2\big(p^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big) \leq \hat{\mathbb{W}}_2\big(p^\psi_{nn(\mathcal{X})}, q^\psi_{nn(\mathcal{X})}\big) + \mathbb{MW}_2\big(q^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big),$$

where the GMM $q^\psi_{nn(\mathcal{X})}$ and the bound $\hat{\mathbb{W}}_2$ are obtained via Algorithm 1, and $\mathbb{MW}_2$ is as defined in Definition 2, with $q^\psi_{nn(\mathcal{X})}$ also depending on ψ through Eqn (19).

Using, Corollary 26, we can approximate ψ^* as follows

$$\psi^* \approx \operatorname*{argmin}_\psi \big\{\beta \hat{\mathbb{W}}_2\big(p^\psi_{nn(\mathcal{X})}, q^\psi_{nn(\mathcal{X})}\big) + (1-\beta)\mathbb{MW}_2\big(q^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big)\big\}, \tag{26}$$

where $\beta \in [0,1]$ allows us to trade off between the gap between the bound $\hat{\mathbb{W}}_2\big(p^\psi_{nn(\mathcal{X})}, q^\psi_{nn(\mathcal{X})}\big)$ and $\mathbb{W}_2\big(p^\psi_{nn(\mathcal{X})}, q^\psi_{nn(\mathcal{X})}\big)$ and the gap between the bound $\mathbb{MW}_2\big(q^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big)$ and $\mathbb{W}_2\big(q^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big)$. For instance, in the case where $q^\psi_{nn(\mathcal{X})}$ and $q_{gmm(\mathcal{X})}$ are Gaussian distributions, $\mathbb{MW}_2\big(q^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big)$ equals $\mathbb{W}_2\big(q^\psi_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\big)$, leading us to choose $\beta < \frac{1}{2}$. As discussed next, under the assumption that the mean and covariance of p_w are differentiable with respect to ψ, the objective in Eqn (26) is piecewise differentiable with respect to ψ. Hence, the optimization problem can be approximately solved using gradient-based optimization techniques, such as Adam (Kingma and Ba, 2014).

Let us first consider the term $\hat{\mathbb{W}}_2$ in the objective, which is iteratively defined in Eqn (16) over the layers of the network using the quantities $\mathbb{E}_{w_k \sim p_{w_k}}[\mathcal{L}^2_{L^{w_k}}]^{1/2}$, $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\big(G_{M_k}(q_{nn(\mathcal{X}),k})\big)$, and $\mathbb{MW}_2\big(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\big)$ for each layer k. In Algorithm 1, we compute these quantities so that the gradients w.r.t. the mean and covariance of p_w are (approximately) analytically tractable:

1. $\mathbb{E}_{w_k \sim p_{w_k}}[\mathcal{L}^2_{L^{w_k}}]^{1/2}$ is computed as described in Appendix C (line 6 of Algorithm 1) as a function of the sum of the variances of the k-th layer's parameters and the spectral norm of the mean

of the k-th layer's weight matrix. Hence, the gradient with respect to the variance is well defined, and the gradient with respect to the mean exists if the largest singular value is unique; otherwise, we can use a sub-gradient derived from the singular value decomposition (SVD) of the mean matrix (Rockafellar, 2015).

2. $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(\mathcal{X}),k})\right)$ is obtained according to Algorithm 2 (line 4 of Algorithm 1) based on the eigendecomposition of the covariance matrix for each component of the GMM $G_{M_k}(q_{nn(\mathcal{X}),k})$ (line 1 of Algorithm 2). Differentiability of $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(\mathcal{X}),k})\right)$ with respect to the mean and covariance of p_w follows from the piecewise differentiability of the eigenvalue decomposition of a (semi-)positive definite matrix (Magnus and Neudecker, 2019),[13] provided that the covariances of $G_{M_k}(q_{nn(\mathcal{X}),k})$ are differentiable with respect to the mean and covariance of p_w. To establish the latter, note that the compression operation G_{M_k}, as performed according to Algorithm 3, given the clustering of the components of the mixture $q_{nn(\mathcal{X}),k}$, reduces to computing the weighted sum of the means and covariances of each cluster. Hence, the covariances of $G_{M_k}(q_{nn(\mathcal{X}),k})$ are piecewise differentiable with respect to the means and covariances of $q_{nn(\mathcal{X}),k}$. Here, $q_{nn(\mathcal{X}),k}$ is an affine combination of $p_{w_{k-1}}$ and $\mathcal{X}$ if $k = 1$, and the support of the signature approximation of $G_{M_k}(q_{nn(\mathcal{X}),k-1})$ otherwise (see Example 1 for the closed-form expression of $q_{nn(\mathcal{X}),k}$ in the case $\mathcal{X} = \{x\}$). As such, from the chain rule, it follows that for all k the means and covariances of $G_{M_k}(q_{nn(\mathcal{X}),k})$, and consequently, $\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(\mathcal{X}),k})\right)$, are piecewise differentiable with respect to the means and covariances of $p_{w_{k-1}}, \ldots, p_{w_0}$.

3. $\mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)$ is taken as the $\mathbb{MW}_2$ distance between $q_{nn(\mathcal{X}),k}$ and $G_{M_k}(q_{nn(\mathcal{X}),k})$ (line 3 in Algorithm 1). Given the solution of the discrete optimal transport problem in Eqn (2), the $\mathbb{MW}_2$ distance between two Gaussian mixtures reduces to the sum of 2-Wasserstein distances between Gaussian components, which have closed-form expressions (Givens and Shortt, 1984) that are differentiable with respect to the means and covariances of the components. In point 2, we showed that the means and covariances of $q_{nn(\mathcal{X}),k}$ and $G_{M_k}(q_{nn(\mathcal{X}),k})$ are piecewise differentiable w.r.t the mean and covariance of p_w; therefore, $\mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)$ is piecewise differentiable with respect to p_w.

From the piecewise differentiability of all three quantities with respect to the mean and covariance of p_w for all layers k, it naturally follows that $\hat{\mathbb{W}}_2\big(p^{\psi}_{nn(\mathcal{X})}, q^{\psi}_{nn(\mathcal{X})}\big)$ is piecewise differentiable with respect to the mean and covariance of p_w using simple chain rules. For the second term of the objective, $\mathbb{MW}_2\big(q^{\psi}_{nn(\mathcal{X})}, q_{gp(\mathcal{X})}\big)$, we can apply the same reasoning from point 3 to conclude that it is piecewise differentiable with respect to the means and covariances of the components of the GMM $q^{\psi}_{nn(\mathcal{X})}$, which, according to point 2, are piecewise differentiable with respect to the mean and covariance of p_w. Therefore, we can conclude that the objective in Eqn (26) is piecewise differentiable with respect to the mean and covariances of the parameter distribution of the SNN. Note that, in practice, the gradients can be computed using automatic differentiation, as demonstrated in Subsection 8.3 where we encode informative priors for SNNs via Gaussian processes by solving the optimization problem in Eqn (26).

Remark 27 *While straightforward automatic differentiation shows to handle the discontinuity in the gradients well in practice, as analyzed in Subsection 8.3, more advanced techniques for non-smooth optimization can be employed to solve Eqn* (26) *(Mäkelä, 2002; Burke et al., 2020).*

13. Note that there are discontinuous changes in the eigenvalues for small perturbations if the covariance matrix is (close to) degenerate, which occurs in our cases if the behavior of the SNN is strongly correlated for points in $\mathcal{X}$. To prevent this, we use a stable derivation implementation for degenerate covariance matrices as provided by Kasim (2020) to compute the eigenvalue decomposition.

8. Experimental Results

In this section we experimentally evaluate the performance of our framework in solving Problem 1 and then demonstrate its practical usefulness in two applications: uncertainty quantification and prior tuning for SNNs. We will start with Section 8.1, where we consider various trained SNNs and analyze the precision of the GMM approximation obtained with our framework. Section 8.2 focuses on analyzing the uncertainty of the predictive distribution of SNNs trained on classification data sets utilizing our GMM approximations. Finally, in Section 8.3, we show how our method can be applied to encode functional information in the priors of SNNs.[14]

Data Sets We study SNNs trained on various regression and classification data sets. For regression tasks, we consider the NoisySines data set, which contains samples from a 1D mixture of sines with additive noise as illustrated in Figure 1, and the Kin8nm, Energy, Boston Housing, and Concrete UCI data sets, which are commonly used as regression tasks to benchmark SNNs (Hernández-Lobato and Adams, 2015). For classification tasks, we consider the MNIST, Fashion-MNIST, and CIFAR-10 data sets.

Networks & Training We consider networks composed of fully-connected and convolutional layers, among which the VGG style architecture (Simonyan and Zisserman, 2014). We denote by $[n_l \times n_n]$ an architecture with n_l fully-connected layers, each with n_n neurons. The composition of different layer types is denoted using the summation sign. For each SNN, we report the inference techniques used to train its stochastic layers in brackets, while deterministic layers are noted without brackets. For instance, VGG-2 + [1x128] (VI) represents the VGG-2 network with deterministic parameters, stacked with a fully-connected stochastic network with one hidden layer of 128 neurons learned using Variational Inference (VI). For regression tasks, the networks use tanh activation functions, whereas ReLU activations are employed for classification tasks. For VI inference, we use VOGN (Khan et al., 2018), and for deterministic and dropout networks, we use Adam.

Metrics for Error Analysis To quantify the precision of our approach for the different settings, we report the Wasserstein distance between the approximate distribution $q_{nn(\mathcal{X})}$ and the true SNN distribution $p_{nn(\mathcal{X})}$ relative to the 2-nd moment of the approximate distribution. That is, we report

$$\overline{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right) \coloneqq \frac{\mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right)}{\mathbb{E}_{z\sim q_{nn(\mathcal{X})}}[\|z\|^2]^{1/2}}, \tag{27}$$

which we refer to as the relative 2-Wasserstein distance. We use $\overline{W}_2$ because, as guaranteed by Lemma 28 in the appendix, it provides an upper bound on the relative difference between the second moment of $p_{nn(\mathcal{X})}$ and $q_{nn(\mathcal{X})}$, i.e.,

$$\frac{\left|\mathbb{E}_{z\sim p_{nn(\mathcal{X})}}[\|z\|^2]^{1/2} - \mathbb{E}_{\tilde{z}\sim q_{nn(\mathcal{X})}}[\|\tilde{z}\|^2]^{1/2}\right|}{\mathbb{E}_{\tilde{z}\sim q_{nn(\mathcal{X})}}[\|\tilde{z}\|^2]^{1/2}} \leq \overline{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right).$$

Note that in our framework, $q_{nn(\mathcal{X})}$ is a GMM, hence $\mathbb{E}_{z\sim q_{nn(\mathcal{X})}}[\|z\|^2]$ can be computed analytically.[15] In our experiments, we report both a formal bound of $\overline{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right)$ obtained using Theorem 17 and an empirical approximation. To compute the empirical approximation of $\overline{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right)$, we use $1e3$ Monte Carlo (MC) samples from $p_{nn(\mathcal{X})}$ and $q_{nn(\mathcal{X})}$ and solve the resulting discrete optimal transport problem to approximate $\mathbb{W}_2\left(p_{nn(x)}, q_{nn(x)}\right)$.

14. Our code is available at `https://github.com/sjladams/experiments-snns-as-mgps`.

15. We have that $\mathbb{E}_{z\sim q_{nn(\mathcal{X})}}[\|z\|^2] = \|m\|^2 + \text{trace}(\Sigma)$, where m and Σ are the mean vector and covariance matrix of $q_{nn(\mathcal{X})}$, respectively. Closed-forms for m and Σ exist for $q_{nn(\mathcal{X})} \in \mathcal{G}_{<\infty}(\mathbb{R}^n)$.

		Grid		Cross
Data Set	Network Architecture	Emp.	Formal	Emp.
NoisySines	[1x64] (VI)	0.00109	0.33105	0.00028
	[1x128] (VI)	0.00130	0.13112	0.00041
	[2x64] (VI)	0.00355	1.63519	0.00305
	[2x128] (VI)	0.01135	2.48488	0.00956
Kin8nm	[1x128] (VI)	0.00012	0.04243	0.00012
	[2x128] (VI)	0.00018	0.46303	0.00005
MNIST	[1x128] (VI)	0.00058	0.05263	0.00037
	[2x128] (VI)	0.00127	0.44873	0.00124
Fashion MNIST	[1x128] (VI)	0.00017	0.06984	0.00013
	[2x128] (VI)	0.02787	1.45466	0.02228
CIFAR-10	VGG-2 + [1x128] (VI)	0.00617	0.57670	0.00363
	VGG-2 + [2x128] (VI)	0.04901	2.35686	0.02890
MNIST	[1x64] (Drop)	0.27501	0.94371	0.18400
	[2x64] (Drop)	0.41874	1.37594	0.08603
CIFAR-10	VGG-2 (Drop) + [1x32] (VI)	0.27492	179.9631	0.18996

Table 2: The empirical estimates and formal bounds on the relative 2-Wasserstein distance between various SNNs and their Gaussian Mixture approximates obtained via Algorithm 1 using signature approximations of size 10 and a compression size (M_k) of 5 for all layers. The Grid column reports the error for approximates constructed using Algorithm 2. The Cross column reports the empirical errors for approximates constructed using the alternative cross-shaped signatures via Algorithm 4, for which formal bounds are intractable for this method. For the dropout networks, instead of Algorithm 3, we use the compression procedure as described in Appendix E. The reported values are the average over 100 randomly selected points from the test sets.

Compression Size:		1	3	No Compression
NoisySines	[2x64] (VI)	1.97279	1.63519	1.08763
	[2x128] (VI)	2.48109	2.48488	2.48398
Kin8nm	[2x128] (VI)	0.48740	0.46303	0.45387
MNIST	[2x128] (VI)	0.45036	0.44873	0.44873
Fashion MNIST	[2x128] (VI)	1.45489	1.45466	1.45466
CIFAR-10	VGG-2 + [2x128] (VI)	2.36616	2.35686	2.35686
Compression Size:		10	100	10000
MNIST	[1x64] (Drop)	0.94371	0.73682	0.3571
	[2x64] (Drop)	1.37594	1.09053	0.89742
CIFAR-10	VGG-2 (Drop) + [1x32] (VI)	179.9631	180.0963	145.2984

Table 3: The formal bounds on the relative 2-Wasserstein distance between various SNNs and their Gaussian Mixture approximations obtained via Algorithm 1 for a signature size of 100 and different compression sizes (M_k). The values are the average of 100 random test points.

8.1 Gaussian Mixture Approximations of SNNs

In this subsection, we experimentally evaluate the effectiveness of Algorithm 1 in constructing a GMM that is ϵ-close to the output distribution of a given SNN. We first demonstrate that, perhaps surprisingly, even a small number of discretization points for the signature operation and a large reduction in the compression operation (i.e., M_k relatively small), often suffice for Algorithm 1 to generate GMMs that accurately approximate SNNs both empirically and formally. Then, we provide empirical evidence that by increasing the signature and compression sizes, the approximation error of the GMM can be made arbitrarily small and the formal error upper bound resulting from Theorem 17 converges to 0.

Baseline Performance We start our analysis with Table 2, where we assess the performance of Algorithm 1 in generating GMM approximations of SNN trained using both VI and dropout. The results show that for the SNNs trained with VI, even with only 10 signature points per-component, Algorithm 1 is able to generate GMMs, whose empirical distance from the true BNN is always smaller than 10^{-2}. This can be partly explained as for VI generally only a few neurons are active (Louizos et al., 2017; Frankle and Carbin, 2018), so it is necessary to focus mostly on these on the signature approximation to obtain accurate GMM approximations at the first K-th layers (See Remark 22). Moreover, the propagation of the GMM approximation through the last fully-connected layer is performed in closed form (i.e., by Eqn 15c for $\mathcal{X} = \{x\}$) and this mitigates the influence of the approximation error of the previous layers on the final approximation error. In contrast, for the dropout networks, the empirical approximation error is, on average, one order of magnitude larger. This can be explained because the input-output distribution of dropout networks is inherently highly multimodal. Thus, as we will show in the next paragraph, to obtain tighter GMM approximations, it is necessary to use $M > 5$ (the value used in Table 2) and allow for a larger size in the GMM approximation.

Note that the formal error bounds tend to become more conservative with increasing network depth. This is related to the transformations in the (stochastic) fully-connected layers, for which the (expected) norm of the weight matrices is used to bound its impact on the Wasserstein guarantees (see the affine operator bound in Table 1). It is also interesting to note how the error bounds tend to be more accurate, and consequently closer to the empirical estimates, for the dropout networks. This is because for VI it is necessary to bound the expected matrix norm of each affine operation, which has to be upper bounded (see Appendix C), thereby introducing conservatism. Fortunately, as we will illustrate in the following paragraph and as we have mathematically proven in the previous sections, in all cases the error bounds can be made arbitrarily small by increasing signature size and M.

Lastly, in settings where formal error bounds are not required, alternative signature-placement methods such as Algorithm 4 in the appendix, which places the signature locations in a cross along the principal axes of the components' covariance matrix, provide a practical alternative. In the cases considered, this cross-shaped placement consistently attains higher empirical accuracy with the same number of signature points and lower runtime, so fewer points often suffice, however, it does not yield formal guarantees. The compute cost of Algorithm 1 is modest, with processing a single test point typically taking on the order of tenths of a second.[16]

Uniform Convergence We continue our analysis with Figure 6, where we conduct an extensive analysis of how the formal error bound from Theorem17 changes with increasing signature size for various SNN architectures trained on both regression and classification data sets. The plot confirms that, with increasing signature sizes, the approximation error, measured as $\overline{W}_2$, tends to decrease

16. All experiments were run on an Intel Core i7-10610U (1.80–2.30 GHz) with 16 GB RAM.

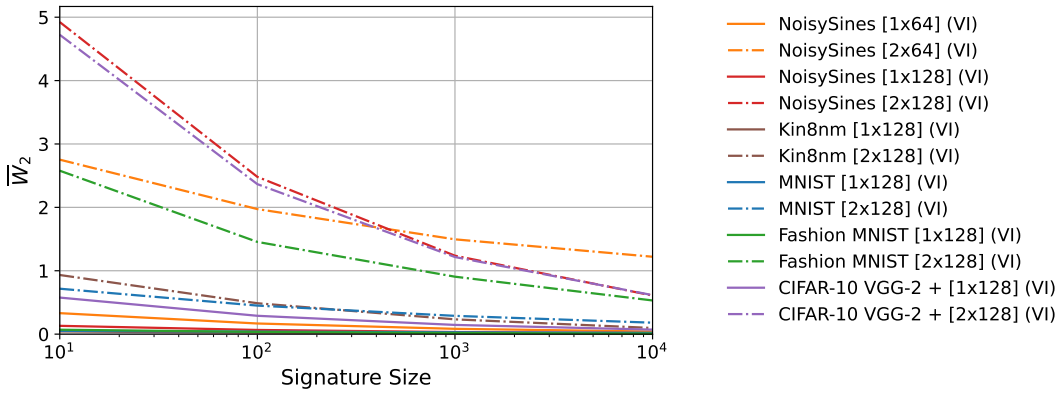


Figure 6: Formal bounds on the relative 2-Wasserstein distance between various SNNs and their Gaussian Mixture approximations obtained via Algorithm 1, for a compression size (M_k) of 3 and different signature sizes. The lines show the average of 100 random test points.

uniformly. In particular, we observe an inverse-linear convergence rate. This behaviour matches our theoretical result in Theorem 21. That theorem implies that a per-dimension grid size of order $N = O(\epsilon^{-1}\sqrt{\log(1/\epsilon)})$ is sufficient to reach a target precision ϵ.[17]

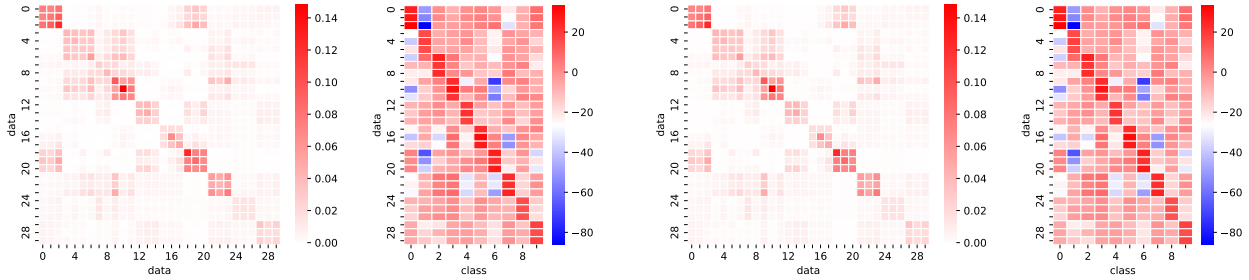
In Table 3, we investigate the other approximation step: the compression operation, which reduces the components of the approximation mixtures (step e in Figure 2). The results show that while the networks trained solely with VI can be well approximated with single Gaussian distributions, the precision of the approximation for the networks employing dropout strongly improves with increasing compression size. This confirms that the distribution of these networks is highly multi-modal and underscores the importance of allowing for multi-modal approximations, such as Gaussian mixtures.

8.2 Posterior Performance Analysis of SNNs

The GMM approximations obtained by Algorithm 1 naturally allow us to visualize the mean and covariance matrix (i.e., kernel) of the predictive distribution at a set of points. The resulting kernel offers the ability to reason about our confidence on the predictions and enhance our understanding of SNN performances (Khan et al., 2019). The state-of-the-art for estimating such a kernel is to rely on Monte Carlo sampling (Van Amersfoort et al., 2020), which, however, is generally time-consuming due to the need to consider a large amount of samples for obtaining accurate estimates.

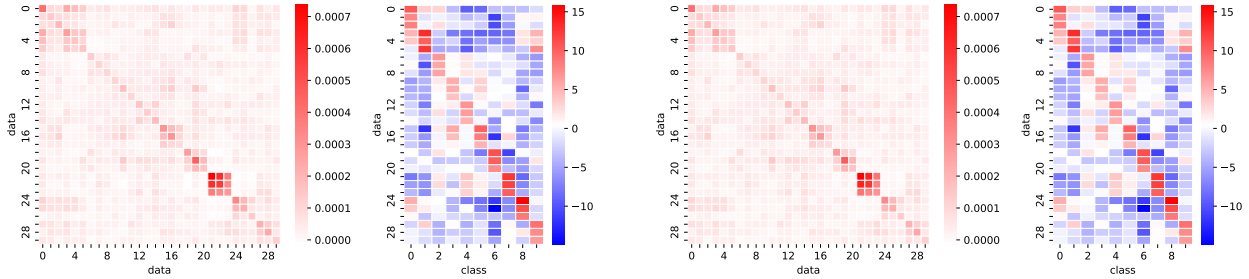
In Figure 7, we compare the mean and covariance of various SNNs (obtained via Monte Carlo sampling using $1e4$ samples) and the relative GMM approximation on a set of 30 randomly collected input points for various neural network architectures on the MNIST and CIFAR-10 data sets. We observe that the GMMs match the MC approximation of the true mean and kernel. However, unlike the GMM approximations, the MC approximations lack formal guarantees of correctness, and additionally, computing the MC approximations is generally two orders of magnitudes slower than computing the GMM approximations. By analyzing Figure 7, it is possible to observe that since the architectures allow for the training of highly accurate networks, each row in the posterior mean reflects that the classes are correctly classified. For the networks trained using VI, the kernel

17. In Appendix F, we experimentally investigate in more detail the uniform convergence of the 2-Wasserstein distance resulting from the signature operation for Gaussian mixtures of different sizes.



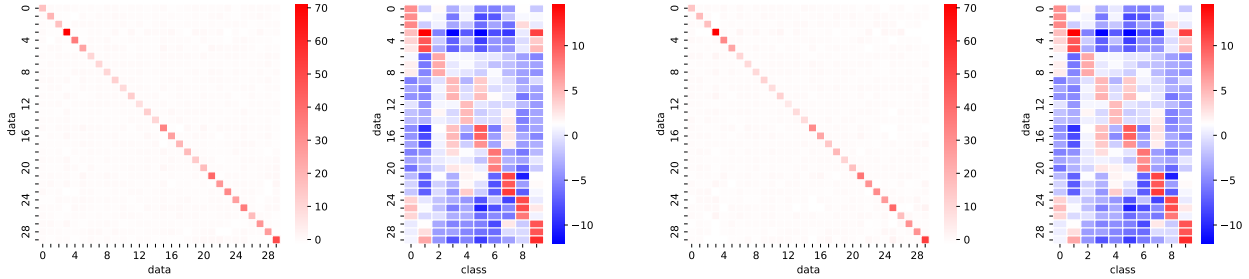
(a) Kernel SNN (b) Mean SNN (c) Kernel GMM (d) Mean GMM

(I) [2x128] (VI) trained on MNIST



(a) Kernel SNN (b) Mean SNN (c) Kernel GMM (d) Mean GMM

(II) VGG-2 + [1x128] (VI) trained on CIFAR-10

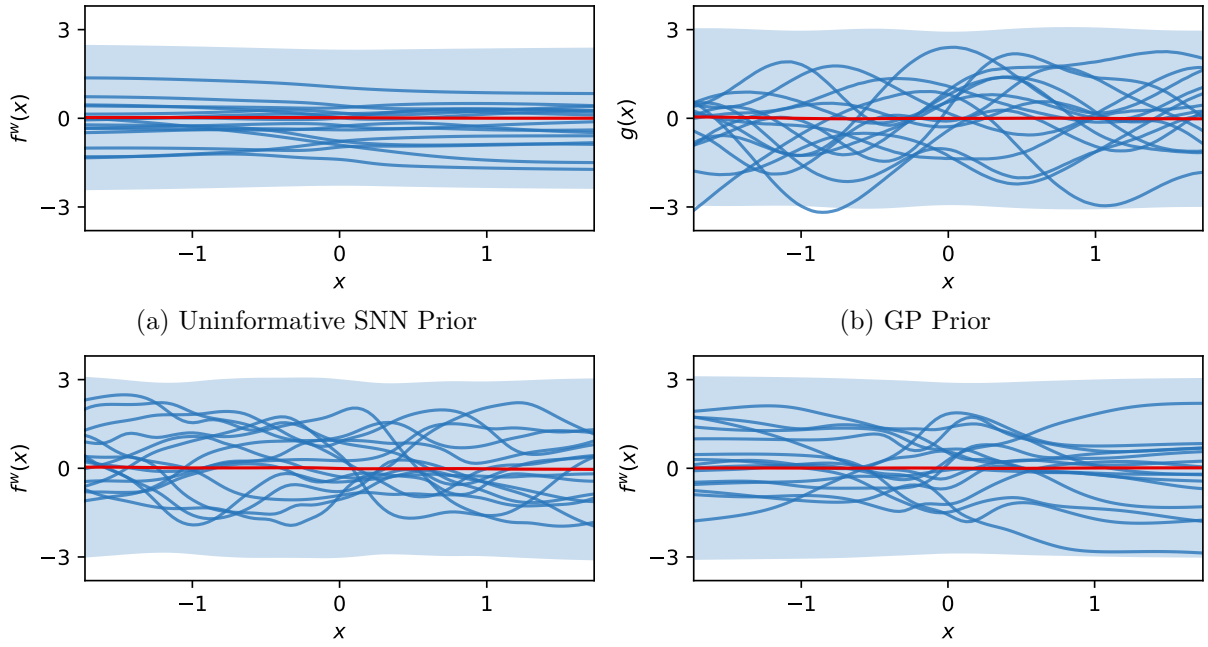


(a) Kernel SNN (b) Mean SNN (c) Kernel GMM (d) Mean GMM

(III) VGG-2 (Drop) + [1x128] (Drop) trained on CIFAR-10

Figure 7: Kernels and means for various SNNs obtained via Monte Carlo sampling (left two columns) and of our GMM approximation (right two columns) for 30 input points on MNIST in (I) and CIFAR-10 in (II) and (III). Instead of the full kernel matrices, which have dimensionality 300×300, we store the trace of the sub-blocks in the output dimension, and show the 30×30 matrix that captures the covariance between the input points. The rows and columns are grouped according to the classes, where the colored regions on the y-axis in the mean plots mark the classes. (I) shows the approximate kernels and the predictive means for a fully-connected SNN trained via VI, which gives 98% test accuracy. We see in the kernel that examples with the same class labels are correlated. In (II), the VGG network with dropout is combined with a linear stochastic layer on the more complex CIFAR-10 data set, achieving 65% accuracy. (III) shows VGG stacked with a linear layer, to which dropout is applied, achieving 64% accuracy on CIFAR-10.

matrix clearly shows the correlations trained by the SNN. For the dropout network, the GMM visualizes the limitations of dropout learning in capturing correlation among data points. This finding aligns with the result in Gal and Ghahramani (2016), which shows that Bayesian inference via dropout is equivalent to VI inference with an isotropic Gaussian over the parameters per layer.



(a) Uninformative SNN Prior

(b) GP Prior

(c) GP Induced SNN Prior (Ours)

(d) GP Induced SNN Prior (Tran et al., 2022)

Figure 8: Visualization of the prior distribution of (a) a SNN with an isotropic Gaussian distribution over its parameters; (b) a zero mean GP with the RBF kernel; (c-d) SNNs with a Gaussian distribution over its weight that is optimized to mimic the GP in (b) via, respectively, our framework and the approach proposed in Tran et al. (2022). The SNN has 2 hidden layers, 128 neurons per layer, and the tanh activation function, while the GP has zero mean and a RBF kernel with length-scale 0.5 and signal variance 1. Red lines, blue shading, and blue lines respectively indicate the mean, 99.7% credible intervals, and a subset of the MC samples from the output distributions.

8.3 Functional Priors for Neural Networks via GMM Approximation

In this section, to highlight another application of our framework, we show how our results can be used to encode functional priors in SNNs. We start from Figure 8a, where we plot functions sampled from a two hidden layer SNN with an isotropic Gaussian distribution over its parameters, which is generally the default choice for priors in Bayesian learning (Fortuin, 2022). As it is possible to observe in the figure, the sampled functions tend to form straight horizontal lines; this is a well-known pathology of neural networks, which becomes more and more exacerbated with deep architectures (Duvenaud et al., 2014), Instead, one may want to consider a prior model that reflects certain characteristics that we expect from the underlying true function, e.g., smoothness or oscillatory behavior. Such characteristics are commonly encoded via a Gaussian process with a given mean and kernel (Tran et al., 2022), as shown for instance in Figure 8b, where we plot samples

from a GP with zero mean and the RBF kernel (MacKay, 1996). We now demonstrate how our framework can be used to encode an informative functional prior, represented as a GP, into a SNN. In particular, in what follows, we assume centered Mean-Field Gaussian priors on the parameters, i.e., $p_w = \mathcal{N}(\bar{0}, \mathrm{diag}(\psi))$, where ψ is a vector of possibly different parameters representing the variance of each weight. Our objective is to optimize ψ to minimize the 2-Wasserstein distance between the distributions of the GP and the SNN at a finite set of input points, which we do by following the approach described in Section 7.2. For the low-dimensional 1D regression problem, we use uniformly sampled input points. For the high-dimensional UCI problems, we use the training sets augmented with noisy samples. We solve the optimization problem as described in Section 7.2, setting β to 0.01, using mini-batch gradient descent with randomly sampled batches.

Length-Scale RBF Kernel	Network Architecture	Uninformative Prior	GP Induced Prior (Tran et al., 2022)	GP Induced Prior (Ours)
1	[2x64]	0.97	0.30	0.25
	[2x128]	0.46	0.30	0.25
0.75	[2x64]	1.09	0.39	0.32
	[2x128]	0.56	0.39	0.31
0.5	[2x64]	1.23	0.53	0.47
	[2x128]	0.69	0.53	0.43

Table 4: The empirical estimates of the relative 2-Wasserstein distance between the prior distributions of zero-mean GPs with the RBF kernel and SNNs with uninformative or GP-induced priors over the parameters, at a subset of 20 points from the test set. The GP-induced prior are either obtained via Algorithm 1 with signatures of size 10 and compression size (M_k) of 1 for all layers as described in Section 7.2, or via the method in Tran et al. (2022).

1D Regression Benchmark We start our analysis with the 1D example in Figure 8, where we compare our framework with the state-of-the-art approach for prior tuning by Tran et al. (2022). Figure 8c demonstrates the ability of our approach to encode the oscillating behavior of the GP in the prior distribution of the SNN. In contrast, the SNN prior obtained using the method in Tran et al. (2022) in Figure 8d is visually less accurate in capturing the correlation imposed by the RBF kernel of the GP. The performance difference can be explained by the fact that Tran et al. (2022) optimize the 1-Wasserstein distance between the GP and SNN, which relates to the closeness in the first moment (the mean), as they rely on its dual formulation. In contrast, we optimize for the 2-Wasserstein distance, which relates to the closeness in the second moment, including both mean and correlations (see Lemma 28). In Table 4, we investigate the quantitative difference in relative 2-Wasserstein distance between the prior distributions of the GP and SNN in Figure 8, among other network architectures and GP settings. The results show that our method consistently outperforms Tran et al. (2022) on all considered settings.

UCI Regression Benchmark We continue our analysis on several UCI regression data sets to investigate whether SNNs with an informative prior lead to improved posterior performance compared to uninformative priors. We evaluate the impact of an informative prior on the predictive performance of SNNs using the root-mean-square error (RMSE) and negative log-likelihood (NLL) metrics. The RMSE solely measures the accuracy of the mean predictions, whereas the NLL evaluates whether the predicted distribution matches the actual distribution of the outcomes, taking into account the uncertainty of the predictions. Lower RMSE and NLL values indicate better perfor-

mance. Figure 9 shows that for all data sets, the GP outperforms the SNN with an uninformative prior both in terms of mean and uncertainty accuracy. Encoding the GP prior in the SNN greatly improves its predictive performance, bringing it closer to that of the GP, with the remaining gap explained by the VI approximation error of the posterior distribution.

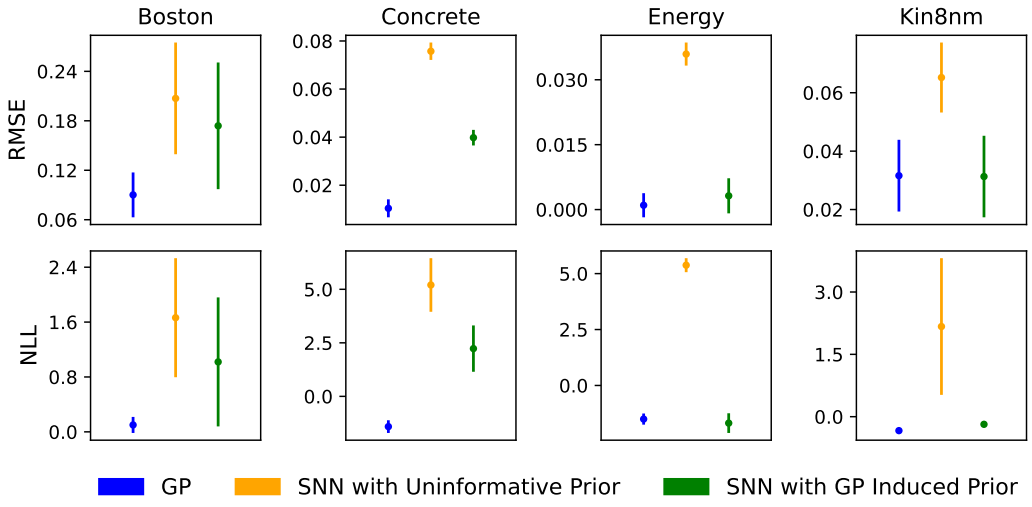


Figure 9: Report of the predictive performance of a GP with zero-mean and RBF kernel, and a $[2 \times 128]$ SNN with tanh activation function on several UCI regression data sets in terms of the root-mean-squared error (RMSE) and negative log-likelihood (NLL). The dots and bars represent the means and standard deviations over the 10 random test splits of the data sets, respectively. The GP results are shown in blue, the SNN results with an uninformative isotropic Gaussian prior are in yellow, and the results for the SNN with the GP-induced informative priors obtained via our approach are in green.

9. Conclusion

We introduced a novel algorithmic framework to approximate the input-output distribution of a SNN with a GMM, providing bounds on the approximation error in terms of the Wasserstein distance. Our framework relies on techniques from quantization theory and optimal transport, and is based on a novel approximation scheme for GMMs using signature approximations. We performed a detailed theoretical analysis of the error introduced by our GMM approximations and complemented the theoretical analysis with extensive empirical validation, showing the efficacy of our methods in both regression and classification tasks, and for various applications, including prior selection and uncertainty quantification. An interesting future direction is to employ the framework in safety-critical control, planning, or decision-making applications, where our method critically enables principled uncertainty quantification. We should stress that in this paper we focused on finding GMM approximations of neural networks at a finite set of input points, thus also accounting for the correlations among these points. While reasoning for a finite set of input points is standard in stochastic approximations (Yaida, 2020; Basteri and Trevisan, 2024) and has many applications, for some other applications, such as adversarial robustness of Bayesian neural networks (Wicker et al., 2020), one may require approximations that are valid for compact sets of input points. We believe this is an interesting future research question.

Acknowledgments

This work was supported in part by the NSF grant 2039062.

Appendix A. Properties of the Wasserstein Distance

The results in this work rely on several less commonly reported properties of the Wasserstein distance, which are recalled below.

Lemma 28 (Closeness in $\mathbb{W}_\rho$ Implies Closeness in ρ-Moments) *For $p, q \in \mathcal{P}_\rho(\mathbb{R}^n)$ and $\rho \in \mathbb{N}$, it holds that*

$$\left|\mathbb{E}_{x\sim p}[\|x\|^\rho]^{1/\rho} - \mathbb{E}_{z\sim q}[\|z\|^\rho]^{1/\rho}\right| \leq \mathbb{W}_\rho(p, q).$$

Proof The result follows directly from Proposition 7.29 in Villani et al. (2009), applied to the 1-Lipschitz function $\phi(x) = \|x\|$. ■

Lemma 29 (Translation and Rotation Invariance of $\mathbb{W}_\rho$) *Let $\tau : \mathbb{R}^n \to \mathbb{R}^n$ be an isometry of $\mathbb{R}^n$, i.e., $\tau(x) = Rx + b$ for some orthogonal matrix $R \in \mathbb{R}^{n\times n}$ and vector $b \in \mathbb{R}^n$. Then, for $\rho \in \mathbb{N}$ and $p, q \in \mathcal{P}(\mathbb{R}^n)$, we have*

$$\mathbb{W}_\rho(\tau\#p, \tau\#q) = \mathbb{W}_\rho(p, q).$$

Proof Since τ preserves the Euclidean norm, $\|\tau(x) - \tau(z)\| = \|x - z\|$, any coupling $\gamma \in \Gamma(p, q)$ is mapped to a coupling $(\tau \times \tau)\#\gamma \in \Gamma(\tau\#p, \tau\#q)$ with the same transport cost. The optimal value is therefore unchanged. ■

Lemma 30 ($\mathbb{W}_2$ for Product Distributions) *Let $p, q \in \mathcal{P}(\mathbb{R}^n)$ be joint distributions that are independent across dimensions. Then,*

$$\mathbb{W}_2^2(p, q) = \sum_{l=1}^{n} \mathbb{W}_2^2\left(p^{(l)}, q^{(l)}\right),$$

where $p^{(l)}$ and $q^{(l)}$ denote the marginals of p and q on the l-th dimension.

Proof See (Panaretos and Zemel, 2019, Section 2, fourth bullet point). ■

Lemma 31 ($\mathbb{W}_2$ Distance for Mixture Distributions) *Let $p, q \in \mathcal{P}(\mathbb{R}^n)$ be mixture distributions, i.e., $p = \sum_{i=1}^{M} \pi_p^{(i)} p_i$ and $q = \sum_{j=1}^{N} \pi_q^{(j)} q_j$, where $\pi_p \in \Pi_M$, $\pi_q \in \Pi_N$, and $p_i, q_j \in \mathcal{P}(\mathbb{R}^n)$. Then,*

$$\mathbb{W}_2^2(p, q) \leq \inf_{\bar{\pi}\in\Gamma(\pi_p,\pi_q)} \sum_{i=1}^{M}\sum_{j=1}^{N} \bar{\pi}^{(i,j)} \mathbb{W}_2^2(p_i, q_j),$$

where $\Gamma(\pi_p, \pi_q) \subset \Pi_{M\times N}$ denotes the set of couplings between π_p and π_q.

Proof Consider any coupling $\bar{\pi} \in \Gamma(\pi_p, \pi_q)$, and for each pair (i, j) let $\gamma_{i,j}$ be an optimal coupling between p_i and q_j. Then the mixture $\gamma = \sum_{i,j} \bar{\pi}_{i,j}\gamma_{i,j}$, is a valid coupling between p and q with transport cost $\sum_{i,j} \bar{\pi}_{i,j} W_2^2(p_i, q_j)$. Taking the infimum over all $\bar{\pi} \in \Gamma(\pi_p, \pi_q)$ completes the proof. ■

Lemma 32 ($\mathbb{W}_\rho$ Distance under Lipschitz Parametrized Stochastic Operations) *Let $\rho \in \mathbb{N}$, $L^w : \mathbb{R}^n \to \mathbb{R}^m$ be a $\mathcal{L}_{L^w}$-Lipschitz continuous function parametrized by $w \in \mathbb{R}^d$, and let w be distributed according to $p_w \in \mathcal{P}_\rho(\mathbb{R}^d)$. Then, for any $p, q \in \mathcal{P}_\rho(\mathbb{R}^n)$, it holds that*

$$\mathbb{W}_\rho^\rho \left(\mathbb{E}_{x\sim p} \left[L^w(x) \# p_w \right], \mathbb{E}_{z\sim q} \left[L^w(z) \# p_w \right] \right) \leq \mathbb{E}_{w\sim p_w} \left[\mathcal{L}_{L^w}^\rho \right] \mathbb{W}_\rho^\rho(p, q).$$

Proof Since $\|L^w(x) - L^w(z)\| \leq \mathcal{L}_{L^w} \|x - z\|$, it follows that

$$\begin{aligned} \mathbb{W}_\rho^\rho \left(\mathbb{E}_{x\sim p}[L^w(x)\# p_w], \mathbb{E}_{z\sim q}[L^w(z)\# p_w] \right) &= \inf_{\gamma \in \Gamma(p_w, p, q)} \mathbb{E}_{(w,x,z)\sim\gamma} \left[\|L^w(x) - L^w(z)\|^\rho \right] \\ &\leq \inf_{\gamma \in \Gamma(p_w, p, q)} \mathbb{E}_{(w,x,z)\sim\gamma} \left[\mathcal{L}_{L^w} \|x - z\|^\rho \right]. \end{aligned}$$

Using that $p_w \times \Gamma(p, q) \subset \Gamma(p_w, p, q)$, we obtain

$$\begin{aligned} \inf_{\gamma \in \Gamma(p_w, p, q)} \mathbb{E}_{(w,x,z)\sim\gamma} \left[\mathcal{L}_{L^w}^\rho \|x - z\|^\rho \right] &\leq \mathbb{E}_{w\sim p_w} \left[\mathcal{L}_{L^w}^\rho \right] \inf_{\gamma \in \Gamma(p,q)} \mathbb{E}_{(x,z)\sim\gamma} \left[\|x - z\|^\rho \right] \\ &= \mathbb{E}_{w\sim p_w} \left[\mathcal{L}_{L^w}^\rho \right] \mathbb{W}_\rho^\rho(p, q), \end{aligned}$$

which concludes the proof. ■

Appendix B. Proofs

We present the proofs for all results discussed in the main text of this work ordered per section.

B.1 Proofs of the Results in Section 5

Proof of Proposition 10: $\mathbb{W}_2$-Error of the Signature of a Standard Gaussian

From Proposition 9, we have that

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}} \# \mathcal{N}(0, 1), \mathcal{N}(0, 1)) = \sum_{i=1}^{N} \pi^{(i)} \mathbb{E}_{z\sim\mathcal{N}(0,1)} \left[(z - c_i)^2 \mid z \in [l_i, u_i] \right].$$

where

$$\pi^{(i)} = \mathbb{P}_{z\sim\mathcal{N}(0,1)}[z \in [l_i, u_i]] = \Phi(u_i) - \Phi(l_i).$$

Using the closed-form expression for the mean $\mu_i \mathbb{E}_{z\sim\mathcal{N}(0,1)}[z \mid z \in [l_i, u_i]]$ and variance $\nu_i = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \left[z^2 \mid z \in [l_i, u_i] \right] - \mu_i^2$ of a standard univariate Gaussian random variable truncated on $[l_i, u_i]$ as given by Eqn (9), respectively, and derived in Section 3 of Manjunath and Wilhelm (2021), we can write the conditional expectation term as:

$$\begin{aligned} &\mathbb{E}_{z\sim\mathcal{N}(0,1)} \left[(z - c_i)^2 \mid z \in [l_i, u_i] \right] \\ &\quad = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \left[z^2 \mid z \in [l_i, u_i] \right] - 2c_i \mathbb{E}_{z\sim\mathcal{N}(0,1)} \left[z \mid z \in [l_i, u_i] \right] + c_i^2 \\ &\quad = \nu_i + \mu_i^2 - 2c_i\mu_i + c_i^2 \\ &\quad = \nu_i + (\mu_i - c_i)^2. \end{aligned} \tag{28}$$

Substituting this expression into the original sum concludes the proof. ■

Proof of Corollary 11: $\mathbb{W}_2$-Error of the Signature of a Multivariate Gaussian

Let $\tau : \mathbb{R}^n \to \mathbb{R}^n$ denote the transformation defined by $\tau(x) = V^T(x - m)$, where V and m are, respectively, the (orthogonal) eigenvector matrix and mean of the Gaussian distribution. By Lemma 29, the 2-Wasserstein is invariant under translations and orthogonal transformations, and thus

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}}\#\mathcal{N}(m,\Sigma),\mathcal{N}(m,\Sigma)) = \mathbb{W}_2^2(\tau\#\Delta_{\mathcal{C}}\#\mathcal{N}(m,\Sigma),\tau\#\mathcal{N}(m,\Sigma)).$$

Since, $\Sigma = V\mathrm{diag}(\lambda)V^T$, it holds that

$$\tau\#\mathcal{N}(m,\Sigma) = \mathcal{N}(\bar{0},\mathrm{diag}(\lambda)) = \mathrm{diag}(\lambda)^{1/2}\#\mathcal{N}(\bar{0},I),$$

where, with a slight abuse of notation, we write $\mathrm{diag}(\lambda)^{1/2}\#p$ to denote the pushforward under the linear map $x \mapsto \mathrm{diag}(\lambda)^{1/2}x$. Next, consider the pushforward of the signature:

$$\begin{aligned}
\tau\#\Delta_{\mathcal{C}}\#\mathcal{N}(m,\Sigma) &= \Delta_{\mathrm{Post}(\mathcal{C}-\{m\},V^T)}\#\tau\#\mathcal{N}(m,\Sigma)\\
&= \Delta_{\mathrm{Post}(\mathcal{C}-\{m\},V^T)}\#\mathrm{diag}(\lambda)^{1/2}\#\mathcal{N}(\bar{0},I)\\
&= \mathrm{diag}(\lambda)^{1/2}\#\Delta_{\mathrm{Post}(\mathcal{C}-\{m\},T)}\#\mathcal{N}(\bar{0},I)\\
&= \mathrm{diag}(\lambda)^{1/2}\#\Delta_{\mathcal{C}^1\times\mathcal{C}^2\times\ldots\times\mathcal{C}^n}\#\mathcal{N}(\bar{0},I),
\end{aligned}$$

where the final equality holds due to the axis-aligned grid assumption in (10). Since both terms are product distributions, we can apply Lemma 30, yielding

$$\begin{aligned}
\mathbb{W}_2^2(\tau\#\Delta_{\mathcal{C}}\#\mathcal{N}(m,\Sigma),\tau\#\mathcal{N}(m,\Sigma)) &= \sum_{l=1}^{n}\mathbb{W}_2^2\left(\sqrt{\lambda^{(l)}}\#\mathcal{N}(0,1),\sqrt{\lambda^{(l)}}\#\Delta_{\mathcal{C}^l}\#\mathcal{N}(0,1)\right)\\
&= \sum_{l=1}^{n}\mathbb{W}_2^2\lambda^{(l)}\left(\mathcal{N}(0,1),\Delta_{\mathcal{C}^l}\#\mathcal{N}(0,1)\right),
\end{aligned}$$

where the final equality follows from the homogeneity of the 2-Wasserstein distance. This completes the proof. ■

Proof of Proposition 13:

If $N = 1$, Eqn (12) gives $\mathcal{C} = \{0\}$ and Proposition 9 yields

$$\mathbb{W}_2^2(\Delta_{\{0\}}\#\mathcal{N}(0,1),\mathcal{N}(0,1)) = \int z^2\,\phi(dz) = 1 = \frac{4\log 1}{1^2} + r(1) = r(1),$$

since $\log 1 = 0$ and $r(1) = 1$ by Eqn (14), and where ϕ is the pdf of a standard (univariate) Gaussian distribution. Hence the bound holds for $N = 1$. For the remainder of the proof assume $N \geq 2$. Let $\eta = \frac{2\sqrt{\log N}}{N}$ and $\mathcal{C} = \{c_i\}_{i=1}^N$ with $c_i = \eta(2i - N - 1)$. The induced partition (Eqn 6) is

$$\mathcal{R}_1 = [-\infty, c_1 + \eta]\,, \quad \mathcal{R}_i = [c_i - \eta, c_i + \eta]\,, \quad \forall i \in \{2,\ldots,N-1\}, \quad \mathcal{R}_N = [c_N - \eta, \infty]\,.$$

By Proposition 9, we have

$$\begin{aligned}
&\mathbb{W}_2^2(\Delta_{\mathcal{C}}\#\mathcal{N}(0,1),\mathcal{N}(0,1))\\
&\quad= \sum_{i=2}^{N-1}\int_{c_i-\eta}^{c_i+\eta}(z-c_i)^2\phi(dz) + \int_{-\infty}^{c_1+\eta}(z-c_1)^2\phi(dz) + \int_{c_N-\eta}^{\infty}(z-c_N)^2\phi(dz)\\
&\quad= \sum_{i=1}^{N}\int_{c_i-\eta}^{c_i+\eta}(z-c_i)^2\phi(dz) + \int_{-\infty}^{-\eta N}(z-c_1)^2\phi(dz) + \int_{\eta N}^{\infty}(z-c_N)^2\phi(dz),
\end{aligned}$$

where in the last step we split the unbounded regions and used $c_1 = -\eta N + \eta$, $c_N = \eta N - \eta$.

Bounded integrals: since $|z - c_i| \le \eta$ for all $z \in \mathcal{R}_i$,

$$\begin{aligned}\sum_{i=1}^{N} \int_{c_i-\eta}^{c_i+\eta} (z-c_i)^2 \phi(dz) &\le \eta^2 \sum_{i=1}^{N} [\Phi(c_i+\eta) - \Phi(c_i-\eta)] \\ &= \eta^2 \left[\Phi\left(\eta N\right) - \Phi\left(-\eta N\right)\right] = \eta^2 \left[2\Phi\left(\eta N\right) - 1\right],\end{aligned}$$

where the last equalities follow from symmetry and telescoping of the CDF terms.

Unbounded integrals: by symmetry ($c_1 = -c_N$),

$$\int_{-\infty}^{-\eta N} (z-c_1)^2 \phi(dz) + \int_{\eta N}^{\infty} (z-c_N)^2 \phi(dz) = 2\int_{\eta N}^{\infty} (z-c_N)^2 \phi(dz).$$

For $z \ge \eta N$, $z - c_N = z - \eta N + \eta \le z$, hence

$$\int_{\eta N}^{\infty} (z-c_N)^2 \mathcal{N}(dz \mid 0,1) \le \int_{\eta N}^{\infty} z^2 \mathcal{N}(dz \mid 0,1) = [1 - \Phi(\eta N)] + \eta N \phi(\eta N),$$

using the truncated second-moment identity for standard Gaussians (Eqn 28).

Combining and denoting $L = \eta N = 2\sqrt{\log N}$,

$$\begin{aligned}\mathbb{W}_2^2(\Delta_{\mathcal{C}} \# \mathcal{N}(0,1), \mathcal{N}(0,1)) &\le \eta^2 \left[2\Phi\left(L\right) - 1\right] + 2[1 - \Phi(L)] + 2L\phi(L) \\ &= -2\eta^2 \left[1 - \Phi\left(L\right)\right] + \eta^2 + 2[1 - \Phi(L)] + 2L\phi(L) \\ &= 2(1-\eta^2)\left(1 - \Phi\left(L\right)\right) + \eta^2 + 2L\phi\left(L\right) \\ &= 2\left(1 - \frac{L^2}{N^2}\right)\left(1 - \Phi\left(L\right)\right) + \frac{L^2}{N^2} + 2L\phi\left(L\right) \qquad (*)\end{aligned}$$

By Mills' ratio $1 - \Phi(L) \le \frac{\phi(L)}{L}$ and $0 \le \frac{L^2}{N^2} \le 1$, we bound the first term in $(*)$ by $2\frac{\phi(L)}{L}$, yielding

$$(*) \le 2\frac{\phi(L)}{L} + \frac{L^2}{N^2} + 2L\phi(L) = 2\phi(L)\left(L + \frac{1}{L}\right) + \frac{L^2}{N^2}.$$

With $L = 2\sqrt{\log N}$ we have $\phi(L) = \frac{1}{\sqrt{2\pi}} e^{-L^2/2} = \frac{1}{\sqrt{2\pi}} e^{-2\log N} = \frac{1}{N^2\sqrt{2\pi}}$, so

$$2\phi(L)\left(L + \tfrac{1}{L}\right) = \frac{4}{N^2\sqrt{2\pi}}\left(\sqrt{\log N} + \frac{1}{4\sqrt{\log N}}\right).$$

Therefore

$$\mathbb{W}_2^2(\Delta_{\mathcal{C}} \# \mathcal{N}(0,1), \mathcal{N}(0,1)) \le \frac{4\log N}{N^2} + \frac{4}{N^2\sqrt{2\pi}}\left(\sqrt{\log N} + \frac{1}{4\sqrt{\log N}}\right) = \frac{4\log N}{N^2} + r(N).$$

This completes the proof. ■

Proof of Corollary 15

From Eqn (11), the 2-Wasserstein error of the component-wise signature of a Gaussian mixture is bounded by the mixture-weighted sum of the signature errors of its components:

$$\mathbb{W}_2^2\big(p, \mathbf{\Delta}_{\{\mathcal{C}_i\}_{i=1}^M} \# p\big) \leq \sum_{i=1}^{M} \tilde{\pi}_i \, \mathbb{W}_2^2\big(\mathcal{N}(m_i, \Sigma_i), \Delta_{\mathcal{C}_i} \# \mathcal{N}(m_i, \Sigma_i)\big).$$

For each component i, diagonalize $\Sigma_i = V_i \mathrm{diag}(\lambda_i) V_i^T$ and apply Corollary 11 together with the axis-aligned grid assumption (10) to obtain

$$\mathbb{W}_2^2\big(\mathcal{N}(m_i, \Sigma_i), \Delta_{\mathcal{C}_i} \# \mathcal{N}(m_i, \Sigma_i)\big) = \sum_{j=1}^{n} \lambda_{i,j} \, \mathbb{W}_2^2\big(\mathcal{N}(0,1), \Delta_{\mathcal{C}_i^j} \# \mathcal{N}(0,1)\big).$$

Proposition 13 applied to each one-dimensional uniform grid of points $\mathcal{C}_i^j$ (of size $N_{i,j}$) yields

$$\mathbb{W}_2^2\big(\mathcal{N}(0,1), \Delta_{\mathcal{C}_i^j} \# \mathcal{N}(0,1)\big) \leq \frac{4 \log N_{i,j}}{N_{i,j}^2} + r(N_{i,j}),$$

with $r(\cdot)$ given in Eqn (14). Combining the three displays finishes the proof:

$$\mathbb{W}_2^2\big(p, \mathbf{\Delta}_{\{\mathcal{C}_i\}_{i=1}^M} \# p\big) \leq \sum_{i=1}^{M} \tilde{\pi}_i \sum_{j=1}^{n} \lambda_{i,j} \left(\frac{4 \log N_{i,j}}{N_{i,j}^2} + r(N_{i,j}) \right). \blacksquare$$

B.2 Proofs of the Results in Section 6

Proofs of the $\mathbb{W}_2$ Distance Bounds in Table 1

The bound for the signature and compression operations directly follows from the triangle inequality. The bound for the affine transformation is a direct application of Lemma 32, as is the bound on $\mathbb{W}(\sigma \# p, \sigma \# q)$.

To derive the bound on $\mathbb{W}_2^2(\sigma \# p, \sigma \# \Delta_{\mathcal{C}} \# p)$, we invoke Proposition 9, which yields

$$\mathbb{W}_\rho^\rho(\sigma \# p, \sigma \# \Delta_{\mathcal{C}} \# p) = \sum_{i=1}^{N} \pi^{(i)} \mathbb{E}_{z \sim p}[\|\sigma(z) - \sigma(c_i)\|^\rho \mid z \in \mathcal{R}_i].$$

Since the activation function σ is $\mathcal{L}_{\sigma|\mathcal{R}_i}$-Lipschitz on region $\mathcal{R}_i$, we have $\|\sigma(z) - \sigma(c_i)\| \leq \mathcal{L}_{\sigma|\mathcal{R}_i} \|z - c_i\|$ for all $z, c_i \in \mathcal{R}_i$. Substituting this bound gives

$$\sum_{i=1}^{N} \pi^{(i)} \mathbb{E}_{z \sim p}[\|\sigma(z) - \sigma(c_i)\|^\rho \mid z \in \mathcal{R}_i] \leq \sum_{i=1}^{N} \mathcal{L}_{\sigma|\mathcal{R}_i}^\rho \pi^{(i)} \mathbb{E}_{z \sim p}[\|z - c_i\|^\rho \mid z \in \mathcal{R}_i].$$

Proof of Theorem 17: $\mathbb{W}_2$-Error of the GMM Approximation of a SNN at a Single Input Point

We prove Theorem 17 via induction and generalize the result to the ρ-Wasserstein distance with $\rho \in \{1, 2\}$. To do so, we first derive the base case of the induction. For $k \in \{1, \ldots, K\}$, we have

that:

$$
\begin{aligned}
&\mathbb{W}_\rho\left(p_{nn(x),k+1}, q_{nn(x),k+1}\right) \\
&\quad \text{(By Eqns 4 \& 15)} \\
&= \mathbb{W}_\rho\left(\mathbb{E}_{z_k\sim p_{nn(x),k}}\left[L^{w_k}(\sigma(z_k))\#p_{w_k}\right], \mathbb{E}_{\tilde{z}_k\sim G_{M_k}(q_{nn(x),k})}\left[L^{w_k}(\sigma(\Delta_{\mathcal{C}_k}(\tilde{z}_k)))\#p_{w_k}\right]\right) \\
&\quad \text{(By the bound for the affine operation in Table 1)} \\
&\leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^\rho_{L^{w_k}}\right]^{1/\rho}\mathbb{W}_\rho\left(\sigma\#p_{nn(x),k}, \sigma\#\Delta_{\mathcal{C}_k}\#G_{M_k}(q_{nn(x),k})\right) \\
&\quad \text{(By the triangle inequality)} \\
&\leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^\rho_{L^{w_k}}\right]^{1/\rho}\left[\mathbb{W}_\rho\left(\sigma\#p_{nn(x),k}, \sigma\#q_{nn(x),k}\right) + \mathbb{W}_\rho\left(\sigma\#q_{nn(x),k}, \sigma\#G_{M_k}(q_{nn(x),k})\right)\right. \\
&\qquad \left.+\mathbb{W}_\rho\left(\sigma\#G_{M_k}(q_{nn(x),k}), \sigma\#\Delta_{\mathcal{C}_k}\#G_{M_k}(q_{nn(x),k})\right)\right] \\
&\quad \text{(By Lemma 32, i.e., applying the bound for the activation operation in Table 1)} \\
&\leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^\rho_{L^{w_k}}\right]^{1/\rho}\mathcal{L}_\sigma\left[\mathbb{W}_\rho\left(p_{nn(x),k}, q_{nn(x),k}\right) + \mathbb{W}_\rho\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)\right. \\
&\qquad \left.+\mathbb{W}_\rho\left(G_{M_k}(q_{nn(x),k}), \Delta_{\mathcal{C}_k}\#G_{M_k}(q_{nn(x),k})\right)\right] \\
&\quad \text{(By the definition of the } \mathbb{MW}_2\text{-distance, and because } \mathbb{W}_1 \leq \mathbb{W}_2) \\
&\leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^\rho_{L^{w_k}}\right]^{1/\rho}\mathcal{L}_\sigma\left[\mathbb{W}_\rho\left(p_{nn(x),k}, q_{nn(x),k}\right) + \mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)\right. \\
&\qquad \left.+\mathbb{W}_\rho\left(G_{M_k}(q_{nn(x),k}), \Delta_{\mathcal{C}_k}\#G_{M_k}(q_{nn(x),k})\right)\right] \\
&\quad \text{(By Eqn 11 and because } \mathbb{W}_1 \leq \mathbb{W}_2) \\
&\leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^\rho_{L^{w_k}}\right]^{1/\rho}\mathcal{L}_\sigma\left[\mathbb{W}_\rho\left(p_{nn(x),k}, q_{nn(x),k}\right) + \mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)\right. \\
&\qquad \left.+\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(x),k})\right)\right]
\end{aligned}
$$

Hence, if $\mathbb{W}_\rho\left(p_{nn(x),k}, q_{nn(x),k}\right) \leq \hat{\mathbb{W}}_{\rho,k}$, then

$$
\begin{aligned}
\mathbb{W}_\rho\left(p_{nn(x),k+1}, q_{nn(x),k+1}\right) \leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^\rho_{L^{w_k}}\right]^{1/\rho}\mathcal{L}_\sigma\left[\hat{\mathbb{W}}_{\rho,k} + \mathbb{MW}_2\left(q_{nn(x),k}, G_{M_k}(q_{nn(x),k})\right)\right. \\
\left.+\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(x),k})\right)\right] = \hat{\mathbb{W}}_{\rho,k+1}
\end{aligned}
$$

The assumption is naturally satisfied for $k = 1$, such that by induction, it holds that

$$
\mathbb{W}_\rho\left(p_{nn(x)}, q_{nn(x)}\right) \leq \hat{\mathbb{W}}_{\rho,K+1}.
$$

■

Proof of Corollary 20: $\mathbb{W}_2$-Error of the GMM Approximation of a SNN at a Set of Input Points

The corollary follows directly from Theorem 17 by replacing L^{w_k} with the stacked affine operation

$$
\boldsymbol{L}^{w_k}\left((z_1^T, \ldots, z_D^T)^T\right) = \left(L^{w_k}(z_1)^T, \ldots, L^{w_k}(z_D)^T\right)^T,
$$

and noting that the Lipschitz constant of $\boldsymbol{L}^{w_k}$ can be taken equal to the Lipschitz constant of L^{w_k}. To see the latter, note that for any $\bar{z} = \left(z_1^T, \ldots, z_D^T\right)^T$ and $\bar{z}' = \left(z_1'^T, \ldots, z_D'^T\right)^T$, we have:

$$
\|\boldsymbol{L}^{w_k}(\bar{z}) - \boldsymbol{L}^{w_k}(\bar{z}')\|^2 = \sum_{i=1}^D \|L^{w_k}(z_i) - L_{w_k}(z_i')\|^2 \leq \sum_{i=1}^D \mathcal{L}^2_{L_{w_k}}\|z_i - z_i'\|^2 = \mathcal{L}^2_{L_{w_k}}\|\bar{z} - \bar{z}'\|^2.
$$

Therefore, $\mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^2_{\boldsymbol{L}^{w_k}}\right]^{1/2} \leq \mathbb{E}_{w_k\sim p_{w_k}}\left[\mathcal{L}^2_{L^{w_k}}\right]^{1/2}$. and the same bound from Theorem 17 applies to the set-valued input case. ■

Proof of Theorem 21: Convergence of the $\mathbb{W}_\rho$-Error of the GMM Approximation of a SNN

We prove the theorem for the ρ-Wasserstein with $\rho \in 1, 2$. Let us denote

$$\hat{\mathbb{W}}_{2,G_{M_k}} = \mathbb{MW}_2\left(q_{nn(\mathcal{X}),k}, G_{M_k}(q_{nn(\mathcal{X}),k})\right), \quad \text{and} \quad \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}} = \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\left(G_{M_k}(q_{nn(\mathcal{X}),k})\right).$$

From Corollary 20:

$$\begin{aligned}
&\mathbb{W}_\rho(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}) \\
&\quad\leq \mathcal{L}_K\left[\hat{\mathbb{W}}_{\rho,K} + \hat{\mathbb{W}}_{2,G_{M_K}} + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_K}}\right] \\
&\quad= \mathcal{L}_{K-1}\mathcal{L}_K\left[\mathbb{W}_{\rho,K-1} + \hat{\mathbb{W}}_{2,G_{M_{K-1}}} + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{K-1}}\right] + \mathcal{L}_K\left[\mathbb{W}_{2,G_{M_K}} + \mathbb{W}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_K}}\right] \\
&\quad\vdots \\
&\quad= \left(\prod_{i=1}^K \mathcal{L}_i\right)\left[\hat{\mathbb{W}}_{\rho,1} + \hat{\mathbb{W}}_{2,G_{M_1}} + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_1}}\right] + \sum_{k=2}^K\left(\prod_{i=k}^K \mathcal{L}_i\right)\left[\hat{\mathbb{W}}_{2,G_{M_k}} + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}}\right] \\
&\quad= \sum_{k=1}^K\left(\prod_{i=k}^K \mathcal{L}_i\right)\mathcal{L}_k\left[\hat{\mathbb{W}}_{2,G_{M_k}} + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\mathcal{C}_k}}\right]
\end{aligned}$$

According to Corollary 15, the construction of the signature sets $\mathcal{C}_k$ in the theorem ensures that

$$\hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\boldsymbol{\mathcal{C}}_k}} \leq \epsilon_k = \frac{\epsilon}{K\prod_{i=k}^K \mathcal{L}_i} - \hat{\mathbb{W}}_{2,G_{M_k}}.$$

Substituting this into the total bound, we get:

$$\begin{aligned}
\sum_{k=1}^K\left(\prod_{i=k}^K \mathcal{L}_i\right)\left[\hat{\mathbb{W}}_{2,G_{M_k}} + \hat{\mathbb{W}}_{2,\boldsymbol{\Delta}_{\mathcal{C}_k}}\right] &\leq \sum_{k=1}^K\left(\prod_{i=k}^K \mathcal{L}_i\right)\left[\hat{\mathbb{W}}_{2,G_{M_k}} + \epsilon_k\right] \\
&= \sum_{k=1}^K\left(\prod_{i=k}^K \mathcal{L}_i\right)\left[\frac{\epsilon}{K\prod_{i=k}^K \mathcal{L}_i}\right] = \sum_{k=1}^K \frac{\epsilon}{K} = \epsilon.
\end{aligned}$$

This completes the proof. ■

B.3 Proofs of the Results in Section 7

Proof of Corollary 26 on the $\mathbb{W}_2$ Distance between any GMM and a SNN

Since $\mathbb{W}_2$ satisfies the triangle inequality, we have:

$$\mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\right) \leq \mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right) + \mathbb{W}_2\left(q_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\right),$$

Given that $\mathbb{MW}_2$ is an upper bound for the 2-Wasserstein distance, we have:

$$\mathbb{W}_2\left(q_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\right) \leq \mathbb{MW}_2(q_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}).$$

Combining these two inequalities, we obtain:

$$\mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\right) \leq \mathbb{W}_2\left(p_{nn(\mathcal{X})}, q_{nn(\mathcal{X})}\right) + \mathbb{MW}_2\left(q_{nn(\mathcal{X})}, q_{gmm(\mathcal{X})}\right),$$

which completes the proof. ■

Proof of Proposition 23 on $\mathbb{W}_2$-Optimal Grid for Signatures for Gaussians

According to Corollary 11:

$$\mathbb{W}_2\left(\Delta_{\mathrm{Post}(\mathcal{C}^1\times\ldots\times\mathcal{C}^n, T^{-1})+\{m\}}\#\mathcal{N}(m,\Sigma), \mathcal{N}(m,\Sigma)\right) = \sum_{j=1}^{n} \lambda^{(j)} \mathbb{W}_2^2\left(\Delta_{\mathcal{C}^j}\#\mathcal{N}(0,1), \mathcal{N}(0,1)\right).$$

This relation allows us to split the optimization problem of interest in n smaller optimization problems as follows:

$$\begin{aligned}
&\underset{\substack{\{\mathcal{C}^1,\ldots,\mathcal{C}^n\}\subset\mathbb{R},\\ \sum_{i=1}^n |\mathcal{C}^j|=N}}{\operatorname{argmin}} \quad \mathbb{W}_2\left(\Delta_{\mathrm{Post}(\mathcal{C}^1\times\ldots\times\mathcal{C}^n, T^{-1})+\{m\}}\#\mathcal{N}(m,\Sigma), \mathcal{N}(m,\Sigma)\right) \\
&\quad = \underset{\substack{\{\mathcal{C}^1,\ldots,\mathcal{C}^n\}\subset\mathbb{R},\\ \sum_{i=1}^n |\mathcal{C}^j|=N}}{\operatorname{argmin}} \sum_{j=1}^{n} \lambda^{(j)} \mathbb{W}_2^2\left(\Delta_{\mathcal{C}^j}\#\mathcal{N}(0,1), \mathcal{N}(0,1)\right), \\
&\quad = \underset{\substack{\{\mathcal{C}^1,\ldots,\mathcal{C}^n\}\subset\mathbb{R},\\ \mathcal{C}^j=N_j^*, \forall j\in\{1,\ldots,n\}}}{\operatorname{argmin}} \sum_{j=1}^{n} \lambda^{(j)} \mathbb{W}_2^2\left(\Delta_{\mathcal{C}^j}\#\mathcal{N}(0,1), \mathcal{N}(0,1)\right), \\
&\quad = \left\{ \underset{\mathcal{C}^j\subset\mathbb{R}, \mathcal{C}^j=N_j^*}{\operatorname{argmin}} \mathbb{W}_2^2\left(\Delta_{\mathcal{C}^j}\#\mathcal{N}(0,1), \mathcal{N}(0,1)\right) \right\}_{j=1}^{n},
\end{aligned}$$

where in step 2 we used the optimal grid-configuration $\{N_1^*, \ldots, N_n^*\}$ as defined in Eqn(25), and in step 3, we use the independence of objective terms across dimensions. ■

Appendix C. Expected Lipschitz Constant for Stochastic Affine Operations

Here, we show how to compute the expected value of a Lipschitz constant $\mathbb{E}_{w_k\sim p_{w_k}}[\mathcal{L}^2_{L^{w_k}}]^{\frac{1}{2}}$ appearing in Theorem 17, in the case where L^{w_k} represents either a fully-connected or convolutional layer, and where $\rho \in \{1, 2\}$.

Fully-connected layers. For a fully-connected layer of the form $L^w(z) = Wz + b$, the Lipschitz constant is $\mathcal{L}_{L^w} = \|W\|$, where $\|\cdot\|$ denotes the spectral norm. When the matrix W is distributed according to $p_W \in \mathcal{P}_\rho(\mathbb{R}^{m\times n})$, computing $\mathbb{E}[\|W\|^\rho]^{1/\rho}$ is generally intractable. Instead of using the spectral norm, we therefore consider the Frobenius norm, which yields a valid Lipschitz constant: while it is not the smallest such constant, it provides a tractable upper bound. Specifically, that

$$\mathbb{E}_{W\sim p_W}[\|W\|^\rho]^{1/\rho} \leq \mathbb{E}_{W\sim p_W}\left[\|W\|_{\mathcal{F}}^\rho\right]^{1/\rho}.$$

This bound can be tightened by centering the distribution of W around its mean $M = \mathbb{E}_{W\sim p_W}[W]$ and by applying Minkowski's inequality:

$$\mathbb{E}_{W\sim p_W}[\|W\|^\rho]^{1/\rho} \leq \mathbb{E}_{W\sim p_W}[\|W-M\|^\rho]^{1/\rho} + \|M\| \leq \mathbb{E}_{W\sim p_W}\left[\|W-M\|_{\mathcal{F}}^\rho\right]^{1/\rho} + \|M\|.$$

Here, we again use that the spectral norm is upper bounded by the Frobenius norm.

For $\rho = 2$, the upper bound simplifies in terms of second moments of the entries of W. For $\rho = 1$, we additionally apply Jensen's inequality to ensure tractability:

$$\mathbb{E}_{W\sim p_W}\left[\|W-M\|_{\mathcal{F}}\right] \leq \mathbb{E}_{W\sim p_W}\left[\|W-M\|_{\mathcal{F}}^2\right]^{1/2}.$$

Convolutional layers. If L^{w_k} corresponds to a 2D convolutional operation with kernel weight matrix W, then the Lipschitz constant is given by $\mathcal{L}_{L^w} = \sqrt{m}\|W\|$, where m denotes the maximum number of times any input element contributes to the output (e.g., the number of spatial positions a kernel covers) Gouk et al. (2021). For example, with unit stride and padding of one, m equals the product of the kernel height and width. We apply the same bounding procedure as in the fully-connected setting, now including the multiplicative $\sqrt{m}$ correction factor.

Appendix D. Alternative Signature Placement: Cross Scheme

As discussed in Remark 24, we constrained signature locations to axis-aligned grids in order to obtain formal error bounds (Corollary 11). When formal bounds are not required, we may instead place signature locations on a *cross*-aligned with the principal axes of each Gaussian component. This "cross scheme" resembles classical sigma–point constructions (Julier and Uhlmann, 2004) in that points lie along eigenvector directions of the covariance; however, it fundamentally differs by (i) allowing an arbitrary number of points per axis, and (ii) assigning exact probabilities via the Voronoi partition w.r.t. to the points, rather than choosing weights to satisfy a finite set of moment constraints.

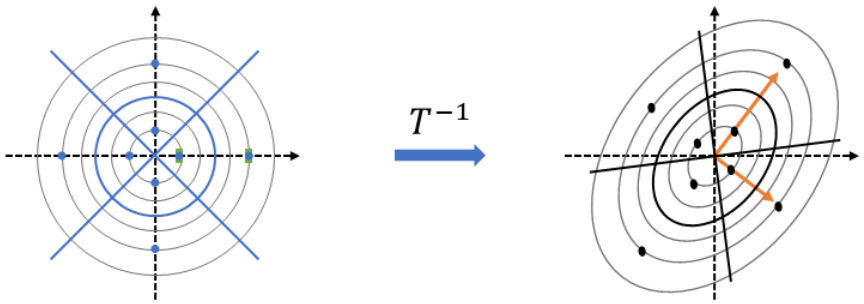


Figure 10: Construction of the cross-shaped signature of a 2D Gaussian (black dots) using the signature of a standard Gaussian (blue dots) as a reference. In the whitened eigenbasis space defined by T (orange axes), we place symmetric points along each principal axis by mirroring the template locations (green markers) across all axes to form the cross. Mapping these points back via T^{-1} yields the signature locations in the original space. The blue (left) and black (right) lines indicate the boundaries of the Voronoi regions induced by the signature locations in the transformed and original spaces, respectively.

The procedure to compute the component-wise signature of a Gaussian mixture (see Eqn 11) is illustrated in Figure 10 and summarized in Algorithm 4. First (line 2) we obtain, via Algorithm 5, the radial template (axis radii) in the transformed space (green stripes in Figure 10). Note that we set the number of radial levels $L = \max\{1, \lfloor N_{\text{cross}}/(2n) \rfloor\}$ (line 1), allocating $2n$ points per level (two per axis) while ensuring at least one level, where $\lfloor \cdot \rfloor$ denotes the floor function. Next we compute the probabilities of the Voronoi regions induced by the eventual signature points by: (i) evaluating cumulative probabilities of the n-dimensional standard Gaussian ball at the radial Voronoi boundaries using the χ^2_n CDF (line 4); and (ii) differencing these to obtain shell masses Δ_ℓ (line 5). The weight of each point at radial level ℓ is then $w_\ell = \Delta_\ell/(2n)$ (line 6). For each mixture component (lines 7-9), at every radial level ℓ and along every principal axis j, we place two symmetric points $m_i \pm r_\ell s_{i,j} a_{i,j}$ with weight w_ℓ and map them back to the original space. The heuristic in Algorithm 5 ensures that the resulting signature matches the second moment of the corresponding Gaussian component.

Algorithm 4: Cross Signatures of Gaussian mixtures

input : $q = \sum_{i=1}^{M} \tilde{\pi}^{(i)} \mathcal{N}(m_i, \Sigma_i)$, desired number of signature locations N_{cross}
output: Signature $d = \sum_{j=1}^{N} \pi^{(j)} \delta_{c_j}$
begin
1 Set number of radial levels $L = \max\{1, \lfloor N_{\text{cross}}/(2n) \rfloor\}$
2 Obtain radii $\{r_\ell\}_{\ell=1}^{L}$ via Algorithm 5
3 Define radial Voronoi boundaries $u_\ell = \frac{1}{2}(r_\ell + r_{\ell+1})$ for $\ell = 1, \ldots, L-1$
4 Compute cumulative n-D standard normal ball probabilities $B(u_\ell) = P(\|Z\|_2 \le u_\ell),\ Z \sim \mathcal{N}(0, I_n)$ using $B(u_\ell) = \Phi_{\chi^2_n}(u_\ell^2)$
5 Set shell masses $\Delta_1 = B(u_1)$, $\Delta_\ell = B(u_\ell) - B(u_{\ell-1})$ for $\ell = 2, \ldots, L-1$, and $\Delta_L = 1 - B(u_{L-1})$
6 Compute per-level point weights $w_\ell = \Delta_\ell/(2n)$
for $i \in \{1, \ldots, M\}$ **do**
7 Compute eigenvalues vector λ_i and eigenvector matrix V_i of Σ_i
8 Set unit axis directions $a_{i,j} = V_i e_j$ and scales $s_{i,j} = \sqrt{\lambda_i^{(j)}}$ for $j = 1, \ldots, n$
9 Construct $d_i = \sum_{\ell=1}^{L} \sum_{j=1}^{n} w_\ell \Big(\delta_{m_i + r_\ell s_{i,j} a_{i,j}} + \delta_{m_i - r_\ell s_{i,j} a_{i,j}} \Big)$
10 Return $d = \sum_{i=1}^{M} \tilde{\pi}^{(i)} d_i$

Algorithm 5: Axis radii for the cross scheme

input : Number of radial levels L, dimension n
output: Axis radii $\{r_\ell\}_{\ell=1}^{L}$
begin
1 Define probability edges $\{p_\ell\}_{\ell=0}^{L}$ with $p_\ell = 0.5 + \frac{\ell}{2L}$ (uniform partition of $[0.5, 1]$)
2 Compute quantiles $q_\ell = \Phi^{-1}(p_\ell)$ for $\ell = 0, \ldots, L$
3 Compute shell probabilities $s_\ell = \Phi(q_\ell) - \Phi(q_{\ell-1})$ for $\ell = 1, \ldots, L$
4 Set radii $r_\ell = \big(\phi(q_{\ell-1}) - \phi(q_\ell)\big)/s_\ell \sqrt{n}$ for $\ell = 1, \ldots, L$ (positive since $\phi(q_{\ell-1}) > \phi(q_\ell)$)
5 Return $\{r_\ell\}_{\ell=1}^{L}$.

Appendix E. Compression for Dropout-Induced Bernoulli Mixtures

We present an alternative to Algorithm 3 for performing compression in the case of mixtures of multivariate Bernoulli distributions. We begin by showing that such mixtures naturally arise in dropout networks and that compression is necessary to enable tractable analysis of their output distributions. We then introduce a structure-aware compression method that yields a closed-form upper bound on the resulting Wasserstein distance.

Let $\phi_b : \mathbb{R}^n \to \mathbb{R}^n$ be the masking function defined by $\phi_b(z) = b \odot z$, where $b \in \{0,1\}^n$ is a binary vector and $\odot$ denotes the element-wise multiplication operation. In dropout, b is sampled from a multivariate Bernoulli distribution d_b, given by

$$d_b = \sum_{b \in \{0,1\}^n} \theta^{\sum_{l=1}^n b^{(l)}} (1-\theta)^{n - \sum_{l=1}^n b^{(l)}} \delta_b,$$

where $\theta \in [0,1]$ is the dropout rate. Note that d_b is a discrete distribution with support $\{0,1\}^n$. Given any input distribution p_{in}, the output distribution of a dropout layer is defined as

$$p_{out} = \mathbb{E}_{b \sim d_b}[\phi_b \# p_{in}],$$

which is a mixture distribution of 2^n components. If the input distribution p_{in} is a discrete distribution of size N, such as the distribution at an intermediate layer of a dropout network, or a signature approximation in a VI-trained network, then the output becomes a discrete distribution with support size $N \cdot 2^n$. Specifically, for $d_{in} = \sum_{i=1}^N \pi^{(i)} \delta_{c_i}$, we have

$$d_{out} = \mathbb{E}_{b \sim d_b}[\phi_b \# d_{in}] = \sum_{i=1}^N \pi^{(i)} \sum_{b \in \{0,1\}^n} \theta^{\sum_{l=1}^n b^{(l)}} (1-\theta)^{n - \sum_{l=1}^n b^{(l)}} \delta_{b \odot c_i}. \tag{29}$$

Analyzing or storing d_{out} becomes quickly intractable for moderately large n due to its exponential growth in support size. Therefore, we require an efficient compression procedure to approximate p_{out} with a tractable distribution.

Given a discrete input distribution d_{in}, one could apply the approach from Algorithm 3 to compress d_{out} into a discrete distribution $\bar{d}_{out} \in \mathcal{D}_M(\mathbb{R}^n)$, where $N \leq M < N \cdot 2^n$. This would involve clustering the support of d_{out} into M groups using M-means, applying moment matching within each cluster, and solving a discrete optimal transport problem to compute the 2-Wasserstein error. However, this approach is computationally demanding when the number of components is large.

Instead, we exploit the structure of the Bernoulli mixture to design a more efficient compression strategy that admits a closed-form upper bound on the 2-Wasserstein distance. Specifically, given d_{in}, we select the $\frac{M-N}{2^n-1}$ components c_i with the largest norms and collect their indices into a set $\mathcal{I}$ using a simple sorting algorithm. We then define the compressed distribution as:

$$\bar{d}_{out} = \sum_{i \in \mathcal{I}^c} \pi^{(i)} \delta_{c_i} + \sum_{i \in \mathcal{I}} \pi^{(i)} \sum_{b \in \{0,1\}^n} \theta^{\sum_{l=1}^n b^{(l)}} (1-\theta)^{n - \sum_{l=1}^n b^{(l)}} \delta_{b \odot c_i} \tag{30}$$

where $\mathcal{I}^c = \{1, \ldots, N\} \setminus \mathcal{I}$. That is, we apply dropout only to the inputs in $\mathcal{I}$, and leave the others unperturbed. The resulting distribution $\bar{d}_{out}$ has support size

$$|\mathcal{I}^c| + |\mathcal{I}| \cdot 2^n = \left(N - \frac{M-N}{2^n-1}\right) + \frac{M-N}{2^n-1} \cdot 2^n = M.$$

The following proposition provides a closed-form upper bound on the 2-Wasserstein distance between d_{out} and $\bar{d}_{out}$.

Proposition 33 *Let $d_{out} \in \mathcal{D}_{N\cdot 2^n}(\mathbb{R}^n)$ and $\bar{d}_{out} \in \mathcal{D}_M(\mathbb{R}^n)$ be as defined in Eqn (29) and Eqn (30), respectively. Then,*

$$\mathbb{W}_2^2(d_{out}, \bar{d}_{out}) \leq \sum_{i\in\mathcal{I}^c}(1-\theta)\|c_i\|^2$$

Proof Note that we can write

$$d_{out} = \sum_{i=1}^{N} \pi^{(i)} p_i,$$

where $p_i = \sum_{b\in\{0,1\}^n} \theta^{\sum_{l=1}^n b^{(l)}}(1-\theta)^{n-\sum_{l=1}^n b^{(l)}}\delta_{b\odot c_i}$, and

$$\bar{d}_{out} = \sum_{i\in\mathcal{I}} \pi^{(i)} p_i + \sum_{i\in\mathcal{I}^c} \pi^{(i)}\delta_{c_i}.$$

Hence, according to Lemma 31,

$$\mathbb{W}_2^2(d_{out}, \bar{d}_{out}) = \sum_{i\in\mathcal{I}} \pi^{(i)}\mathbb{W}_2^2\left(p_i, p_i\right) + \sum_{i\in\mathcal{I}^c} \pi^{(i)}\mathbb{W}_2^2\left(p_i, \delta_{c_i}\right) = \sum_{i\in\mathcal{I}^c} \pi^{(i)}\mathbb{W}_2^2\left(p_i, \delta_{c_i}\right)$$

To compute $\mathbb{W}_2^2\left(p_i, \delta_{c_i}\right)$, we observe that the optimal transport plan moves all probability mass from p_i to c_i. Thus,

$$\mathbb{W}_2^2\left(p_i, \delta_{c_i}\right) = \mathbb{E}_{b\sim d_b}[\|b\odot c_i - c_i\|^2] = \mathbb{E}_{b\sim d_b}\left[\sum_{j=1}^{n}\left(b^{(j)}c_i^{(j)} - c_i^{(j)}\right)^2\right] = \sum_{j=1}^{n} {c_i^{(j)}}^2 \mathbb{E}_{\beta\sim p_\beta}\left[(\beta-1)^2\right],$$

where $p_\beta = \theta\delta_1 + (1-\theta)\delta_0$. Thus

$$\mathbb{E}_{\beta\sim p_\beta}\left[(\beta-1)^2\right] = 1 - \mathbb{E}_{\beta\sim p_\beta}\left[\beta^2 - 2\beta\right] = 1 - \mathbb{E}_{\beta\sim p_\beta}\left[\beta\right] = 1-\theta.$$

So, $\mathbb{W}_2^2\left(p_i, \delta_{c_i}\right) = (1-\theta)\|c_i\|^2$, which completes the proof. ■

Appendix F. Results on Signatures of Gaussian Mixture Distributions

Here, we experimentally evaluate the effectiveness of Algorithm 2 in constructing signatures on Gaussian mixtures. Specifically, we examine the conservatism of the formal bound on the 2-Wasserstein distance resulting from the signature operation provided by Algorithm 2, and analyze the restrictiveness of having the signature locations of each component of the mixture on a grid as in Algorithm 2.

In Figure 11, we analyze how the approximation error from the signature of a Gaussian mixture with M components, obtained according to Algorithm 2, changes with increasing signature size. The plots show that as the signature size increases, the approximation error, measured as the 2-Wasserstein distance, decreases uniformly. Furthermore, in line with Corollary 11, the formal upper bound on $\mathbb{W}_2$ is exact for Gaussians ($M = 1$). For Gaussian mixtures ($M > 1$), the conservatism introduced by the formal bounds grows approximately linearly with increasing M. This is because, in Algorithm 1, we bound the Wasserstein distance resulting from the signature operation on the GMMs by the weighted sum of the 2-Wasserstein distance between each Gaussian component in the mixture and their signatures (see line 7 in Algorithm 2 and Eqn 11).

Recall from Subsection 5.2 that to obtain a closed-form bound on the approximation error in Algorithm 2, we place the signature locations of each component of the mixture on a grid

in the transformed space induced by the element's covariance matrix, as illustrated in Figure 4. Specifically, following Proposition 23, we choose the grids such that for a given signature size N, the 2-Wasserstein distance from the signature operation is minimized. In Figure 5b, we estimate the optimality gap if we could instead place the signatures freely to minimize the 2-Wasserstein distance. Perhaps surprisingly, the plots show that placing the signatures freely only slightly reduces the 2-Wasserstein distance. Thus, although optimal signature locations are not grid-aligned (in line with literature, Graf and Luschgy, 2000), for the Gaussian mixtures with distinct modes considered here the optimally chosen grid closely approximates the unconstrained optimal locations, indicating that component-wise grids can be near-optimal for such mixtures.

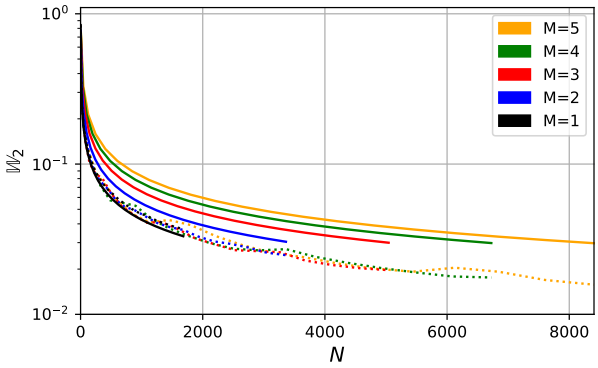


Figure 11: Empirical estimates (dashed lines) and formal bounds (solid lines) on $\mathbb{W}_2$ between a 2D Gaussian mixture distribution with M components and the signature with N grid-constrained locations obtained via Algorithm 2. Formal bounds are computed with Algorithm 2. Empirical estimates are computed from 10^4 samples from each distribution and computing the 2-Wasserstein distance between the resulting empirical distributions.

References

Steven Adams, Andrea Patane, Morteza Lahijanian, and Luca Laurenti. Bnn-dp: robustness certification of bayesian neural networks via dynamic programming. In *International Conference on Machine Learning*, pages 133–151. PMLR, 2023.

Robert J Adler and Jonathan E Taylor. *Random fields and geometry*. Springer Science & Business Media, 2009.

Joseph M Antognini. Finite size corrections for neural network gaussian processes. *arXiv preprint arXiv:1908.10030*, 2019.

Nicola Apollonio, Daniela De Canditiis, Giovanni Franzina, Paola Stolfi, and Giovanni Luca Torrisi. Normal approximation of random gaussian neural networks. *Stochastic Systems*, 15(1):88–110, 2025.

Krishnakumar Balasubramanian, Larry Goldstein, Nathan Ross, and Adil Salim. Gaussian random field approximation via stein's method with applications to wide random neural networks. *Applied and Computational Harmonic Analysis*, 72:101668, 2024.

Andrea Basteri and Dario Trevisan. Quantitative gaussian approximation of randomly initialized deep neural networks. *Machine Learning*, pages 1–21, 2024.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

Alberto Bordino, Stefano Favaro, and Sandra Fortini. Non-asymptotic approximations of gaussian neural networks via second-order poincaré inequalities. *arXiv preprint arXiv:2304.04010*, 2023.

Luca Bortolussi, Ginevra Carbone, Luca Laurenti, Andrea Patane, Guido Sanguinetti, and Matthew Wicker. On the robustness of bayesian neural networks to adversarial attacks. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Daniele Bracale, Stefano Favaro, Sandra Fortini, Stefano Peluchetti, et al. Large-width functional asymptotics for deep gaussian neural networks. In *International Conference on Learning Representations*, 2021.

James V Burke, Frank E Curtis, Adrian S Lewis, Michael L Overton, and Lucas EA Simões. Gradient sampling methods for nonsmooth optimization. *Numerical nonsmooth optimization: State of the art algorithms*, pages 201–225, 2020.

Valentina Cammarota, Domenico Marinucci, Michele Salvi, and Stefano Vigogna. A quantitative functional central limit theorem for shallow neural networks. *Modern Stochastics: Theory and Applications*, 11:85–108, 2023.

Guillermo Canas and Lorenzo Rosasco. Learning probability measures with respect to optimal transport metrics. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

Timothy A Davis, Sivasankaran Rajamanickam, and Wissam M Sid-Lakhdar. A survey of direct methods for sparse linear systems. *Acta Numerica*, 25:383–566, 2016.

Julie Delon and Agnes Desolneux. A wasserstein-type distance in the space of gaussian mixture models. *SIAM Journal on Imaging Sciences*, 13(2):936–970, 2020.

David Duvenaud, Oren Rippel, Ryan Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, pages 202–210. PMLR, 2014.

Ethan Dyer and Guy Gur-Ari. Asymptotics of wide networks from feynman diagrams. In *International Conference on Learning Representations*, 2020.

Ronen Eldan, Dan Mikulincer, and Tselil Schramm. Non-asymptotic approximations of neural networks by gaussian processes. In *Conference on Learning Theory*, pages 1754–1775. PMLR, 2021.

Stefano Favaro, Boris Hanin, Domenico Marinucci, Ivan Nourdin, and Giovanni Peccati. Quantitative clts in deep neural networks. *Probability Theory and Related Fields*, 191(3):933–977, 2025.

Daniel Flam-Shepherd, James Requeima, and David Duvenaud. Mapping gaussian process priors to bayesian neural networks. In *NIPS Bayesian deep learning workshop*, volume 3, 2017.

Daniel Flam-Shepherd, James Requeima, and David Duvenaud. Characterizing and warping the function space of bayesian neural networks. In *NeurIPS Workshop on Bayesian Deep Learning*, page 18, 2018.

Vincent Fortuin. Priors in bayesian deep learning: A review. *International Statistical Review*, 90 (3):563–591, 2022.

Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability theory and related fields*, 162(3):707–738, 2015.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.

Adrià Garriga-Alonso and Mark van der Wilk. Correlated weights in infinite limits of deep convolutional neural networks. In *Uncertainty in Artificial Intelligence*, pages 1998–2007. PMLR, 2021.

Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow gaussian processes. In *International Conference on Learning Representations*, 2019.

Alan Genz and Frank Bretz. *Computation of multivariate normal and t probabilities*, volume 195. Springer Science & Business Media, 2009.

Alison L Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International statistical review*, 70(3):419–435, 2002.

Clark R Givens and Rae Michael Shortt. A class of wasserstein metrics for probability distributions. *Michigan Mathematical Journal*, 31(2):231–240, 1984.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110:393–416, 2021.

Siegfried Graf and Harald Luschgy. *Foundations of quantization for probability distributions*. Springer Science & Business Media, 2000.

Boris Hanin. Random neural networks in the infinite width limit as gaussian processes. *The Annals of Applied Probability*, 33(6A):4798–4819, 2023.

Tamir Hazan and Tommi Jaakkola. Steps toward deep kernel methods from infinite neural networks. *arXiv preprint arXiv:1508.05133*, 2015.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869. PMLR, 2015.

John R Hershey and Peder A Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–317. IEEE, 2007.

Alexander Immer, Matthias Bauer, Vincent Fortuin, Gunnar Rätsch, and Khan Mohammad Emtiyaz. Scalable marginal likelihood estimation for model selection in deep learning. In *International Conference on Machine Learning*, pages 4563–4573. PMLR, 2021.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Simon J Julier and Jeffrey K Uhlmann. Unscented filtering and nonlinear estimation. *IEEE*, 92 (3):401–422, 2004.

Muhammad Firmansyah Kasim. Derivatives of partial eigendecomposition of a real symmetric matrix for degenerate cases. *arXiv preprint arXiv:2011.04366*, 2020.

Mohammad Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *International Conference on Machine Learning*, pages 2611–2620. PMLR, 2018.

Mohammad Emtiyaz E Khan, Alexander Immer, Ehsan Abedi, and Maciej Korzepa. Approximate inference turns deep networks into gaussian processes. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

J Kieffer. Exponential rate of convergence for lloyd's method i. *IEEE Transactions on Information Theory*, 28(2):205–210, 1982.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Adam Klukowski. Rate of convergence of polynomial networks to gaussian processes. In *Conference on Learning Theory*, pages 701–722. PMLR, 2022.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

Christos Louizos, Karen Ullrich, and Max Welling. Bayesian compression for deep learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

David JC MacKay. Bayesian non-linear modeling for the prediction competition. In *Maximum Entropy and Bayesian Methods: Santa Barbara, California, USA, 1993*, pages 221–234. Springer, 1996.

Jan R Magnus and Heinz Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019.

Marko Mäkelä. Survey of bundle methods for nonsmooth optimization. *Optimization methods and software*, 17(1):1–29, 2002.

BG Manjunath and Stefan Wilhelm. Moments calculation for the doubly truncated multivariate normal density. *Journal of Behavioral Data Science*, 1(1):17–33, 2021.

Takuo Matsubara, Chris J Oates, and François-Xavier Briol. The ridgelet prior: A covariance function approach to prior specification for bayesian neural networks. *The Journal of Machine Learning Research*, 22(1):7045–7101, 2021.

Alexander G de G Matthews, Mark Rowland, Jiri Hron, Richard E Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018.

Geoffrey J McLachlan and David Peel. *Finite mixture models*, volume 299. John Wiley & Sons, 2000.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Jiri Hron, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. In *International Conference on Learning Representations*, 2018.

Gilles Pagès and Jacques Printems. Optimal quadratic quantization for numerics: the gaussian case. *Monte Carlo Methods Appl.*, 9(2):135–165, 2003.

Gilles Pages and Benedikt Wilbertz. Optimal delaunay and voronoi quantization schemes for pricing american style options. In *Numerical Methods in Finance: Bordeaux, June 2010*, pages 171–213. Springer, 2012.

Victor M Panaretos and Yoav Zemel. Statistical aspects of wasserstein distances. *Annual review of statistics and its application*, 6(1):405–431, 2019.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

R. Tyrrell Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, Princeton, NJ, revised edition edition, 2015. ISBN 978-0691015866.

Samuel S Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. In *International Conference on Learning Representations*, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Charlie Tang and Russ R Salakhutdinov. Learning stochastic feedforward neural networks. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

Ba-Hien Tran, Simone Rossi, Dimitrios Milios, and Maurizio Filippone. All you need is a good functional prior for bayesian deep learning. *Journal of Machine Learning Research*, 23(74):1–56, 2022.

Volker Tresp. Mixtures of gaussian processes. In *Advances in Neural Information Processing Systems*, volume 13, 2000.

Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, pages 9690–9700. PMLR, 2020.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Matthew Wicker, Luca Laurenti, Andrea Patanè, and Marta Kwiatkowska. Probabilistic safety for Bayesian neural networks. In *Uncertainty in Artificial Intelligence*. PMLR, 2020.

Christopher Williams. Computing with infinite networks. In *Advances in Neural Information Processing Systems*, volume 9, 1996.

Sho Yaida. Non-gaussian processes and neural networks at finite widths. In *Mathematical and Scientific Machine Learning*, pages 165–192. PMLR, 2020.

Tianyuan Yu, Yongxin Yang, Da Li, Timothy Hospedales, and Tao Xiang. Simple and effective stochastic neural networks. In *AAAI Conference on Artificial Intelligence*, volume 35, pages 3252–3260, 2021.

Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5852–5861. PMLR, 2018.