

QFT - NN correspondence

EKATERINA S. IVSHINA¹

¹*Department of Mathematics, Princeton University, Princeton, NJ 08544, USA*

ABSTRACT

In this paper, we review the recent developments in establishing the correspondence between Quantum Field Theory (QFT) and Neural Networks (NNs), pioneered by [Halverson et al. \(2021\)](#). We first introduce the concept of neural networks and then discuss the NN-QFT correspondence. The correspondence consists of two main ingredients developed in QFT: Effective Field Theory and Renormalization Group flow. The correspondence relies on the observation that many asymptotic neural networks are drawn from Gaussian Processes (GPs), the analog of non-interacting field theories. Moving away from the asymptotic limit yields a non-Gaussian process and corresponds to introducing particle interactions, allowing for the computation of correlation functions of neural network outputs with Feynman diagrams. Wilsonian renormalization group flow allows for determining the most relevant non-Gaussian terms. The formalism is valid for any architecture that becomes a GP in an asymptotic limit, a property preserved under certain types of training.

1. INTRODUCTION

In this paper, we discuss the NN-QFT correspondence and its experimental verification proposed in [Halverson et al. \(2021\)](#). We note that [Halverson et al. \(2021\)](#) experiments involve only randomly initialized networks, rather than trained networks, which means understanding of learning is left to future work and we discuss possible approaches in Section 7. The establishment of the correspondence was inspired by an observation that randomly initialized infinite width NNs are GPs ([Yang 2019a, 2020a,b](#)). The primary goal of the work was to develop EFT techniques for treating the non-gaussian processes (NGPs) associated with neural networks. Furthermore, the NN-QFT correspondence is hypothesized to persist under neural network training, given that it has been previously shown in the literature that the GP property persists under appropriate training ([Yang 2019a, 2020b](#); [Yang & Hu 2020](#); [Lee et al. 2020](#); [Jacot et al. 2018](#)). The latter point is crucial for making the correspondence useful in real-world applications.

This paper is organized as follows. Section 2 surveys the basics of neural networks. Section 3 discusses the theoretical results relating the behavior of neural networks in the infinite-width limit to Gaussian Processes. In Section 4, we apply the techniques of

Effective Field Theory to analyzing neural networks and their associated non-gaussian processes. Section 5 discusses how Renormalization Group Flow can further shed light on the behavior of neural networks. Section 6 discusses the experimental verification of the NN-QFT correspondence. In Section 7, we suggest possible future research directions. Section 8 summarizes the main aspects of the NN-QFT correspondence.

2. NEURAL NETWORKS

Machine learning is a branch of artificial intelligence and computer science which focuses on the use of data and algorithms to emulate the way that humans learn. Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering new insights in different applications. Neural networks is a sub-field of machine learning. The reason why neural networks attract great interest and are successfully applied in many different contexts nowadays lies in the *universality* results: it has been shown that neural networks can be used to approximate any continuous function to an arbitrary accuracy when the depth of the neural network is large enough (Hornik et al. 1989; Zhou 2018; Kratsios 2021). Although different types of machine learning exist (e.g. supervised (Kotsiantis 2007), unsupervised (Alloghani et al. 2020), and reinforcement learning (Arulkumaran et al. 2017)), in this paper we focus on supervised learning with neural networks.

First, let us briefly review the definition of neural networks. A neural network is a function $f_\theta : \mathbb{R}^{d_{in}} \rightarrow \mathbb{R}^{d_{out}}$ with "learnable" parameters θ . A general setup in supervised machine learning is as follows. We start with collecting some data (for example, thousands of images of handwritten digits from 0 to 9). Note that in modern applications of machine learning, collecting data tends to be the step that requires the most time and attention because creating a balanced, unbiased dataset is key to ensuring that the trained neural network performs well (Buda et al. 2018). After collecting the data, we assign a "label" to each sample (in the case of a classification task) or associate a scalar value/vector to each sample that we would like our neural network to learn to predict. In the example with images of digits, each sample is an image and the corresponding label is the digit displayed. This way, we create a *training dataset*.

At this point, we must decide on the architecture of the neural network; this decision depends on the problem in question (Philipp 2021). The two most famous types of architectures are feed-forward (Baldi & Vershynin 2019) and convolutional (LeCun & Bengio 1998). Feed-forward neural networks are most suitable for problems that involve 1D vector inputs and 1D vector outputs. Feed-forward neural networks have simple architectures: they repeatedly compose affine transformations followed by non-linearities at each layer (we discuss a simple example below). Convolutional neural networks, on the other hand, have achieved great success in Computer Vi-

sion applications and are suitable for problems that involve imagery data (LeCun & Bengio 1998).

After defining the architecture, we then would like our neural network to learn to extract patterns from the dataset to be able to make predictions on new data (for example, to identify digits in new images). To achieve this goal, we define a *loss function*, which is a function that quantifies the "distance" (however it is defined) between the neural network outputs and the labels (Janocha & Czarnecki 2017). In addition to quantifying the distance between predictions and labels, the loss function may encode information that allows to keep the neural network from overparametrization or other unwanted behavior (e.g., see Li et al. (2022); Du & Lee (2018)).

Next, we choose a training algorithm (e.g., stochastic gradient descent (Ruder 2017)) and train the network: We first randomly initialize the neural network, i.e. we randomly pick values for each of the learnable parameters θ . Then, the neural network f_θ is evaluated on our dataset, producing an output for each input. Depending on how far the predictions are from the labels (which is quantified by the loss function), the training algorithm updates θ parameters to minimize the loss function. There is no strict rule for how long the training process needs to be (i.e. for how many times the learnable parameters θ need to be updated before the loss function is close to its minimum, assuming the minimum exists). The question of convergence in training is still widely researched nowadays (Skorski et al. 2020; Min et al. 2021). After the training has been completed, we can apply our neural network with the learned parameters θ to new data.

For a concrete and simple example, let us consider a feed-forward neural network with one hidden layer. Such neural network is defined by $f(x) = W_1(\sigma(W_0x + b_0)) + b_1$, where the weights and biases, W_i and b_i , characterize the affine transformations for each layer, and σ is an element-wise non-linearity. This notation is somewhat abstract, but the basic point is simple: we repeatedly compose affine transformations followed by non-linearities at each layer. We explain concretely what this means in Equations 2 and 3 below. Including the spaces associated with the hidden layers, we can describe a 1-layer feed-forward neural network as the following sequence of transformations:

$$f_{\theta,N} : \mathbb{R}^{d_{in}} \xrightarrow{W_0, b_0} \mathbb{R}^N \xrightarrow{\sigma} \mathbb{R}^N \xrightarrow{W_1, b_1} \mathbb{R}^{d_{out}}. \quad (1)$$

Note that here we introduce N in our notation for the neural network $f_{\theta,N}$; this parameter specifies the number of "units" at the hidden layer in our 1-layer feed-forward neural network (see Figure 1). While in this paper we are considering the simplest feed-forward neural network with just one hidden layer, in practice a neural network consists of many layers, i.e. we repeatedly compose affine transformations followed by non-linearities at each layer.

For initializing the 1-layer feed-forward neural network, the weight and bias parameters, collectively labeled as θ , are i.i.d. and drawn from a Gaussian distribution. In this work, the biases are drawn from $\mathcal{N}(\mu_b, \sigma_b^2)$ and the weights in each layer W_0 ,

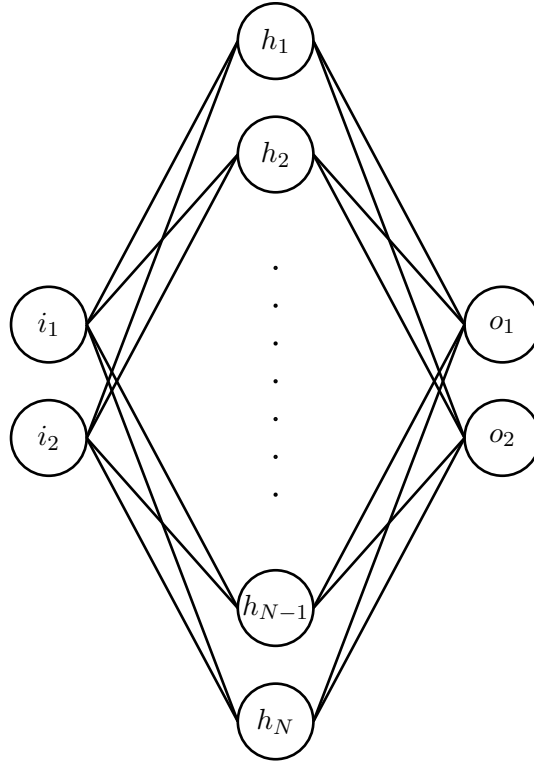


Figure 1. Single-layer fully-connected network of width N , with $d_{in} = d_{out} = 2$. Arrows represent the linear layers of weights and biases, and the nonlinear activation function is applied element-wise to each node ("unit") in the hidden layer.

W_1 are drawn from $\mathcal{N}(\mu_W, \sigma_W^2/d_{in})$ and $\mathcal{N}(\mu_W, \sigma_W^2/N)$, respectively, i.e. the weights are normalized with respect to the input dimension of the associated layer. The first linear layer takes the input x to a *pre-activation* z_0

$$z_0^j = \sum_{i=1} W_0^{ij} x^i + b_0^j, \quad (2)$$

which is then acted on by the elementwise nonlinearity σ , giving a *post-activation* $x_1^j = \sigma(z_0^j)$, that is acted on by the final linear layer, yielding

$$f_{\theta, N}(x) = z_1^k = \sum_{j=1}^N W_1^{jk} x_1^j + b_1^k, \quad (3)$$

the output of the neural network. Note that the type of non-linearity that we apply at each layer can be chosen; examples of commonly used non-linearities include ReLU, tanh, and sigmoid (Nwankpa et al. 2018).

3. INFINITE-WIDTH NEURAL NETWORKS AND GAUSSIAN PROCESSES

By the Central Limit Theorem (CLT), in the infinite width limit the network outputs are drawn from a Gaussian distribution on function space (Neal 1995), i.e. the network outputs are drawn from a Gaussian Process (GP). This follows from the observation that the weight part of the output layer of the network defined in Equation 3 is the

sum of N i.i.d. random variables (we remind the reader that this paper focuses on understanding the behavior of neural networks at the initialization stage as opposed to during/after training). In the infinite width limit $N \rightarrow \infty$, we obtain a finite sum over these terms. Thus by the CLT we have that the neural network output is drawn from a Gaussian distribution, i.e. the neural network evaluated on any finite collection of samples is drawn from a multivariate Gaussian distribution.

Note that since neural networks output real scalars/vectors, it is natural to translate them to scalar fields, which correspond to spin-0 particles, in the NN-QFT correspondence. Let us now review the concept of correlation functions which will help us understand and verify the QFT-NN correspondence.

Consider a neural network with learnable parameters θ ,

$$f_\theta : \mathbb{R}^{d_{in}} \rightarrow \mathbb{R}^{d_{out}}, \quad (4)$$

where at initialization $\theta \sim P(\theta)$. The parameter distribution $P(\theta)$ and the network architecture induce a distribution on function space from which the neural network is drawn, $P(f)$. Later in the text, we may drop the subscript θ and write f instead of f_θ .

For many types of neural networks there is a limit $N \rightarrow \infty$ in which the distribution on functions becomes a Gaussian process (Yang 2019b), i.e. the neural network outputs $\{f(x_1), \dots, f(x_k)\}$, evaluated on a fixed set of k inputs $\{x_1, \dots, x_k\}$, are drawn from a multivariate Gaussian distribution (μ, Ξ^{-1}) ,

$$\{f(x_1), \dots, f(x_k)\} \sim (\mu, \Xi^{-1}), \quad (5)$$

By assumption of Halverson et al. (2021), $\mu = 0$. The covariance matrix Ξ is determined by the kernel function $K(x, x')$ as $(\Xi^{-1})_{ij} = K_{ij} := K(x_i, x_j)$. Since $\mu = 0$, the GP is determined by its covariance.

Correlation functions between n outputs (a.k.a. n -pt functions) are defined as

$$G^{(n)}(x_1, \dots, x_n) = \frac{\int df f_1 \dots f_n e^{-\frac{1}{2} f_i \Xi_{ij} f_j}}{Z}, \quad (6)$$

where the partition function is $Z = \int df e^{-S}$, and $S = -\frac{1}{2} f_i \Xi_{ij} f_j$ is the log-likelihood (action in physics). Einstein summation convention is assumed, and $f_i := f(x_i)$ is a vector of outputs on a fixed set of inputs $\{x_i\}$ with dimension $d_{in} = d$.

In the continuous limit, the correlation functions become

$$G^{(n)}(x_1, \dots, x_n) = \frac{\int df f(x_1) \dots f(x_n) e^{-S}}{Z}, \quad (7)$$

where the log-likelihood is

$$S = \frac{1}{2} \int d^{d_{in}} x d^{d_{in}} x' f(x) \Xi(x, x') f(x') \quad (8)$$

and $\Xi(x, x') = K^{-1}(x, x')$ is the inverse covariance function, defined by

$$\int d^{d_{\text{in}}} x' K(x, x') \Xi(x', x'') = \delta^{(d_{\text{in}})}(x - x''), \quad (9)$$

where $\delta^{(d_{\text{in}})}(x - x'')$ is the d_{in} -dimensional Dirac delta function. This equation is the continuum analog of the relation $(\Xi^{-1})_{ij} = K_{ij}$ in the discrete case.

Both the discrete and continuum versions of the GP n -pt functions may be computed exactly using standard Gaussian integral techniques.

A direct physics analog of a Gaussian process is a free field theory $\phi(x)$, whose path integral is

$$Z = \int D\phi e^{-S[\phi]} \quad (10)$$

where $S[\phi]$ is the free scalar theory action, which is quadratic in ϕ . For example, $S[\phi]$ can be

$$S[\phi] = \int d^d x \phi(x) (\square + m^2) \phi(x), \quad (11)$$

with $\square := \partial_\mu \partial^\mu$ and m being the mass of the bosonic particle associated to ϕ . The functional inverse of $(\square + m^2)$ is the *propagator*, the 2-pt correlation function in the free field theory, and is the analog of the GP kernel.

3.1. Neural Network Correlation Functions with Feynman Diagrams

As we develop the NN-QFT correspondence throughout the paper, it is useful to discuss a diagrammatic approach to computation of correlation functions in neural networks. We derive Feynman rules in this section.

Consider the following partition function of a Gaussian process with source $J(x)$

$$Z_{GP}[J] = \frac{\int df e^{-S_{GP} - \frac{1}{2} \int d^{d_{\text{in}}} x J(x) f(x) - \frac{1}{2} \int d^{d_{\text{in}}} y J(y) f(y)}}{Z_{GP,0}}, \quad (12)$$

where $Z_{GP,0} := \int df e^{-S_{GP}}$, and S_{GP} is the action, or (negative) log-likelihood,

$$S_{GP} = \frac{1}{2} \int d^{d_{\text{in}}} x d^{d_{\text{in}}} y f(x) \Xi(x, y) f(y). \quad (13)$$

The n -pt correlation function is defined by

$$G_{GP}^{(n)}(x_1, \dots, x_n) = \frac{\int df f(x_1) \dots f(x_n) e^{-S_{GP}}}{Z_{GP,0}}, \quad (14)$$

By performing the Gaussian integral, we obtain

$$Z_{GP}[J] = \exp \left(\frac{1}{2} \int d^{d_{\text{in}}} x d^{d_{\text{in}}} y J(x) K(x, y) J(y) \right). \quad (15)$$

GP / asymptotic NN	Free QFT
input x	external space or momentum space point
kernel $K(x_1, x_2)$	Feynman propagator
asymptotic NN $f(x)$	free field
log-likelihood	free action S_{GP}

Table 1. Correspondence between GP / asymptotic neural network and free QFT

To compute the n -point correlation function, we use Wick’s theorem (Ferialdi & Diósi 2021):

$$G_{GP}^{(n)}(x_1, \dots, x_n) = \sum_{p \in Wick(x_1, \dots, x_n)} K(a_1, b_1) \dots K(a_{n/2}, b_{n/2}) \quad (16)$$

where we write each element $p \in (x_1, \dots, x_n)$ as $p = (a_1, b_1), \dots, (a_{n/2}, b_{n/2})$ and we sum over all possible Wick contractions:

$$Wick(x_1, \dots, x_n) = \{P \in \text{Partitions}(x_1, \dots, x_n) \mid |p| = 2 \forall p \in P\}. \quad (17)$$

In other words, to compute $G_{GP}^{(n)}(x_1, \dots, x_n)$, we sum over all ways of pairing up elements in $\{x_1, \dots, x_n\}$, and in each term write a kernel factor $K(a_i, b_i)$ for each of the pairs (a_i, b_i) . Translating this into Feynman rules, we sum over all ways of connecting the points $\{x_1, \dots, x_n\}$ in pairs, and in each term draw a line between the points in the pair (a_i, b_i) . To go from Feynman diagrams to analytic expressions, write a factor of $K(a_i, b_i)$ for each line in the diagram connecting a_i and b_i .

For instance, in the case when $n = 4$, we have

$$G_{GP}^{(4)}(x_1, x_2, x_3, x_4) = K(x_1, x_2)K(x_3, x_4) + K(x_1, x_3)K(x_2, x_4) + K(x_1, x_4)K(x_2, x_3)$$

$$= \begin{array}{c} x_1 \\ | \\ x_2 \end{array} \begin{array}{c} x_3 \\ | \\ x_4 \end{array} + \begin{array}{c} x_1 \quad x_3 \\ \text{---} \quad \text{---} \\ x_2 \quad x_4 \end{array} + \begin{array}{c} x_1 \quad x_3 \\ \diagdown \quad \diagup \\ x_2 \quad x_4 \end{array}$$

Note that $G_{GP}^{(n)}(x_1, \dots, x_n) = 0$ for any odd n .

The above discussion shows that asymptotic neural networks are analogous to free field theories. The precise correspondence is summarized in Table 1. Remembering that the GP can often be realized by asymptotic neural networks, in this analogy the neural network inputs are translated to points in space, and the kernel is the Feynman propagator, which represents the amplitude of propagation of a particle from one point to another. While in this analogy asymptotic neural networks do not give rise to interactions in QFT, we will show in Section 4 that the interactions do arise when Z_{GP} is corrected by non-Gaussian terms.

We note that in this correspondence, function space perspective is taken as opposed to the usual parameter space perspective, i.e. here, initializing neural networks is viewed as drawing functions (neural networks) from a particular function space distribution.

4. NON-GAUSSIAN PROCESSES: FINITE WIDTH NEURAL NETWORKS

In this Section, we explain how to use Effective Field Theory (EFT) to analyze neural network architectures and their associated NGPs. The key idea is as follows: the action S_{GP} , associated to the asymptomatic behavior of a neural net as $N \rightarrow \infty$, does not suffice to compute correlation functions $G^{(n)}$ of finite-width neural networks. EFT allows to determine ΔS correction to the GP log-likelihood, giving an effective log-likelihood for the NGP associated to the finite-width neural networks,

$$S = S_{GP} + \Delta S \quad (18)$$

In the QFT-NN correspondence, Effective Field Theory techniques are applied to achieve the following goals:

- Give a candidate ΔS for the NGP.
- Fix coefficients of operators in ΔS by conducting experiments.
- Once the coefficients are fixed, make predictions for other experiments and verify them.

The QFT-NN correspondence in the case of finite-width neural networks is summarized in Table 2. We identify the NN input $x \in \mathbb{R}^{d_{in}}$ with a point in space or momentum space in field theory, the NN kernel with free or exact propagator in QFT (whether the propagator is free or exact depends on what we obtain in the infinite width limit). The NN output $f(x)$ and log-likelihood now correspond to the interacting field and the EFT action, respectively.

In EFT, one organizes the study of a physical system according to length or momentum scale; with neural networks, we organize our analysis according to input scale. Let us perform some dimensional analysis. S_{GP} must be dimensionless; it scales as x^0 , and we write $[S_{GP}] = 0$. Since $d^{d_{in}}x$ scales as $x^{d_{in}}$ it has input dimensions of d_{in} , $[d^{d_{in}}x] = d_{in}$, and $[d^{d_{in}}y] = d_{in}$. The condition $[S] = 0$ then gives a relation between dimensions of f and Ξ , $[S] = 2d_{in} + 2[f] + [\Xi] = 0$, which determines the classical scaling dimension of the neural network f ,

$$[f] = -\frac{2d_{in} + [\Xi]}{2}. \quad (19)$$

This can then be used to determine the dimensions of the coefficients of operators that might appear in ΔS . For instance, consider operators

$$\mathcal{O}_k := g_k f(x)^k \quad (20)$$

appearing in ΔS as $\int dx_k$. Then $[\Delta S] = 0$ requires $d_{in} + k[f] + [g_k] = 0$ and we have

$$[g_k] = -d_{in} + \frac{k(2d_{in} + [\Xi])}{2}. \quad (21)$$

NGP / finite NN	Interacting QFT
input x	external space or momentum space point
kernel $K(x_1, x_2)$	free or exact propagator
network output $f(x)$	interacting field
non-Gaussianities	interactions
non-Gaussian coefficients	coupling strengths
log probability	effective action S

Table 2. Correspondence between quantities in the NGP / finite-width neural network and QFT. See text for discussion on whether the kernel is the free or exact propagator.

Using Equation (9) and the fact that $[\delta^{(d_{\text{in}})}(x)] = -d_{\text{in}}$ we have $d_{\text{in}} + [\Xi] + [K] = -d_{\text{in}}$, and therefore

$$[g_k] = -d_{\text{in}} - \frac{k [K]}{2}. \quad (22)$$

Most importantly, Wilsonian EFT provides the following rules for constructing the effective action of an NGP associated to a neural network admitting a known GP limit:

- Determine the symmetries (or desired symmetries) respected by the system of interest.
- Fix an upper bound k on the dimension of any operator appearing in ΔS .
- Define ΔS to contain all operators of dimension $\leq k$ that respect the symmetries.

Since the GP limit is known, the NGP is defined by $S = S_{GP} + \Delta S$. This allows to determine correlation functions. By experimentally measuring them, one may fix coefficients of terms in ΔS and make subsequent predictions. The desired symmetries and the choice of a relevant value of k may be constrained by the problem in question.

4.1. Correlation Functions in NGPs with Interacting Feynman Diagrams

We now introduce a method for computing NGP correlation functions. Let us first briefly discuss the basics of cutoffs and perturbation theory.

We start with perturbation theory. Consider an NGP corresponding to a finite-width neural network architecture, with associated effective action $S = S_{GP} + \Delta S$. The GP n -pt correlation function is

$$G^{(n)}(x_1, \dots, x_n) = \frac{\int df f(x_1) \dots f(x_n) e^{-S}}{Z_0}, \quad \text{where } Z_0 = \int df e^{-S}. \quad (23)$$

We can use perturbation theory to approximate n -pt functions if the coefficients of operators in ΔS are sufficiently small. In QFT, this gives rise to interactions. For instance, consider corrections to the n -pt function coming from $\Delta S = \int d^{d_{\text{in}}} x g_k f(x)^k$,

with $k > 2$. Multiplying the numerator and denominator by $1/Z_{GP,0}$ and expanding under the assumption of small g_k , we obtain

$$G^{(n)}(x_1, \dots, x_n) = \frac{\int df f(x_1) \dots f(x_n) [1 - \int d^{d_{\text{in}}}x g_k f(x)^k + O(g_k^2)] e^{-S_{GP}}/Z_{GP,0}}{\int df [1 - \int d^{d_{\text{in}}}x g_k f(x)^k + O(g_k^2)] e^{-S_{GP}}/Z_{GP,0}}. \quad (24)$$

Truncating at a given order in g_k (in this case just the leading correction), we can approximate the n -pt using Wick's theorem and Feynman diagrams.

We now introduce cutoffs. Computing NGP correlation functions via perturbation theory can lead to integrals that are divergent. We can treat the divergence by introducing a cutoff $\Lambda > 0$:

$$S \rightarrow S_\Lambda, \quad (25)$$

where S_Λ differs from S only in the fact that all integrals are bounded from below by $-\Lambda$ and above by Λ . In QFT, introduction of a cutoff is justified by the fact that physical theories have finite ranges of applicability. For example, a model that describes a scattering process done at a given momentum scale can not be valid up to arbitrarily high momenta. In this case, low energy experiments with momenta $|p| \ll \Lambda$ should not depend Λ . This requirement gives rise to the Wilsonian renormalization group equations (RGEs), which are differential equations that impose relations on the coefficients of operators in S . We will discuss this further in Section 5.



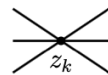

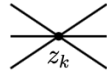

For experimental verification of the QFT-NN correspondence, [Halverson et al. \(2021\)](#) focuses on 1-layer feed-forward neural networks introduced in Section 2. Moreover, we set $d_{\text{out}} = 1$. We may thus consider non-Gaussian terms of the form

$$\Delta S = \int dx [g f(x)^3 + \lambda f(x)^4 + \alpha f(x)^5 + \kappa f(x)^6 + \dots]. \quad (26)$$

However, all odd-point functions vanish, since the mean value of the weights and biases is assumed to be zero. Thus, $g = \alpha = 0$. This result can also be understood with a symmetry argument: randomly initialized neural networks f and $-f$ should be on equal footing, which implies S must be invariant under $f \rightarrow -f$ transformation. Hence, the effective action that we are considering is

$$S = S_{GP} + \int d^{d_{\text{in}}}x [\lambda f(x)^4 + \kappa f(x)^6]. \quad (27)$$

With this effective action for the NGP, we can compute correlation functions in perturbation theory. One may also represent the correlation functions diagrammatically by the Feynman rules summarized below:

- 1) For each of the n external points x_i , draw .
- 2) For each y_j , draw . For each z_k , draw .
- 3) Determine all ways to pair up the loose ends associated to x_i 's, y_j 's, and z_k 's. This will yield some number of topologically distinct diagrams. Draw them with solid lines.
- 4) Write a sum over the diagrams with an appropriate combinatoric factor out front, which is the number of ways to form that diagram. Each diagram corresponds to an analytic term in the sum.
- 5) For each diagram, write $-\int d^{d_{\text{in}}} y_j \lambda$ for each , and $-\int d^{d_{\text{in}}} z_k \kappa$ for each .
- 6) Write $K(u, v)$ for each .
- 7) Throw away any terms containing vacuum bubbles; see Footnote [15](#).

Note that the above discussion and Feynman rules apply only in the case of $S = S_{GP} + \Delta S$, where ΔS is comprised of only non-Gaussian corrections, i.e. S_{GP} is the only Gaussian part of the action. In that case the two-point function is

$$G^{(2)}(x_1, x_2) = K(x_1, x_2) + \lambda\text{- and } \kappa\text{- corrections.} \quad (28)$$

In particular, $K(x_1, x_2)$ is the analog of the free-theory propagator in QFT. In some neural net architectures, however, it may happen that the GP kernel is the exact 2-pt function for the NGP as well

$$G^{(2)}(x_1, x_2) = K(x_1, x_2), \quad (29)$$

in which case the Feynman rules get slightly modified, as discussed in [Halverson et al. \(2021\)](#).

5. WILSONIAN RENORMALIZATION IN NEURAL NET NON-GAUSSIAN PROCESSES

Let us consider a concrete example when we can introduce a cutoff scale. Suppose we have a dataset of grayscale handwritten digits. Let the input space be $\mathbb{R}^{28 \times 28}$, with the real numbers representing brightness: each of 28×28 pixels ranging from $-\infty$ (pure black), 0 (pure grey) to ∞ (pure white). Images in the dataset that are grey correspond to the inputs that are close to zero. Computing NGP correlation functions using perturbation theory, i.e. integrating from pure black to pure white colors, may introduce divergences. We may thus limit the integration to range from some threshold black to white colors, leading to the replacement $\Delta S \rightarrow \Delta S_\Lambda$. These thresholds correspond to a maximum brightness and maximum darkness scale, which should be unimportant because a small change in very high threshold amount of black

and white colors should not affect correlation functions associated with grey images, i.e. inputs near zero. Thus, brightness thresholds in this example are analogous to the cutoffs in EFT.

The above discussion supports a central observation in renormalization theory: all S_Λ with sufficiently high cutoffs should give the same theoretical predictions for experiments:

$$\frac{dG^{(n)}(x_1, \dots, x_n)}{d\Lambda} = 0. \quad (30)$$

The differential equations are known as renormalization group equations (RGEs), which include the β -function associated to the coupling, defined as

$$\beta(g_i) := \frac{dg_i}{d \log \Lambda}, \quad (31)$$

This gives rise to what is known as the Renormalization Group (RG) flow. Theoretical considerations based on RG flow considerations may then be used to argue which couplings are essential in an NGP associated to a given neural network architecture. Non-perturbative renormalization for the QFT-NN correspondence is further discussed in [Erbin et al. \(2021\)](#).

6. EXPERIMENTAL VERIFICATION

One must derive the 2-point function ("kernel") associated to a given infinite width architecture in order to compute correlation functions in the associated GP. Derivations of three types of kernels are done in Appendix B of [Halverson et al. \(2021\)](#). Assuming the couplings are constants, they can then be measured from experiments. To keep this paper manageable, we leave the calculations out of this manuscript. Here, it suffices to say that the NN-QFT correspondence has been verified in the case of a 1-layer feed-forward neural network with three different types of kernels.

7. FUTURE DIRECTIONS

There are a number of research directions that can be pursued within the established NN-QFT correspondence introduced in this paper.

- First, given that the GP property of neural nets in the infinite width limit persists under appropriate training [Yang \(2019b, 2020\)](#); [Yang & Hu \(2020\)](#); [Lee et al. \(2019\)](#); [Jacot et al. \(2018\)](#), the next step is to further study the training process and experimentally verify the NN-QFT correspondence in trained networks.

[Yang \(2019b\)](#) has recently shown that Convolutional, Feed-forward and Recurrent Neural Networks all admit a GP limit. It will be interesting to associate an NGP to a Convolutional Neural Network and experimentally measure the couplings. This would serve as another test of the proposed NN-QFT correspondence.

- The fundamental notion of symmetries that is widely utilized throughout physics can also serve as a guiding principle in designing neural network architectures. Generally, symmetries in neural networks are realized as linear maps $g \in G$ that act on the activations ϕ as $\rho(g)\phi$, where ρ is a representation matrix. Besides translations, other examples of G in ML are rotations (Cohen et al. 2018) and permutations (Maron et al. 2019). Developing a theory of symmetries in neural networks can further shed light on their performance. ’t Hooft’s notion of technical naturality (that the couplings may be small when setting them to zero recovers a symmetry) may serve as a guiding principle to determining whether the coefficient of a non-Gaussian term in the NGP likelihood associated to a given neural network is a constant.
- In addition, quantum computing is believed to influence the field of machine learning in the future. Accordingly, quantum theory-based formalism of deep learning has recently been proposed in Bondesan & Welling (2021), which included a discussion of implementing ML models on a quantum computer.
- How does information propagate through neural networks? The question of neural net architectures is of importance. Why does it need to be linear? There are other widely used architectures that utilize skip connections, filters, etc. However, there is no general theory of NN architectures. It is possible that homotopy group theory can serve as a foundation for the analysis of NN architectures.
- Lastly, arrangement and representation of input data is important and is often overlooked in practice. Understanding the structure of the input space is critical for designing machine learning algorithms and their training schemes. In the NN-QFT correspondence, I would guess that studying the data space would translate to QFT in curved spacetime or more general manifolds.

8. CONCLUSIONS

In this paper, we discussed the applications of various techniques from QFT to the study of neural networks. We introduced perturbation theory, Feynman diagrams, Effective Field Theory and Renormalization Group flow and have shown their usefulness for understanding the behavior of neural networks that become a Gaussian Process in an asymptotic limit.

ACKNOWLEDGMENTS

We thank Wenli Zhao for helpful discussions throughout the semester.

REFERENCES

- | | |
|--|--|
| <p>Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., & Aljaaf, A. J. 2020, A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science, ed. M. W.</p> | <p>Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. 2017, IEEE Signal Processing Magazine, 34, 26–38, doi: 10.1109/msp.2017.2743240</p> |
|--|--|

- Baldi, P., & Vershynin, R. 2019, The capacity of feedforward neural networks.
<https://arxiv.org/abs/1901.00434>
- Bondesan, R., & Welling, M. 2021, The Hinton in your Neural Network: a Quantum Field Theory View of Deep Learning.
<https://arxiv.org/abs/2103.04913>
- Buda, M., Maki, A., & Mazurowski, M. A. 2018, *Neural Networks*, 106, 249–259, doi: [10.1016/j.neunet.2018.07.011](https://doi.org/10.1016/j.neunet.2018.07.011)
- Cohen, T. S., Geiger, M., Koehler, J., & Welling, M. 2018, Spherical CNNs.
<https://arxiv.org/abs/1801.10130>
- Du, S. S., & Lee, J. D. 2018, On the Power of Over-parametrization in Neural Networks with Quadratic Activation.
<https://arxiv.org/abs/1803.01206>
- Erbin, H., Lahoche, V., & Ousmane Samary, D. 2021, arXiv e-prints, arXiv:2108.01403.
<https://arxiv.org/abs/2108.01403>
- Ferialdi, L., & Diósi, L. 2021, *Physical Review A*, 104, doi: [10.1103/physreva.104.052209](https://doi.org/10.1103/physreva.104.052209)
- Halverson, J., Maiti, A., & Stoner, K. 2021, *Machine Learning: Science and Technology*, 2, 035002, doi: [10.1088/2632-2153/abeca3](https://doi.org/10.1088/2632-2153/abeca3)
- Hornik, K., Stinchcombe, M., & White, H. 1989, *Neural Networks*, 2, 359, doi: [https://doi.org/10.1016/0893-6080\(89\)90020-8](https://doi.org/10.1016/0893-6080(89)90020-8)
- Jacot, A., Gabriel, F., & Hongler, C. 2018, arXiv e-prints, arXiv:1806.07572.
<https://arxiv.org/abs/1806.07572>
- Jacot, A., Gabriel, F., & Hongler, C. 2018, in *NeurIPS*
- Janocha, K., & Czarnecki, W. M. 2017, On Loss Functions for Deep Neural Networks in Classification.
<https://arxiv.org/abs/1702.05659>
- Kotsiantis, S. B. 2007, in *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real World AI Systems with Applications in EHealth, HCI, Information Retrieval and Pervasive Technologies (NLD: IOS Press)*, 3–24
- Kratsios, A. 2021, *Annals of Mathematics and Artificial Intelligence*, 89, 435–469, doi: [10.1007/s10472-020-09723-1](https://doi.org/10.1007/s10472-020-09723-1)
- LeCun, Y., & Bengio, Y. 1998, *Convolutional Networks for Images, Speech, and Time Series* (Cambridge, MA, USA: MIT Press), 255–258
- Lee, J., Xiao, L., Schoenholz, S. S., et al. 2019, ArXiv, abs/1902.06720
- Lee, J., Xiao, L., Schoenholz, S. S., et al. 2020, *Journal of Statistical Mechanics: Theory and Experiment*, 2020, 124002, doi: [10.1088/1742-5468/abc62b](https://doi.org/10.1088/1742-5468/abc62b)
- Li, M., Zhang, X., Thrampoulidis, C., Chen, J., & Oymak, S. 2022, AutoBalance: Optimized Loss Functions for Imbalanced Data.
<https://arxiv.org/abs/2201.01212>
- Maron, H., Ben-Hamu, H., Shamir, N., & Lipman, Y. 2019, Invariant and Equivariant Graph Networks.
<https://arxiv.org/abs/1812.09902>
- Min, H., Tarmoun, S., Vidal, R., & Mallada, E. 2021, On the Explicit Role of Initialization on the Convergence and Implicit Bias of Overparametrized Linear Networks.
<https://arxiv.org/abs/2105.06351>
- Neal, R. M. 1995, PhD thesis, University of Toronto
- Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. 2018, Activation Functions: Comparison of trends in Practice and Research for Deep Learning.
<https://arxiv.org/abs/1811.03378>
- Philipp, G. 2021, The Nonlinearity Coefficient - A Practical Guide to Neural Architecture Design.
<https://arxiv.org/abs/2105.12210>

- Ruder, S. 2017, An overview of gradient descent optimization algorithms.
<https://arxiv.org/abs/1609.04747>
- Skorski, M., Temperoni, A., & Theobald, M. 2020, Revisiting Initialization of Neural Networks.
<https://arxiv.org/abs/2004.09506>
- Yang, G. 2019a, arXiv e-prints, arXiv:1910.12478.
<https://arxiv.org/abs/1910.12478>
- . 2019b, arXiv e-prints, arXiv:1910.12478.
<https://arxiv.org/abs/1910.12478>
- . 2020a, arXiv e-prints, arXiv:2006.14548.
<https://arxiv.org/abs/2006.14548>
- . 2020b, arXiv e-prints, arXiv:2009.10685.
<https://arxiv.org/abs/2009.10685>
- Yang, G. 2020, Tensor Programs III: Neural Matrix Laws.
<https://arxiv.org/abs/2009.10685>
- Yang, G., & Hu, E. J. 2020, arXiv e-prints, arXiv:2011.14522.
<https://arxiv.org/abs/2011.14522>
- Yang, G., & Hu, E. J. 2020, Feature Learning in Infinite-Width Neural Networks.
<https://arxiv.org/abs/2011.14522>
- Zhou, D.-X. 2018, Universality of Deep Convolutional Neural Networks.
<https://arxiv.org/abs/1805.10769>