BAYESIAN CONVOLUTIONAL NEURAL NETWORKS WITH MANY CHANNELS ARE GAUSSIAN PROCESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

There is a previously identified equivalence between wide fully connected neural networks (FCNs) and Gaussian processes (GPs). This equivalence enables, for instance, test set predictions that would have resulted from a fully Bayesian, infinitely wide trained FCN to be computed without ever instantiating an FCN, but by instead evaluating the corresponding GP. In this work, we derive an analogous equivalence for multi-layer convolutional neural networks (CNNs) both with and without pooling layers. Surprisingly, in the absence of pooling layers, the corresponding GP is identical for CNNs with and without weight sharing. This means that translation equivariance in SGD-trained finite CNNs has no corresponding property in the Bayesian treatment of the infinite width limit – a qualitative difference between the two regimes that is not present in the FCN case. We confirm experimentally that in some scenarios, while the performance of trained finite CNNs becomes similar to that of the corresponding GP with increasing channel count, with careful tuning SGD-trained CNNs can significantly outperform their corresponding GPs. Finally, we introduce a Monte Carlo method to estimate the GP corresponding to a NN architecture, even in cases where the analytic form has too many terms to be computationally feasible.

1 INTRODUCTION

Neural networks (NNs) demonstrate remarkable performance (He et al., 2016; Oord et al., 2016; Silver et al., 2017; Vaswani et al., 2017), but are still only poorly understood from a theoretical perspective (Goodfellow et al., 2014; Choromanska et al., 2015; Pascanu et al., 2014; Zhang et al., 2017). NN performance is often motivated in terms of model architectures, initializations, and training procedures together specifying biases, constraints, or implicit priors over the class of functions learned by a network. This induced structure in learned functions is believed to be well matched to structure inherent in many practical machine learning tasks, and in many real-world datasets. For instance, properties of NNs which are believed to make them well suited to modeling the world include: hierarchy and compositionality (Lin et al., 2017; Poggio et al., 2017), Markovian dynamics (Tiňo et al., 2004; 2007) , and equivariances in time and space for RNNs (Werbos, 1988) and CNNs (Fukushima & Miyake, 1982; Rumelhart et al., 1985) respectively.

The recent discovery of an equivalence between deep neural networks and GPs (Lee et al., 2018; de G. Matthews et al., 2018) allow us to express an analytic form for the prior over functions encoded by deep NN architectures and initializations. This transforms an implicit prior over functions into an *explicit prior*, which can be analytically interrogated and easily reasoned about. Previous work studying these Neural Network-equivalent Gaussian Processes (NN-GPs) has established the correspondence only for fully connected networks (FCNs). Additionally, previous work has not used analysis of NN-GPs to gain specific insights into the equivalent NNs.

In the present work, we extend the equivalence between NNs and NN-GPs to deep *Convolutional* Neural Networks (CNNs), both with and without pooling. CNNs are a particularly interesting architecture for study, since they are frequently held forth as a success of motivating NN design based on invariances and equivariances of the physical world (Cohen & Welling, 2016) – specifically, designing a NN to respect translation equivariance (Fukushima & Miyake, 1982; Rumelhart et al., 1985). As we will see in this work, absent pooling, this quality can vanish in the Bayesian treatment of the infinite width limit.

The specific novel contributions of the present work are:

1. We show analytically that CNNs with many channels, both with and without average pooling, trained in a fully Bayesian fashion, correspond to an NN-GP (§2, §3). We prove convergence as the number of channels in hidden layers go to infinity uniformly (§A.4.3), strengthening and extending the result of de G. Matthews et al. (2018).
2. We show that in the absence of pooling, the NN-GP for a CNN and a Locally Connected Network (LCN) are identical (§5.1). An LCN has the same local connectivity pattern as a CNN, but without weight sharing or translation equivariance.
3. We experimentally compare trained CNNs and LCNs and find that under certain conditions both often perform similarly to the respective NN-GP (Figures 5, 6). Moreover, both architectures tend to perform better with increased channel count, suggesting that similarly to FCNs (Neyshabur et al., 2015; Novak et al., 2018) CNNs benefit from overparameterization (Figures 1, 5).

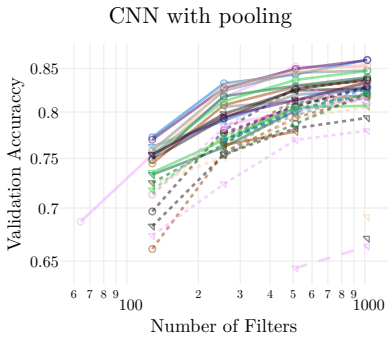

Figure 1: SGD-trained CNNs with pooling generally benefit from overparameterization. Each line corresponds to a particular setting of hyper-parameters, with best learning rate and weight decay selected independently for each number of filters (x-axis). All networks have reached 100% training accuracy on CIFAR10. Best achieved generalization (y-axis) increases as the number of free parameters grows. See §A.6.5 for experimental details.

4. However, we also show that careful tuning of hyper-parameters allows finite CNNs trained with SGD to outperform their corresponding NN-GP by a significant margin. We experimentally disentangle and quantify the contributions stemming from local connectivity, equivariance, and invariance in a convolutional model in one such setting (Table 2), and suggest a potential explanation for the observed difference in performance (§5.3).
5. We introduce a Monte Carlo method to compute NN-GP kernels for situations (such as CNNs with pooling) where evaluating the NN-GP is otherwise computationally infeasible (§4).

1.1 Related work

In early work on neural network priors, Neal (1994) demonstrated that, in a fully connected network with a single hidden-layer, certain natural priors over network *parameters* give rise to a Gaussian process prior over *functions* when the number of hidden units is taken to be infinite. Followup work extended the conditions under which this correspondence applied (Williams, 1997; Le Roux & Bengio, 2007; Hazan & Jaakkola, 2015). An exactly analogous correspondence for infinite width, finite depth *deep* fully connected networks was developed recently (Lee et al., 2018; de G. Matthews et al., 2018).

The line of work examining signal propagation in random deep networks (Poole et al., 2016; Schoenholz et al., 2017; Yang & Schoenholz, 2017; Xiao et al., 2018; Chen et al., 2018) is related to the construction of the GPs we consider. They apply a mean-field approximation in which the pre-activation signal is replaced with a Gaussian, and the derivation of the correlation function with depth gives the kernel function of the corresponding GP. Recently, Xiao et al. (2018) extended this to convolutional architectures without pooling. Xiao et al. (2018) also analyzed properties of the convolutional kernel at large depths to construct a *phase diagram* which will be relevant to NN-GP performance, as discussed in §A.3. Compositional kernels coming from convolutional and fully connected layers also appeared outside of the GP context in (Daniely et al., 2016). In this work, they prove approximation guarantees between a network and its corresponding kernel, and show that empirical kernels will converge as the number of channels increases.

There is a line of work considering stacking of GPs, such as *deep GP*s (Lawrence & Moore, 2007; Damianou & Lawrence, 2013; Kumar et al., 2018). These no longer correspond to GPs, though they can describe a rich class of probabilistic models beyond GPs. Alternatively, *deep kernel learning* (Wilson et al., 2016b;a; Bradshaw et al., 2017) utilizes GPs with base kernels which take in features produced by a deep neural network (often a CNN), and train the resulting model end-to-end. Similarly, van der Wilk et al. (2017) incorporates convolutional structure into GP kernels. Our work differs from all of these in that our GP corresponds exactly to a fully Bayesian CNN in the many channel limit.

Borovykh (2018) also studies CNNs converging to GP behavior. However this work considers the infinite *filter size* limit, and works with CNNs over temporal data. They show that network outputs at different time points are described by a GP after marginalization over an input distribution. Thus they do not address the distribution over network outputs for specific inputs.

In concurrent work Garriga-Alonso et al. (2018) derive an NN-GP kernel equivalent to one of the kernels considered in our work. In addition to explicitly specifying kernels corresponding to pooling and vectorizing, we also compare the NN-GP performance to finite-width SGD-trained CNNs and analyze the differences between the two models.

2 Wide Bayesian CNNs are Gaussian processes

2.1 Preliminaries

Consider a series of L convolutional hidden layers, $l = 0, ..., L-1$. The parameters of the network are the convolutional filters and biases, $\omega^l_{ij,\beta}$ and b^l_i, respectively, with outgoing (incoming) channel index i (j) and filter relative spatial location $\beta = -k, ..., k$.[1] Assume a Gaussian prior on both the filter weights and biases,

$$p\left(\omega^l_{ij,\beta}\right) = \mathcal{N}\left(0, v_\beta \frac{\sigma_w^2}{n^l}\right), \qquad p\left(b^l_i\right) = \mathcal{N}\left(0, \sigma_b^2\right). \tag{1}$$

The weight and bias variance are σ_w^2, σ_b^2, respectively. n^l is the number of channels (filters) in layer l, $2k+1$ is the filter size, and v_β is the fraction of the receptive field variance at location β (with $\sum_\beta v_\beta = 1$). Typically in practice we utilize uniform $v_\beta = 1/(2k+1)$, but nonuniform $v_\beta \neq 1/(2k+1)$ enables kernel properties that are better suited for ultra-deep networks, as in (Xiao et al., 2018).

Let $\mathcal{X}$ denote a set of input images (training set or validation set or both). The network has activations $x^l(x)$ and pre-activations $z^l(x)$ for each input image $x \in \mathcal{X} \subset \mathbb{R}^{n^0 \times d}$, with input channel count n^0, number of pixels d, and where

$$x^l_{i,\alpha}(x) = \begin{cases} x_{i,\alpha} & l = 0 \\ \phi\left(z^{l-1}_{i,\alpha}(x)\right) & l > 0 \end{cases}, \qquad z^l_{i,\alpha}(x) = \sum_{j=1}^{n^l} \sum_{\beta=-k}^{k} \omega^l_{ij,\beta} x^l_{j,\alpha+\beta}(x) + b^l_i. \tag{2}$$

We emphasize the dependence of the quantities on the input x. ϕ is a pointwise nonlinearity . x^l is assumed to be zero padded.

A recurring quantity in this work will be the empirical covariance functions of the activations x^l, defined as

$$\left[K^l\right]_{\alpha,\alpha'}(x, x') \equiv \frac{1}{n^l} \sum_{i=1}^{n^l} x^l_{i,\alpha}(x) x^l_{i,\alpha'}(x'). \tag{3}$$

K is therefore a 4-dimensional random variable indexed by two inputs x, x' and two top-layer spatial locations α, α' (the dependence on layer widths $n^1, \ldots, n^l$ and their weights and biases is implied and by default not stated explicitly). K^0, the empirical covariance of inputs, is deterministic. In obtaining Bayesian predictions from the GP, x and x' will be examples from the training or test set.

[1] We will use Roman letters to index channels and Greek letters for spatial location. We use letters i, j, i', j', etc to denote channel indices, α, α', etc to denote spatial indices and β, β', etc the filter indices. For notational simplicity, we treat the 1D case with spatial dimension d in the text, but the single spatial index can be extended to higher dimensions by replacing with tuples.

2.2 CORRESPONDENCE BETWEEN GAUSSIAN PROCESSES AND PRIORS OVER DEEP CNNS WITH INFINITELY MANY CHANNELS

We next consider the prior over functions computed by a CNN in the limit of infinitely many channels in the hidden (excluding input and output) layers, $n^l \to \infty$ for $1 \leqslant l \leqslant L$, and derive its equivalence to a GP with a compositional kernel. The following section gives a proof which uses the empirical covariance functions K to characterize finite width intermediate layers and relies on explicit marginalization over hidden layers. In Appendix A.4 we give several alternative derivations of the correspondence.

2.2.1 A SINGLE CONVOLUTIONAL LAYER IS A GP CONDITIONED ON THE SECOND MOMENT TENSOR OF THE PREVIOUS LAYER'S ACTIVATIONS

As can be seen in Equation 2, the pre-activations z^l are weighted sums of the Gaussian entries in ω^l and b^l, where the previous layer activations x^{l-1} provide the weighting factors. A weighted sum of Gaussian random variables is itself Gaussian. Specifically,

$$p\left(z^l|x^l\right) = \prod_i p\left(z_i^l|x^l\right) = \prod_i \mathcal{N}\left(z_i^l; 0, \mathcal{A}\left(K^l\right)\right), \tag{4}$$

where the first equality in Equation 4 follows from the independence of the weights and biases for each channel i. The covariance matrix $\mathcal{A}\left(K^l\right)$ for the pre-activations z^l is derived in (Xiao et al., 2018), where $\mathcal{A}$ is an affine transform (a cross-correlation operator followed by a shifting operator) and has functional form:

$$\left[\mathcal{A}\left(K\right)\right]_{\alpha,\alpha'}\left(x,x'\right) \equiv \sum_\beta v_\beta(\sigma_w^2\left[K\right]_{\alpha+\beta,\alpha'+\beta}\left(x,x'\right) + \sigma_b^2). \tag{5}$$

This result is exact due to the choice of a Gaussian prior over weights and biases.

2.2.2 SECOND MOMENT MATRIX BECOMES DETERMINISTIC WITH INCREASING CHANNEL COUNT

The summands in Equation 3 are i.i.d., due to the independence of the weights and biases for each channel i. Subject to weak restrictions on the nonlinearity ϕ, we can apply the law of large number and conclude that,

$$\forall K^{l-1} \quad \lim_{n^l\to\infty} p\left(K^l|K^{l-1}\right) = \delta\left(K^l - \left(\mathcal{C}\circ\mathcal{A}\right)\left(K^{l-1}\right)\right) \quad \text{in probability,} \tag{6}$$

where $\mathcal{C}$ is defined as

$$\left[\mathcal{C}\left(K\right)\right]_{\alpha,\alpha'}\left(x,x'\right) \equiv \mathbb{E}_{u\sim\mathcal{N}(0,K)}\left[\phi\left(u_\alpha(x)\right)\phi\left(u_{\alpha'}(x')\right)\right]. \tag{7}$$

For nonlinearities such as ReLU (Nair & Hinton, 2010) and the error function (erf) $\mathcal{C}$ can be computed in closed form as derived in (Cho & Saul, 2009) and (Williams, 1997) respectively.

2.2.3 BAYESIAN MARGINALIZATION OVER ALL HIDDEN LAYERS

The distribution over the CNN outputs z^L can be evaluated by marginalizing over all intermediate layer covariances in the network (see Figure 2):

$$p\left(z^L|x^0\right) = \int dK^0\cdots dK^L p\left(z^L, K^0\ldots K^L|x^0\right) \tag{8}$$

$$= \int dK^0\cdots dK^L p\left(K^0|x^0\right)\prod_{l=1}^L p\left(K^l|K^{l-1}\right)p\left(z^L|K^L\right). \tag{9}$$

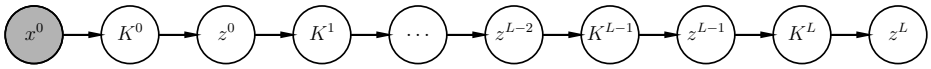

Figure 2: Graphical model for computation performed by neural network (Lee et al., 2018). Per Equation 4 $p\left(z^l|x^l\right)$ only depend on the empirical covariance K^l.

In the limit of infinitely many channels in the hidden layers, $\min\{n^1, \ldots, n^L\} \to \infty$[2] , all the conditional distributions except for $p\left(z^L|K^L\right)$ converge weakly to delta functions and can be integrated out. Then, as elaborated in Appendix A.4 Theorem A.2, Equation 9 reduces to

$$\lim_{\min\{n^1,\ldots,n^L\}\to\infty} p\left(z^L|x^0\right) = \prod_i \mathcal{N}\left(z_i^L; 0, \mathcal{A}\left(K_\infty^L\right)\right), \quad \text{where} \quad K_\infty^L \equiv \left(\mathcal{C}\circ\mathcal{A}\right)^L\left(K^0\right) \tag{10}$$

i.e. $(\mathcal{C}\circ\mathcal{A})$ composed with itself l times and applied to K^0. In other words, K_∞^L is the (deterministic) covariance of the CNN activations in the limit of infinitely-many (hence ∞ subscript) channels in each of the convolutional layers from 0 to $L-1$.

Equation 10 states that the outputs for any set of input examples and pixel indices are jointly Gaussian distributed – i.e. the output of a CNN with infinitely many channels in its L hidden layers is described by a GP with a covariance function $\mathcal{A}\left(K_\infty^L\right)$.

3 Transforming a GP over spatial locations into a GP over classes

In §2.2 we have shown that a deep CNN is a GP indexed by input samples and spatial locations of the top layer. In the infinite channel limit its covariance tensor K_∞^L can be computed in closed form.

Here we show that transformations to obtain class predictions that are common in CNN classifiers can be represented as either vectorization or projection (as long as we treat classification as regression, similarly to (Lee et al., 2018)). Both of these operations preserve the GP equivalence and allow the computation of the covariance tensor $\mathcal{K}$ of the respective GP (now indexed by input samples and target classes) as a simple transformation of K_∞^L.

3.1 Vectorization

One popular way is to vectorize (flatten) the output of the last convolutional layer into a vector $\text{vec}\left(z^L\right) \in \mathbb{R}^{n^{L+1}d}$ and stack a fully connected layer on top:

$$z_i^{L+1} = \sum_{j=1}^{n^{L+1}d} \omega_{ij}^{L+1}\phi\left(\text{vec}\left(z^L\right)\right)_j + b_i^{L+1}, \tag{11}$$

where the weights $\omega^{L+1} \in \mathbb{R}^{c\times n^{L+1}d}$ and biases $b^{L+1} \in \mathbb{R}^c$ are i.i.d. Gaussian, $\omega_{ij}^{L+1} \sim \mathcal{N}(0, \sigma_\omega^2/(n^{L+1}d))$, $b_i^{L+1} \sim \mathcal{N}(0, \sigma_b^2)$. The sample-sample kernel of the output (identical for each class i) of this particular GP, denoted by $\mathcal{GP}^{\text{vec}}$, is

$$\mathcal{K}^{\text{vec}}\big|K^{L+1} = \mathbb{E}\left[{z_i^{L+1}}^T z_i^{L+1}\big|K^{L+1}\right] = \frac{\sigma_\omega^2}{d}\sum_\alpha \left[K^{L+1}\right]_{\alpha,\alpha} + \sigma_b^2, \tag{12}$$

And the limit of infinite width is derived identically to §2.2:

$$\mathcal{K}_\infty^{\text{vec}} \equiv \lim_{\min\{n^1,\ldots,n^L\}\to\infty}\left(\mathcal{K}^{\text{vec}}\big|x^0\right) = \frac{\sigma_\omega^2}{d}\sum_\alpha \left[K_\infty^{L+1}\right]_{\alpha,\alpha} + \sigma_b^2. \tag{13}$$

As observed in Xiao et al. (2018), to compute any diagonal terms of $\left[K_\infty^{L+1}\right](x,x')$, one needs only the corresponding diagonal terms of $\left[K_\infty^l\right](x,x')$. Consequently, we only need to store $\left\{\left[K_\infty^l\right]_{\alpha,\alpha}(x,x') : x,x' \in \mathcal{X}^2, \alpha \in \{1\ldots d\}\right\}_{l=0,\ldots,L}$ and the memory cost is $\mathcal{O}\left(|\mathcal{X}|^2 d\right)$. Note that this approach ignores pixel-pixel covariances and produces a GP corresponding to a locally-connected network (see §5.1).

[2]Unlike de G. Matthews et al. (2018), we do not require each n^l to be strictly increasing.

3.2 PROJECTION

Another approach is to project a multidimensional Gaussian into a 1-dimensional one. Let $v \in \mathbb{R}^d$ be a deterministic vector, $\omega_{ij}^{L+1} \sim \mathcal{N}(0, \sigma_\omega^2/c)$ for $1 \leq j \leq c$, b^{L+1} be the same as above. Define the output to be

$$z_i^{L+1} = \sum_{j=1}^{n^{L+1}} \omega_{ij}^{L+1} \left(\phi\left(z^L\right) v\right)_j + b_i^{L+1}. \tag{14}$$

Any choice of v induces a GP whose sample-sample kernel is

$$\mathcal{K}_\infty^v \equiv \sigma_\omega^2 \sum_{\alpha,\alpha'} v_\alpha v_\beta \left[K_\infty^{L+1}\right]_{\alpha,\beta} + \sigma_b^2, \tag{15}$$

where the limiting behavior is derived identically to Equations 12 and 13. Examples of this approach include

1. **Average pooling:** take $v = \frac{1}{d}\mathbf{1_d}$ and denote this particular GP as $\mathcal{GP}^{\text{pool}}$. Then

$$\mathcal{K}_\infty^{\text{pool}} \equiv \frac{\sigma_\omega^2}{d^2} \sum_{\alpha,\beta} \left[K_\infty^{L+1}\right]_{\alpha,\beta} + \sigma_b^2. \tag{16}$$

   This approach corresponds to using a global average pooling right after the last convolution layer. Spatially local average pooling layers can be constructed in a similar fashion. This approach takes all pixel-pixel covariance into consideration and makes the kernel translation invariant. However, it requires $\mathcal{O}\left(|\mathcal{X}|^2 d^2\right)$ memory to compute the sample-sample covariance of the GP. It is impractical to use this method to evaluate the GP, and we propose to use a Monte Carlo approach (see §4).

2. **Subsampling one particular pixel:** take $v = e_\alpha$,

$$\mathcal{K}_\infty^{e_\alpha} \equiv \sigma_\omega^2 \left[K_\infty^{L+1}\right]_{\alpha,\alpha} + \sigma_b^2. \tag{17}$$

   This approach (denoted $\mathcal{GP}^{e_\alpha}$) makes use of only one pixel-pixel covariance, and requires the same amount of memory as $\mathcal{GP}^{\text{vec}}$ to compute.

We compare the performance of presented strategies in Figure 3. Note that all described strategies admit stacking additional FC layers on top while retaining the GP equivalence, using the recursion relationship in (Lee et al., 2018; de G. Matthews et al., 2018).

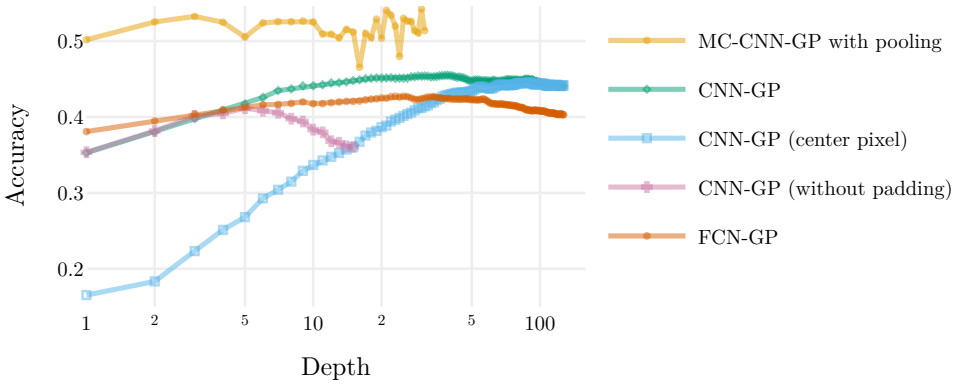

Figure 3: Different dimensionality collapsing strategies described in §3. Validation accuracy of an **MC-CNN-GP with pooling** (item 3.2.1) is consistently better than other models due to translation invariance of the kernel. **CNN-GP with zero padding** (§3.1) outperforms an analogous **CNN-GP without padding** as depth increases. At depth of 15 the spatial dimension of the output without padding is reduced 1×1, making the **CNN-GP without padding** equivalent to the **center pixel selection strategy** (item 3.2.2) – which performs worse than **CNN-GP** (we conjecture, due to overfitting to centrally-located features) but approaches the latter (right) in the limit of large depth, as information becomes more uniformly spatially distributed (Xiao et al., 2018). CNN-GPs generally outperform **FCN-GP**, presumably due to the local connectivity prior, but can fail to capture nonlinear interactions between spatially-distant pixels at shallow depths (left). Values are reported on a 2K/4K train/validation subsets of CIFAR10. See §A.6.2 for experimental details.

4 MONTE CARLO EVALUATION OF INTRACTABLE GP KERNELS

We introduce a Monte Carlo estimation method for NN-GP kernels which are computationally impractical to compute analytically, or for which we do not know the analytic form. Similar in spirit to traditional random feature methods (Rahimi & Recht, 2007), the core idea is to instantiate many random *finite* width networks and use the empirical second moments of activations to estimate the Monte Carlo-GP (MC-GP) kernel,

$$\left[K^l_{n,M}\right]_{\alpha,\alpha'}(x,x') \equiv \frac{1}{Mn}\sum_{m=1}^{M}\sum_{c=1}^{n} x^l_{c\alpha}\left(x;\theta_m\right) x^l_{c\alpha'}\left(x';\theta_m\right) \tag{18}$$

where θ consists of M draws of the weights and biases from their prior distribution, $\theta_m \sim p\left(\theta\right)$, and n is the width or number of channels in hidden layers. The MC-GP kernel converges to the analytic kernel with increasing width, $\lim_{n\to\infty} K^l_{n,M} = K^l_\infty$ in probability.

For finite width networks, the uncertainty in $K^l_{n,M}$ is $\mathrm{Var}[K^l_{n,M}] = \mathrm{Var}_\theta\left[K^l_n\left(\theta\right)\right]/M$. From Daniely et al. (2016), we know that $\mathrm{Var}_\theta\left[K^l_n\left(\theta\right)\right] \propto \frac{1}{n}$, which leads to $\mathrm{Var}_\theta[K^l_{n,M}] \propto \frac{1}{Mn}$. For finite n, $K^l_{n,M}$ is also a biased estimate of K^l_∞, where the bias depends solely on network width. We do not currently have an analytic form for this bias, but we can see in Figures 4 and 9 that for the hyperparameters we probe it is small relative to the variance. In particular, $\left\|K^l_{n,M}\left(\theta\right) - K^L_\infty\right\|^2_{\mathrm{F}}$ is nearly constant for constant Mn. We thus treat Mn as the effective sample size for the Monte Carlo kernel estimate. Increasing M and reducing n can reduce memory cost, though potentially at the expense of increased compute time and bias.

The MC-GP reduces the memory requirements to compute $\mathcal{GP}^{\text{pool}}$ from $\mathcal{O}\left(\left|\mathcal{X}\right|^2 d^2\right)$ to $\mathcal{O}\left(\left|\mathcal{X}\right|^2 + n^2 + nd\right)$, making the evaluation of CNN-GPs with pooling practical.

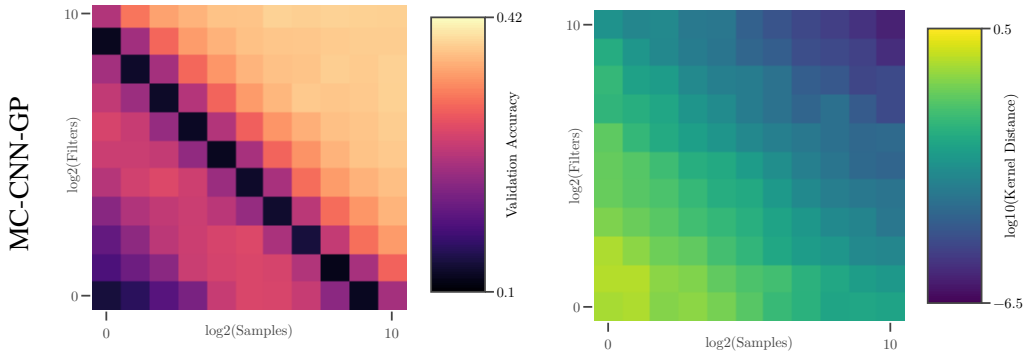

Figure 4: **Validation accuracy** (left) of an MC-CNN-GP increases with $n \times M$ (i.e. filter count times number of samples) and approaches that of the exact CNN-GP (not shown), while the **distance** (right) to the exact kernel decreases. The dark band corresponds to ill-conditioning of $K^L_{n,M}$ when the number of outer products contributing to $K^L_{n,M}$ approximately equals its rank. Values reported for a 3-layer model on a 2K/4K train/validation subset of CIFAR10 downsampled to 8×8. See Figure 9 for similar results with other architectures and §A.6.3 for experimental details.

5 DISCUSSION

5.1 BAYESIAN CNNS WITH MANY CHANNELS ARE IDENTICAL TO LOCALLY CONNECTED NETWORKS, IN THE ABSENCE OF POOLING

Locally Connected Networks (LCNs) (Fukushima, 1975; Lecun, 1989) are CNNs without weight sharing between spatial locations. LCNs preserve the connectivity pattern, and thus topology, of a CNN. However, they do not possess the equivariance property of a CNN – if an input is translated, the latent representation in an LCN will be completely different, rather than also being translated.

The CNN-GP predictions without spatial pooling in §3.1 and §3.2 depend only on sample-sample covariances, and do not depend on pixel-pixel covariances. LCNs destroy pixel-pixel covariances: $\left[K_\infty^L\right]_{\alpha,\alpha'}^{\text{LCN}}(x,x') = 0$, for $\alpha \neq \alpha'$ and all $(x,x') \in \mathcal{X}^2$ and $L > 0$. However, LCNs preserve the correlations between input examples at every pixel: $\left[K_\infty^L\right]_{\alpha,\alpha}^{\text{LCN}}(x,x') = \left[K_\infty^L\right]_{\alpha,\alpha}^{\text{CNN}}(x,x')$. As a result, in the absence of pooling, LCN-GPs and CNN-GPs are identical. Moreover, LCN-GPs with pooling are identical to CNN-GPs with vectorization of the top layer (under suitable scaling of x^{L+1}). We confirm these findings experimentally in trained networks in the limit of large width in Figure 5, as well as by demonstrating convergence of MC-GPs of the respective architectures to the same CNN-GP (modulo scaling of x^{L+1}) in Figures 4 and 9.

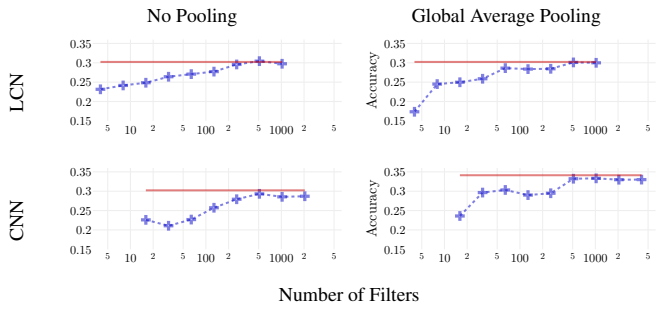

Figure 5: Best validation accuracy of **trained neural networks** fitting the whole training set approaches that of a respective **CNN-GP** as the number of channels increases. Notice that unless a network has *both* weight sharing and pooling (bottom-right), performance of all models converges to the same CNN-GP kernel (modulo a scaling term on the final layer activations x^{L+1}) performance. Translation invariance is only enabled with both weight sharing and pooling (bottom-right), allowing for an improved performance of both CNNs and the respective CNN-GP. Values are reported for 3-layer models on a 0.5K/4K train/validation subsets of CIFAR10 downsampled to 8×8. All networks have a 100% training accuracy. See Figures 8 and 7 for training and validation loss, and §A.6.1 for experimental details.

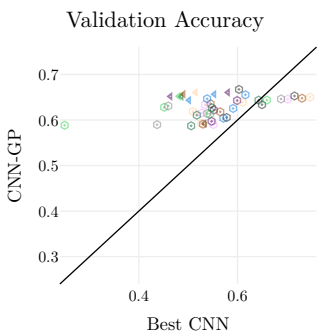

Figure 6: CNN-GP can often outperform the best trained CNN of the same architecture that fit the training set. Each point corresponds to the validation accuracy of: (y-axis) a specific CNN-GP; (x-axis) the best CNN with the same parameters selected among the 100%-accurate models on the full training CIFAR10 dataset with different learning rates, weight decay and number of filters. See §A.6.5 for experimental details.

5.2 Pooling leverages equivariance to provide invariance

The only kernel leveraging pixel-pixel covariances is that of the CNN-GP with pooling. This enables the predictions of this GP and the corresponding CNN to be invariant to translations (modulo edge effects) – a clearly beneficial quality for an image classifier. We observe strong experimental evidence supporting the benefits of invariance throughout this work (Figures 3, 4, 5; Tables 1 and 2), in both CNNs and CNN-GPs.

5.3 Finite-channel SGD-trained CNNs can outperform infinite-channel Bayesian CNNs, in the absence of pooling

Absent pooling benefits of equivariance and weight sharing are more challenging to explain in terms of priors on predictions made with image classifiers (since it is not a property of outputs, but of the intermediary representations). Indeed, in this work we find that CNN-GPs can often perform competitively to their finite-width SGD-trained counterparts (Figures 5, 6)[3].

However, as can be seen in Table 1 and Figure 6, the best CNN *overall* outperforms the best CNN-GP by a significant margin - an observation specific to CNNs and not FCNs or LCNs[4]. In Table 2 we

[3]This observation is conditioned on the respective NN fitting the training set to 100%. Underfitting breaks the correspondance to an NN-GP, since train set predictions of such a network no longer correspond to the true training labels.

[4]While we did not perform as extensive an evaluation of these models as of CNNs in this work, we have not observed these architectures to outperform their Bayesian infinite width limits by a substantial margin.

demonstrate this large gap in performance by evaluating different models with equivalent settings that are chosen for good SGD-trained CNN performance.

We conjecture that equivariance, a property lacking in LCNs and the Bayesian treatment of the infinite width CNN limit, contributes to the performance of SGD-trained finite-width CNNs with the correct settings of hyper-parameters. Indeed, notice that in the context of gradient optimization, weight sharing is easier to relate to the outputs of the classifier. It is straightforward to demonstrate that through equivariance of internal representations, weight sharing enforces *equivariance of gradients*[5] (modulo edge effects). There is no corresponding qualitative gradient property of an FCN or an LCN (with or without pooling), which is consistent with best SGD-trained networks of these architectures performing similarly to NN-GPs.

Nonetheless, more work is needed to disentangle and quantify the separate contributions of stochastic optimization and infinite width limit to provide a comprehensive explanation.

Model	CIFAR10	MNIST	Fashion-MNIST
CNN with pooling	14.85 (15.65)	–	–
CNN with ReLU and large learning rate	24.76 (17.64)	–	–
CNN-GP	32.86	0.88	7.40
CNN with small learning rate	33.31(22.89)	–	–
CNN with erf (any learning rate)	33.31(22.17)	–	–
Convolutional GP (van der Wilk et al., 2017)	35.40	1.17	–
Deep Convolutional GP (Kumar et al., 2018)	45.00	1.34	–
ResNet GP (Garriga-Alonso et al., 2018)	–	0.84	–
Residual CNN-GP (Garriga-Alonso et al., 2018)	–	0.96	–
CNN-GP (Garriga-Alonso et al., 2018)	–	1.03	–
FCN-GP	41.06	1.22	8.22
FCN-GP (Lee et al., 2018)	44.34	1.21	–
FCN	45.52 (44.73)	–	–

Table 1: **Test error (%)** of the different models. All networks have reached 100% training accuracy. Note that without such condition, higher test performance is achieved (values reported in the parentheses), though this breaks the correspondence to a NN-GP, since train set predictions of an underfit CNN no longer correspond to the true training labels. See §A.6.5 for experimental details.

Quality:	Compositionality		Local connectivity		Equivariance	Invariance
Model:	FCN	FCN-GP	LCN (w/ pooling)	CNN-GP	CNN	CNN w/ pooling
Error:	46.26	41.45	36.52 (36.23)	36.71	19.93	16.54

Table 2: **Test error (%)** of different models of the same architecture and a conjectured explanation for the observed difference. Values reported for 8-layer ReLU models on CIFAR10. See §A.6.6 for experimental details.

6 Conclusion

In this work we have derived a Gaussian process that corresponds to a fully Bayesian, infinitely wide CNN. The covariance of this GP can be efficiently computed either in closed form or by using Monte Carlo sampling, depending on the architecture.

CNN-GP can often perform competitively to the best CNNs (that fit the training set) of equivalent architecture and weight priors, which makes it an appealing choice for small datasets, as it eliminates all training-related hyper-parameters. However, we found that the best *overall* performance is achieved by finite SGD-trained CNNs and not by their infinitely wide Bayesian counterparts. We hope our work stimulates future research into disentangling the contributions of the two qualities (Bayesian treatment and infinite width) to the performance gap observed.

[5] An addition of global average pooling guarantees gradient invariance.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

Anastasia Borovykh. A gaussian process perspective on convolutional neural networks. 05 2018.

John Bradshaw, Alexander G de G Matthews, and Zoubin Ghahramani. Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks. *arXiv preprint arXiv:1707.02476*, 2017.

Minmin Chen, Jeffrey Pennington, and Samuel Schoenholz. Dynamical isometry and a mean field theory of RNNs: Gating enables signal propagation in recurrent neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 873–882, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/chen18i.html.

Youngmin Cho and Lawrence K Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pp. 342–350, 2009.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pp. 192–204, 2015.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pp. 2990–2999, 2016.

Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pp. 207–215, 2013.

Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pp. 2253–2261, 2016.

Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1-nGgWC-.

Kunihiko Fukushima. Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3-4):121–136, 1975.

Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pp. 267–285. Springer, 1982.

Adrià Garriga-Alonso, Laurence Aitchison, and Carl Edward Rasmussen. Deep convolutional networks as shallow Gaussian processes. *arXiv preprint arXiv:1808.05587*, aug 2018. URL https://arxiv.org/abs/1808.05587.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1487–1495. ACM, 2017.

Ian J Goodfellow, Oriol Vinyals, and Andrew M Saxe. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

Vinayak Kumar, Vaibhav Singh, PK Srijith, and Andreas Damianou. Deep gaussian processes with convolutional kernels. *arXiv preprint arXiv:1806.01655*, 2018.

Neil D Lawrence and Andrew J Moore. Hierarchical gaussian process latent variable models. In *Proceedings of the 24th international conference on Machine learning*, pp. 481–488. ACM, 2007.

Nicolas Le Roux and Yoshua Bengio. Continuous neural networks. In *Artificial Intelligence and Statistics*, pp. 404–411, 2007.

Yann Lecun. Generalization and network design strategies. In *Connectionism in perspective*. Elsevier, 1989.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Sam Schoenholz, Jeffrey Pennington, and Jascha Sohl-dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1EA-M-0Z.

Henry W Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Radford M. Neal. Priors for infinite networks (tech. rep. no. crg-tr-94-1). *University of Toronto*, 1994.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *Proceeding of the international Conference on Learning Representations workshop track*, abs/1412.6614, 2015.

Roman Novak, Yasaman Bahri, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJC2SzZCW.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Razvan Pascanu, Yann N Dauphin, Surya Ganguli, and Yoshua Bengio. On the saddle point problem for non-convex optimization. *arXiv preprint arXiv:1405.4604*, 2014.

Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, Oct 2017. ISSN 1751-8520. doi: 10.1007/s11633-017-1054-2. URL https://doi.org/10.1007/s11633-017-1054-2.

Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances In Neural Information Processing Systems*, pp. 3360–3368, 2016.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

Ali Rahimi and Ben Recht. Random features for large-scale kernel machines. In *In Neural Information Processing Systems*, 2007.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

Samuel S Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. *ICLR*, 2017.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Peter Tiňo, Michal Cernansky, and Lubica Benuskova. Markovian architectural bias of recurrent neural networks. *IEEE Transactions on Neural Networks*, 15(1):6–15, 2004.

Peter Tiňo, Barbara Hammer, and Mikael Bodén. Markovian bias of neural-based architectures with feedback connections. In *Perspectives of neural-symbolic integration*, pp. 95–133. Springer, 2007.

Mark van der Wilk, Carl Edward Rasmussen, and James Hensman. Convolutional gaussian processes. In *Advances in Neural Information Processing Systems 30*, pp. 2849–2858, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Paul J Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

Christopher KI Williams. Computing with infinite networks. In *Advances in neural information processing systems*, pp. 295–301, 1997.

Andrew G Wilson, Zhiting Hu, Ruslan R Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pp. 2586–2594, 2016a.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pp. 370–378, 2016b.

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington. Dynamical isometry and a mean field theory of CNNs: How to train 10,000-layer vanilla convolutional neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5393–5402, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/xiao18a.html.

Ge Yang and Samuel Schoenholz. Mean field residual networks: On the edge of chaos. In *Advances in neural information processing systems*, pp. 7103–7114, 2017.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

A APPENDIX

A.1 ADDITIONAL FIGURES

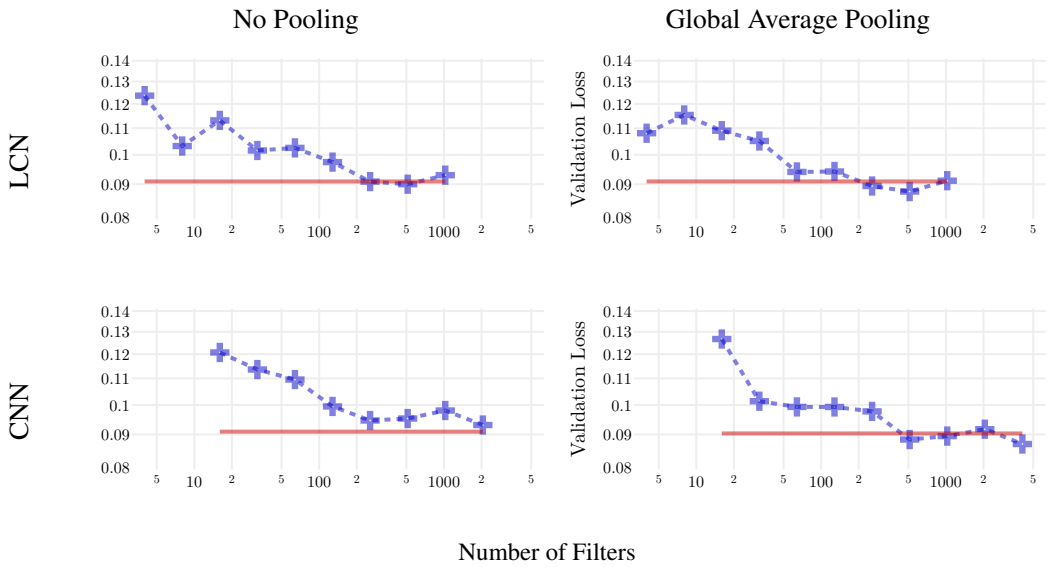

Figure 7: Best validation loss (vertical axis) of **trained neural networks** (dashed line) as the number of channels increases (horizontal axis) approaches that of a respective **(MC-)CNN-GP** (solid horizontal line). See Figure 5 for validation accuracy, Figure 8 for training loss and §A.6.1 for experimental details.

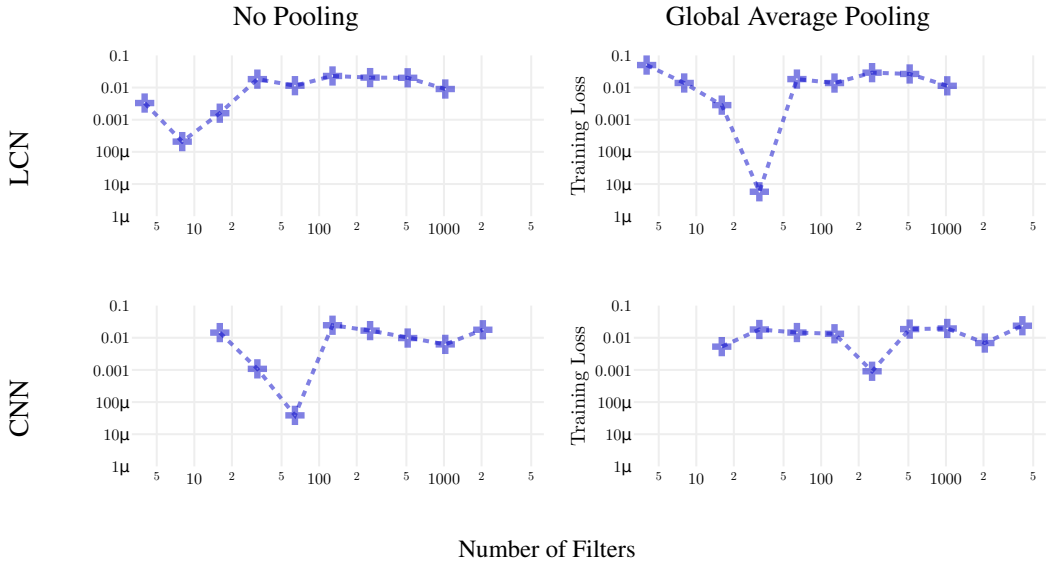

Figure 8: **Training loss** (vertical axis) of best (in terms of validation loss) neural networks as the number of channels increases (horizontal axis). While perfect 0 loss is not achieved (but 100% accuracy is), we observe no consistent improvement when increasing the capacity of the network (left to right). This eliminates underfitting as a possible explanation for why small models perform worse in Figure 5. See Figure 7 for validation loss and §A.6.1 for experimental details.

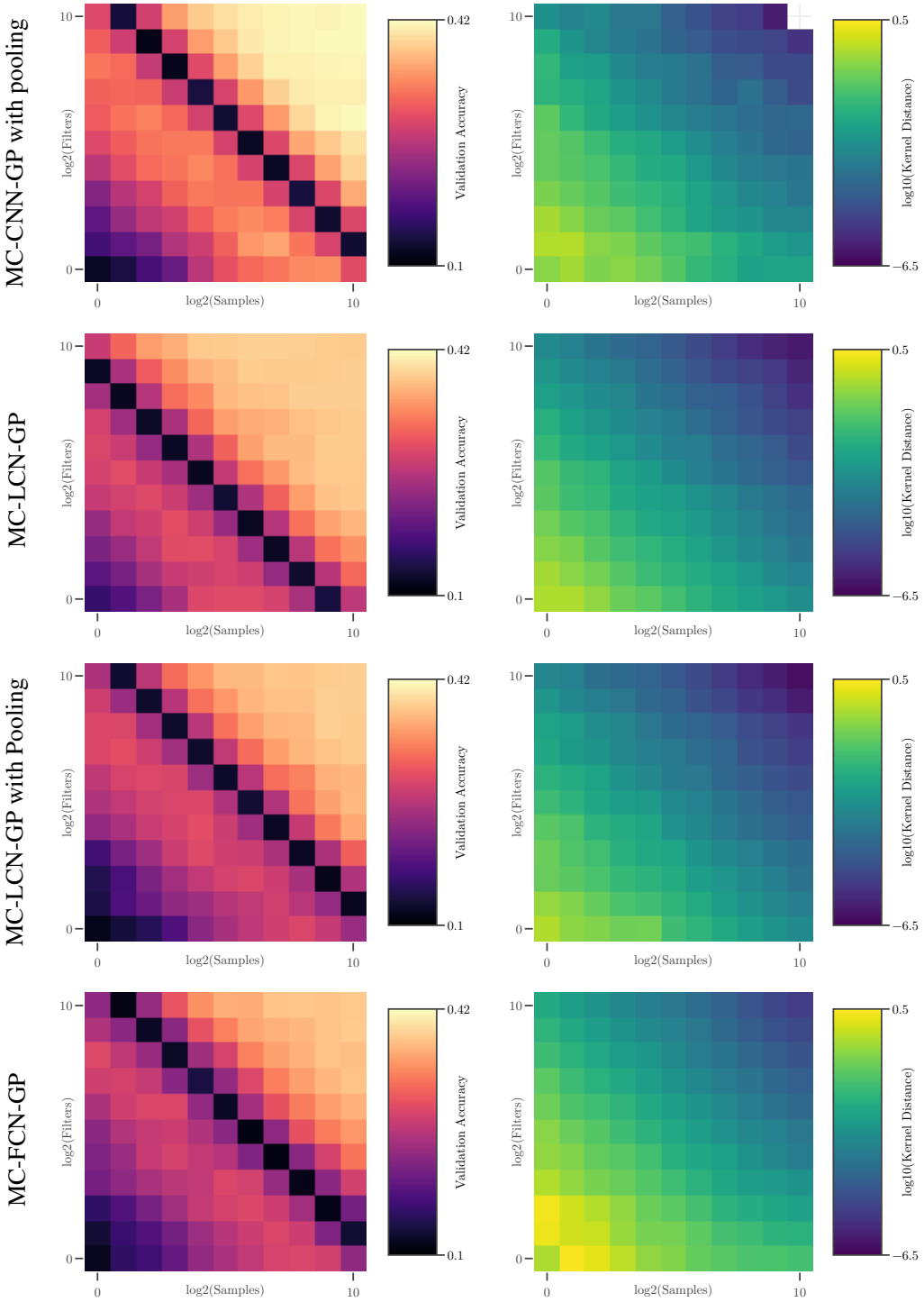

Figure 9: As in Figure 4, **validation accuracy** (left) of MC-GPs increases with $n \times M$ (i.e. width times number of samples), while the **distance** (right) to the the respective exact GP (or the best available estimate in the case of CNN-GP with pooling, top row) decreases. We remark that when using shared weights, convergence is slower as smaller number of independent random parameters are being used. For example a single-layer MC-LCN-GP kernel is expected to converge approximately $\mathrm{Var}[K^l_{\text{CNN}}]/\mathrm{Var}[K^l_{\text{LCN}}] \sim \sqrt{\#\text{ LCN params}/\#\text{ CNN params}} = \sqrt{\text{spatial size of the output layer}}$ times faster than MC-CNN-GP, which is in agreement with our results obtained in the two top rows. I.e. the geometric mean of the ratios of the kernel distance from (3-layer) MC-CNN-GP and MC-LCN-GP to the respective CNN-GP is $\approx 2.2 \gtrsim \sqrt{\text{spatial size of the output layer}} = \sqrt{2 \times 2} = 2$). See §A.6.3 for experimental details.

A.2 REVIEW OF EXACT BAYESIAN REGRESSION WITH GPS

Our discussion in the paper has focused on model *priors*. A crucial benefit we derive by mapping to a GP is that Bayesian inference is straightforward to implement and can be done *exactly* for regression (Rasmussen & Williams, 2006, chapter 2), requiring only simple linear algebra. Let $\mathcal{X}$ denote training inputs $x_1, ..., x_{|\mathcal{X}|}$, $\mathbf{t}^T = (t_1, ..., t_{|\mathcal{X}|})$ the training targets, and collectively $\mathcal{D}$ for the training set. The integral over the posterior can be evaluated analytically to give a posterior predictive distribution on a test point x_* which is Normal, $z^*|\mathcal{D}, x^* \sim \mathcal{N}(\mu_*, \sigma_*^2)$, with

$$\mu_* = \mathcal{K}(x^*, \mathcal{X})(\mathcal{K}(\mathcal{X}, \mathcal{X}) + \sigma_\epsilon^2 \mathbb{I}_{|\mathcal{X}|})^{-1}\mathbf{t}, \tag{19}$$

$$\sigma_*^2 = \mathcal{K}(x^*, x^*) - \mathcal{K}(x^*, \mathcal{X})(\mathcal{K}(\mathcal{X}, \mathcal{X}) + \sigma_\epsilon^2 \mathbb{I}_{|\mathcal{X}|})^{-1}\mathcal{K}(\mathcal{X}, x^*). \tag{20}$$

We use the shorthand $\mathcal{K}(\mathcal{X}, \mathcal{X})$ to denote the $|\mathcal{X}| \times |\mathcal{X}|$ matrix formed by evaluating the GP covariance on the training inputs, and likewise $\mathcal{K}(x^*, \mathcal{X})$ is a $|\mathcal{X}|$-length vector formed from the covariance between the test input and training inputs. Computationally, the costly step in GP posterior predictions comes from the matrix inversion, which in all experiments were carried out exactly, and typically scales as $\mathcal{O}(|\mathcal{X}|^3)$ (though algorithms scaling as $\mathcal{O}(|\mathcal{X}|^{2.4})$ exist for sufficiently large matrices). Nonetheless, there is a broad literature on approximate Bayesian inference with GPs which can be utilized for efficient implementation (Rasmussen & Williams, 2006, chapter 8); (Quiñonero-Candela & Rasmussen, 2005).

A.3 RELATIONSHIP TO DEEP SIGNAL PROPAGATION

The recurrence relation linking the GP kernel at layer $l+1$ to that of layer l following from Equation 10 (i.e. $K_\infty^{l+1} = (\mathcal{C} \circ \mathcal{A})\left(K_\infty^l\right)$) is precisely the *covariance map* examined in a series of related papers on signal propagation (Xiao et al., 2018; Poole et al., 2016; Schoenholz et al., 2017; Lee et al., 2018) (modulo notational differences; denoted as $F, \mathcal{C}$ or $\mathcal{A}\star\mathcal{C}$ in (Xiao et al., 2018)). In those works, the action of this map on hidden-state covariance matrices was interpreted as defining a dynamical system whose large-depth behavior informs aspects of trainability. In particular, as $l \to \infty$, $K_\infty^{l+1} = (\mathcal{C} \circ \mathcal{A})\left(K_\infty^l\right) \approx K_\infty^l \equiv K_\infty^*$, i.e. the covariance approaches a fixed point K_∞^*. The convergence to a fixed point is problematic for learning because the hidden states no longer contain information that can distinguish different pairs of inputs. It is similarly problematic for GPs, as the kernel becomes pathological as it approaches a fixed point. Precisely, in the chaotic regime outputs of the GP become asymptotically decorrelated and therefore independent, while in the ordered regime they approach perfect correlation of 1. Either of these scenarios captures no information about the training data in the kernel and makes learning infeasible. This problem can be ameliorated by judicious hyperparameter selection, which can reduce the rate of exponential convergence to the fixed point. For hyperpameters chosen on a critical line separating two untrainable phases, the convergence rates slow to polynomial, and very deep networks can be trained, and inference with deep NN-GP kernels can be performed – see Table 3.

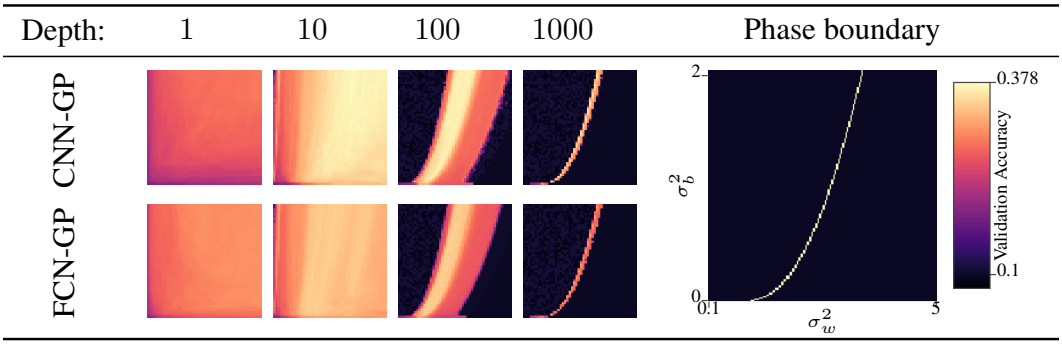

Table 3: **Validation accuracy** of CNN- and FCN- GPs as a function of weight (σ_w^2, horizontal axis) and bias (σ_b^2, vertical axis). As expected, the regions of good performance concentrate around the critical line (phase boundary, right) as the depth increases (left to right). All plots share common axes ranges. See §A.6.3 for experimental details.

A.4 KERNEL CONVERGENCE PROOF

In this section, we present three different approaches to illustrate the weak convergence of neural networks to Gaussian process as the number of channels goes to infinity. Although the first A.4.1 and second approaches A.4.2 (taking iterated limits) are less formal, they provide some intuitions to the convergence of neural network to Gaussian process. The approach in Section A.4.3 is more standard and the proof is more involved. We only provide the arguments for convolution neural networks. It is straightforward to extend them to locally- or fully connected networks.

We will use the following well-known theorem.

Theorem A.1 (Portmanteau Theorem)**.** *Let X_n be a sequence of real-valued random variables. The following are equivalent:*

1. *$X_n \to X$ in distribution,*
2. *for all bounded continuous function f,*

$$\lim_{n\to\infty} \mathbb{E}[f(X_n)] = \mathbb{E}[f(X)], \tag{21}$$

3. *The characteristic functions of X_n, i.e. $\mathbb{E}e^{itX_n}$ converges to that of X point-wisely, i.e., for all t,*

$$\lim_{n\to\infty} \mathbb{E}\left[e^{itX_n}\right] = \mathbb{E}\left[e^{itX}\right]. \tag{22}$$

A.4.1 FORWARD MODE

We show that when taking $n^1 \to \infty, \ldots, n^L \to \infty$ sequentially, a CNN converges to a Gaussian Process in the following sense: preactivations of each layers ($l \geq 1$) converge to a Gaussian in distribution. We will proceed by induction. Let $n^1 \to \infty$. It is not difficult to see that $\left\{z_j^0\right\}$ are pair-wisely independent Gaussian with identical distribution and thus iid Gaussian. Assume $\left\{z_j^l\right\}$ are i.i.d. Gaussian (unconditionally). We claim that so are $\left\{z_j^{l+1}\right\}_{1\leq j\leq\infty}$. $\left\{z_j^{l+1}\right\}$. Indeed, since both the connection weights from layer l to layer $l+1$ and the biases from different channels are independent, $\left\{z_j^{l+1}\right\}$ are pair-wisely independent and have the same distribution. To prove that they are mutually independent, we only need to show that for each j, z_j^{l+1} converges to a Gaussian in distribution as $n^l \to \infty$. Since $\left\{x_j^{l+1}\right\} = \left\{\phi\left(z_j^l\right)\right\}$ are i.i.d., thus the outcomes of the inner sum of Equation 2 are i.i.d. We can then apply a multivariate central limit theorem[6] to conclude that z_j^{l+1} converges to a Gaussian in distribution (note that we have applied the fact that b_j^{l+1} is a Gaussian).

A.4.2 REVERSE MODE

Conditioning on K^{l-1}, K^l is a random variable that converges to $(\mathcal{C}\circ\mathcal{A})\left(K^{l-1}\right)$ in probability as the number of channels $n^l \to \infty$ (the law of large number, see Equation 6).

It is clear that different channels of z^L are uncorrelated and have the same distribution. We will show that for any channel index j, the random variable z_j^L "converges" to the Gaussian

$$\mathcal{N}\left(0, \mathcal{A}\circ(\mathcal{C}\circ\mathcal{A})^L\left(K^0\right)\right) \tag{23}$$

in the sense that its characteristic function converges point-wisely to that of $\mathcal{N}\left(0, \mathcal{A}\circ(\mathcal{C}\circ\mathcal{A})^L\left(K^0\right)\right)$, i.e. for each j and for all vectors t

$$\lim_{n^1\to\infty}\cdots\lim_{n^L\to\infty}\left(G_{n^1,\ldots,n^L}(t) \equiv \mathbb{E}\left[e^{iz_0^L\cdot t}\right]\right) = e^{-\frac{1}{2}t^T\mathcal{A}\circ(\mathcal{C}\circ\mathcal{A})^L\left(K^0\right)t}. \tag{24}$$

[6] Assuming the covariance of $\phi\left(z_j^l\right)$ is finite.

Proof. Applying Fubini's Theorem and the formula of the characteristic function of multivariate Gaussian

$$G_{n^1,\ldots,n^L}(t) = \int e^{iz_0^L \cdot t} p\left(z_0^L|K^L\right) p\left(K^L|K^{L-1}\right) \cdots p\left(K^1|K^0\right) dK^L \cdots dK^0 dz_0^L \tag{25}$$

$$= \int e^{iz_0^L \cdot t} p\left(z_0^L|K^L\right) dz_0^L p\left(K^L|K^{L-1}\right) \cdots p\left(K^1|K^0\right) dK^L \cdots dK^0 \tag{26}$$

$$= \int e^{-\frac{1}{2}t^T K^L t} p\left(K^L|K^{L-1}\right) \cdots p\left(K^1|K^0\right) dK^L \cdots dK^0 \tag{27}$$

$$= \int \left(\int e^{-\frac{1}{2}t^T K^L t} p\left(K^L|K^{L-1}\right) dK^L \right) \cdot \tag{28}$$

$$p\left(K^{L-1}|K^{L-2}\right) \cdots p\left(K^1|K^0\right) dK^{L-1} \cdots dK^0 \tag{29}$$

We now apply $\lim_{n^L \to \infty}$ and switch the order of it with the outer integral. The Lebesgue dominant theorem allows us to do so because the inner integral is bounded above by the constant function $g = 1$ which is absolutely integrable w.r.t. the outer integral. We then apply Theorem A.1, since $e^{-\frac{1}{2}t^T K^L t}$ is bounded and continuous in K^L and $K^L \to \mathcal{A} \circ (\mathcal{C} \circ \mathcal{A})\left(K^{L-1}\right)$.

$$\lim_{n^L \to \infty} G_{n^1,\ldots,n^L}(t) = \int e^{-\frac{1}{2}t^T \mathcal{A} \circ (\mathcal{C} \circ \mathcal{A})(K^{L-1})t} p\left(K^{L-1}|K^{L-2}\right) \cdots p\left(K^1|K^0\right) dK^{L-1} \cdots dK^0. \tag{30}$$

Repeatedly applying the same argument[7] gives

$$\lim_{n^1 \to \infty} \cdots \lim_{n^L \to \infty} G_{n^1,\ldots,n^L}(t) = e^{-\frac{1}{2}t^T \mathcal{A} \circ (\mathcal{C} \circ \mathcal{A})^L \left(K^0\right) t}. \tag{31}$$

Note that the addition of various layers on top (as discussed in §3) does not change the proof in a qualitative way.

□

A.4.3 More general setting

In this section, we present a sufficient condition on the activation function ϕ so that the neural networks will converge to a Gaussian process as all the widths approach to infinity simultaneously. More precisely, we are interested in the case $n^l = n^l(t) \to \infty$ as $t \to \infty$, i.e.,

$$\lim_{t \to \infty} \min\left\{n^1(t), \ldots, n^L(t)\right\} \to \infty. \tag{32}$$

Using Theorem A.1 and the arguments in the above section, it is not difficult to see that a sufficient condition is that the empirical covariance converges in probability to the analytic covariance.

Corollary A.1.1. *If K^L converges to K^L_∞ in probability as $t \to \infty$, then*

$$z_i^L \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, K^L_\infty\right) = \mathcal{N}\left(0, \mathcal{A} \circ (\mathcal{C} \circ \mathcal{A})^L \left(K^0\right)\right). \tag{33}$$

In the remaining section, we provide a sufficient condition for Corollary A.1.1, borrowing some ideas from Daniely et al. (2016). Let PSD_m denote the set of $m \times m$ positive semi-definite matrices and for $R \geq 1$, let

$$\text{PSD}_m(R) = \left\{\Sigma \in \text{PSD}_m : 1/R \leq \Sigma_{\alpha,\alpha} \leq R \quad \text{for} \quad 1 \leq \alpha \leq m\right\}. \tag{34}$$

Let $\mathcal{T}_\infty$ and $\mathcal{T}_n : \text{PSD}_2 \to \mathbb{R}$ be a function and a random variable (induced by the activation ϕ) given by

$$\mathcal{T}_\infty(\Sigma) = \mathbb{E}_{(x,y)\sim\mathcal{N}(0,\Sigma)}[\phi(x)\phi(y) \tag{35}$$

and

$$\mathcal{T}_n(\Sigma) = \frac{1}{n}\sum_{i=1}^n \phi(x_i)\phi(y_i), \tag{36}$$

[7] Here we need $\mathcal{C}$ to be continuous.

where $\{(x_i, y_i)\}_{i=1}^n$ are drawn independently from $\mathcal{N}(0, \Sigma)$. We assume ϕ is uniformly squared integrable with respect to the Gaussian measure $\mathcal{N}(0, r)$ for $1/R \leq r \leq R$, i.e.,

$$\sup_{r \in [1/R, R]} \|\phi\|_{L^2(\mathcal{N}(0,r))} < \infty. \tag{37}$$

By the law of large numbers, $\mathcal{T}_n(\Sigma) \to \mathcal{T}_\infty(\Sigma)$ in probability for all $\Sigma \in \mathrm{PSD}_2(R)$, but this is not enough for our purpose and we need a stronger type of convergence: uniform convergence in probability. Note that the function $\mathcal{T}_\infty$ is closely related to the mapping $\mathcal{C}$: if Σ is the 2×2 symmetric matrix with $\Sigma_{\alpha,\alpha'} = [K]_{\alpha,\alpha'}(x, x')$, $\Sigma_{\alpha,\alpha} = [K^l]_{\alpha,\alpha}(x, x)$ and $\Sigma_{\alpha',\alpha'} = [K]_{\alpha',\alpha'}(x', x')$, then

$$[\mathcal{C}\left(K^l\right)]_{\alpha,\alpha'}(x, x') = \mathcal{T}_\infty(\Sigma) \tag{38}$$

We choose R large enough so that for $l = 0, \ldots, L$, all $x \in \mathcal{X}$ and α,

$$1/R < \left[K_\infty^l\right]_{\alpha,\alpha}(x, x), \left[\mathcal{A}(K_\infty^l)\right]_{\alpha,\alpha}(x, x) < R, \tag{39}$$

which implies for $0 \leq l \leq L$, K_∞^l and $\mathcal{A}(K_\infty^l)$ are interior points of $\mathrm{PSD}_{|\mathcal{X}|d}(R)$. Let $\Omega(R)$ denote the space of measurable functions with the following properties:

1. **Uniformly Squared Integrable:**

$$\sup_{r \in [1/R, R]} \|\phi\|_{L^2(\mathcal{N}(0,r))} < \infty; \tag{40}$$

2. **Lipschitz Continuity:** there exists $\beta = \beta(\phi, R) > 0$ such that for all $\Sigma, \Sigma' \in \mathrm{PSD}_2(R)$,

$$|\mathcal{T}_\infty(\Sigma) - \mathcal{T}_\infty(\Sigma')| \leq \beta \left\|\Sigma - \Sigma'\right\|_\infty; \tag{41}$$

3. **Uniform Convergence in Probability:** for every $\epsilon > 0$,

$$\sup_{\Sigma \in \mathrm{PSD}_2(R)} P\left(|\mathcal{T}_n(\Sigma) - \mathcal{T}_\infty(\Sigma)| > \epsilon\right) \to 0 \quad \text{as} \quad n \to \infty. \tag{42}$$

It is easy to see that $\Omega(R)$ is a vector space since it is closed under scalar multiplication and addition. The following is the main result of this section.

Theorem A.2. *For $\phi \in \Omega(R)$, $K^L \to K_\infty^L$ in probability and $z_i^L \to \mathcal{N}\left(0, \mathcal{A} \circ (\mathcal{C} \circ \mathcal{A})^L \left(K^0\right)\right)$ in distribution.*

It is not difficult to see that $\Omega(R)$ contains the space of Schwartz functions. Indeed, Daniely et al. (2016) prove (Lemmas 12 and 13) that $\Omega(R)$ contains all C^2-functions, i.e. twice continuously differentiable functions with $\|\phi\|_\infty, \|\phi'\|_\infty, \|\phi''\|_\infty < \infty$ and the ReLu function. Therefore it contains all finite linear combination of such functions. Characterizing the space $\Omega(R)$ and its topology is left for future work.

Proof. Let $\phi \in \Omega(R)$. Note that the affine transform $\mathcal{A}$ is σ_ω^2-Lipschitz and the second property of $\Omega(R)$ implies that the $\mathcal{C}$ operator is $|\mathcal{X}|^2 d^2\beta$-Lipschitz. Thus $\mathcal{C} \circ \mathcal{A}$ is $\bar{\beta}$-Lipschitz with $\bar{\beta} = \sigma_\omega^2 |\mathcal{X}|^2 d^2\beta$. We proceed by induction. Assume $K^l \to K_\infty^l$ in probability as $t \to \infty$. Without loss of generality, we assume $\bar{\beta} \geq 1$. Let $\epsilon > 0$ be sufficiently small so that the $\bar{\epsilon}$-neighborhood of $\mathcal{A}(K_\infty^l)$ is contained in $\mathrm{PSD}_{|\mathcal{X}|d}(R)$, where $\bar{\epsilon} = \max\left\{\epsilon, \sigma_\omega^2 \epsilon/\bar{\beta}\right\}$. Since

$$\begin{aligned}\left\|K_\infty^{l+1} - K^{l+1}\right\|_\infty &\leq \left\|K_\infty^{l+1} - \mathcal{C} \circ \mathcal{A}\left(K^l\right)\right\|_\infty + \left\|\mathcal{C} \circ \mathcal{A}\left(K^l\right) - K^{l+1}\right\|_\infty \\ &\leq \left\|\mathcal{C} \circ \mathcal{A}\left(K_\infty^l\right) - \mathcal{C} \circ \mathcal{A}\left(K^l\right)\right\|_\infty + \left\|\mathcal{C} \circ \mathcal{A}\left(K^l\right) - K^{l+1}\right\|_\infty,\end{aligned}$$

to prove $K^{l+1} \to K_\infty^{l+1}$ in probability, it suffices to show that for every $\delta > 0$, there is a t^* so that for all $t > t^*$,

$$P(\left\|\mathcal{C} \circ \mathcal{A}\left(K_\infty^l\right) - \mathcal{C} \circ \mathcal{A}\left(K^l\right)\right\|_\infty > \epsilon/2) + P\left(\left\|\mathcal{C} \circ \mathcal{A}\left(K^l\right) - K^{l+1}\right\|_\infty > \epsilon/2\right) < \delta. \tag{43}$$

By our induction assumption, there is a t_0 so that for all $t > t_0$

$$p\left(|K_\infty^l - K^l| > \epsilon/2\bar{\beta}\right) < \delta/3. \tag{44}$$

Since $\mathcal{C} \circ \mathcal{A}$ is $\bar{\beta}$-Lipschitz, then

$$P\left(\left\|\mathcal{C} \circ \mathcal{A}\left(K_{\infty}^{l}\right) - \mathcal{C} \circ \mathcal{A}\left(K^{l}\right)\right\|_{\infty} > \epsilon/2\right) < \delta/3\,. \tag{45}$$

To bound the second term in Equation 43, let $U(t)$ denote the event

$$U(t) = \left\{\mathcal{A}\left(K^{l}\right) \in \mathrm{PSD}_{|\mathcal{X}|d}(R)\right\} \tag{46}$$

and $U(t)^c$ denote its complement and

$$[V(t)]_{\alpha,\alpha'}(x,x') = \left\{\left|[\mathcal{C} \circ \mathcal{A}\left(K^{l}\right)]_{\alpha,\alpha'}(x,x') - \left[K^{l+1}\right]_{\alpha,\alpha'}(x,x')\right| > \epsilon/2\right\}. \tag{47}$$

Our assumption implies for all $t > t_0$

$$P\left(U(t)^c\right) < \delta/3. \tag{48}$$

The fact

$$\left\{\left\|\mathcal{C} \circ \mathcal{A}\left(K^{l}\right) - K^{l+1}\right\|_{\infty} > \epsilon/2\right\} \subseteq U(t)^c \bigcup \left(\bigcup_{x,x',\alpha,\alpha'} [V(t)]_{\alpha,\alpha'}(x,x') \bigcap U(t)\right)$$

implies

$$P\left(\left\{\left\|\mathcal{C} \circ \mathcal{A}\left(K^{l}\right) - K^{l+1}\right\|_{\infty} > \epsilon/2\right\}\right) \leq \delta/3 + |\mathcal{X}|^2 d^2 \sup_{x,x',\alpha,\alpha'} P\left([V(t)]_{\alpha,\alpha'}(x,x') \cap U(t)\right).$$

If we use Σ to denote the 2×2 matrix with $\Sigma_{\alpha,\alpha'} = [\mathcal{A}\left(K^{l}\right)]_{\alpha,\alpha'}(x,x')$, $\Sigma_{\alpha,\alpha} = [\mathcal{A}\left(K^{l}\right)]_{\alpha,\alpha}(x,x)$ and $\Sigma_{\alpha',\alpha'} = [\mathcal{A}\left(K^{l}\right)]_{\alpha',\alpha'}(x',x')$, then

$$[\mathcal{C} \circ \mathcal{A}\left(K^{l}\right)]_{\alpha,\alpha'}(x,x') = \mathcal{T}_{\infty}(\Sigma) \tag{49}$$

and $\left[K^{l+1}\right]_{\alpha,\alpha'}(x,x')$ and $\mathcal{T}_{n^{l+1}}(\Sigma)$ have the same distribution. Applying the third property of $\Omega(R)$, we conclude that there is a n so that for all $n^{l+1} \geq n$,

$$\sup_{x,x',\alpha,\alpha'} P\left([V(t)]_{\alpha,\alpha'}(x,x') \cap U(t)\right) \leq \frac{\delta}{3|\mathcal{X}|^2 d^2}$$

and

$$P\left(\left\{\left\|\mathcal{C} \circ \mathcal{A}\left(K^{l}\right) - K^{l+1}\right\|_{\infty} > \epsilon/2\right\} \cap U(t)\right) < 2\delta/3.$$

Therefore we just need to choose $t^* > t^0$ so that $n^{l+1}(t) > n$ for all $t > t^*$. □

A.5 Glossary

We use the following shorthands in this work:

1. NN - neural network;
2. CNN - convolutional neural network;
3. LCN - locally-connected network, a.k.a. convolutional network without weight sharing;
4. FCN - fully connected network, a.k.a. multilayer perceptron (MLP);
5. GP - Gaussian process;
6. X-GP - a GP equivalent to a Bayesian infinitely wide neural network of architecture X (§2).
7. MC-(X-)-GP - a Monte Carlo estimate (§4) of the X-GP.
8. Width, number of filters, number of channels represent the same property for CNNs and LCNs.
9. Pooling - referring to architectures as "with" or "without pooling" means having a single global average pooling layer (collapsing the spatial dimensions of the activations x^{L+1}) before the final linear FC layer outputting the regression outputs z^{L+1}.
10. Invariance and equivariance are always discussed w.r.t. translations in the spatial dimensions.

A.6 EXPERIMENTAL SETUP

Throughout this work we only consider 3×3 (possibly unshared) convolutional filters with stride 1 and no dilation.

All inputs are normalized to have zero mean and unit variance, i.e. lie on the $d-$dimensional sphere of radius $\sqrt{d}$, where d is the total dimensionality of the input.

All labels are treated as regression targets with zero mean. I.e. for a single-class classification problem with C classes targets are $C-$dimensional vectors with $-1/C$ and $(C-1)/C$ entries in incorrect and correct class indices respectively.

If a subset of a full dataset is considered for computational reasons, it is randomly selected in a balanced fashion. No data augmentation is used.

All experiments were implemented in Tensorflow (Abadi et al., 2016) and executed with the help of Vizier (Golovin et al., 2017).

All neural networks are trained using Adam (Kingma & Ba, 2014) minimizing the mean squared error loss.

A.6.1 WIDE CNNS AND LCNS

Relevant figures: 5, 7, 8.

We use a training and validation subsets of CIFAR10 of sizes 500 and 4000 respectively. All images are bilinearly downsampled to 8×8 pixels.

All models have 3 hidden layers with an erf nonlinearity. No ("valid") padding is used.

Weight and bias variances are set to $\sigma_w^2 \approx 1.7562$ and $\sigma_b^2 \approx 0.1841$, corresponding to the preactivation variance fixed point $q^* = 1$ (Poole et al., 2016) for the erf nonlinearity.

NN training proceeds for 2^{19} gradient updates, but aborts if no progress on training loss is observed for the last 100 epochs. If the training loss does not reduce by at least 10^{-4} for 20 epochs, the learning rate is divided by 10.

All computations are done with 32-bit precision.

The following NN parameters are considered[8]:

1. Architecture: CNN or LCN.
2. Pooling: no pooling or a single global average pooling (averaging over spatial dimensions) before the final FC layer.
3. Number of filters: 2^k for k from 0 to 12.
4. Initial learning rate: 10^{-k} for k from 0 to 15.
5. Weight decay: 0 and 10^{-k} for k from 0 to 8.
6. Batch size: 10, 25, 50, 100, 200.

For NNs, all models are filtered to only 100%-accurate ones on the training set and then for each configuration of $\{$architecture, pooling, number of filters$\}$ the model with the lowest validation loss is selected among the configurations of $\{$learning rate, weight decay, batch size$\}$.

For GPs, the same CNN-GP is plotted against CNN and LCN networks without pooling. For LCN with pooling, inference was done with an appropriately rescaled CNN-GP kernel, i.e. $\left(\mathcal{K}_\infty^{\text{vec}} - \sigma_b^2\right)/d + \sigma_b^2$, where d is the spatial size of the penultimate layer. For CNNs with pooling, a Monte Carlo estimate was computed (see §4) with $n = 2^{12}$ filters and $M = 2^6$ samples.

A.6.2 TRANSFORMING A GP OVER SPATIAL LOCATIONS INTO A GP OVER CLASSES

Relevant figure: 3.

[8]Due to time and memory limitations certain large configurations could not be evaluated. We believe this did not impact the results of this work in a qualitative way.

We use the same setup as in §A.6.3, but rescale the input images to size of 31×31, so that at depth 15 the spatial dimension collapses to a 1×1 patch if no padding is used (hence the curve of the CNN-GP without padding halting at that depth).

For MC-CNN-GP with pooling, we use samples of networks with $n = 16$ filters. Due to computational complexity we only consider depths up to 31 for this architecture. The number of samples M was selected independently for each depth among $\left\{2^k\right\}$ for k from 0 to 15 to maximize the validation accuracy on a separate 500-points validation set. This allowed us to avoid the poor conditioning of the kernel.

A.6.3 Monte Carlo Evaluation of Intractable GP Kernels

Relevant figures: 4, 9.

We use the same setup as in §A.6.1, but training and validation sets of sizes 2000 and 4000 respectively.

For MC-GPs we consider the number of channels n (width in FCN setting) and number of NN instantiations M to accept values of 2^k for k from 0 to 10.

Kernel distance is computed as:

$$\frac{\|\mathcal{K}_\infty - K_{n,M}\|_{\mathrm{F}}^2}{\|\mathcal{K}_\infty\|_{\mathrm{F}}^2}, \tag{50}$$

where $\mathcal{K}_\infty$ is substituted with $K_{2^{10},2^{10}}$ for the CNN-GP pooling case (due to impracticality of computing the exact $\mathcal{K}_\infty^{\text{pool}}$).

A.6.4 Relationship to Deep Signal Propagation

Relevant Table: 3.

We use a training and validation subsets of CIFAR10 of sizes 500 and 1000 respectively.

We use the erf nonlinearity. For CNN-GP, images are zero-padded ("same" padding) to maintain the spatial shape of the activations as they are propagated through the network.

Weight and bias variances (horizontal axis σ_w^2 and vertical axis σ_b^2 respectively) are sampled from a uniform grid of size 50×50 on the range $[0.1, 5] \times [0, 2]$ including the endpoints.

If an experiment fails due to a numerical reason it is assigned the 0.1 (random chance) validation accuracy.

All computations are done with 64-bit precision.

A.6.5 CNN-GP on full datasets

Relevant Table: 1, figures 1, 6.

We use full training, validation, and test sets of sizes 50000, 10000, and 10000 respectively for MNIST and Fashion-MNIST, 45000, 5000, and 10000 for CIFAR10. We use validation accuracy to select the best configuration for each model (we do not retrain on valdiation sets).

GPs are computed with 64-bit precision, and NNs are trained with 32-bit precision.

Zero-padding ("same") is used.

The following parameters are considered:

1. Architecture: CNN of FCN.
2. Nonlinearity: erf or ReLU.
3. Depth: 2^k for k from 0 to 4 (and up to 2^5 for MNIST and Fashion-MNIST datasets).
4. Weight and bias variances. For erf: q^* from $\{0.1, 1, 2, \ldots, 8\}$. For ReLU: a fixed weight variance $\sigma_\omega^2 = 2$ and bias variance σ_b^2 from $\{0.1, 1, 2, \ldots, 8\}$.

On CIFAR10, we additionally train NNs for 2^{18} gradient updates with a batch size of 128 with corresponding parameters in addition to[9]

1. Pooling: no pooling or a single global average pooling (averaging over spatial dimensions) before the final FC layer (only for CNNs).
2. Number of filters or width: 2^k for k from 1 to 9 (and up to 2^{10} for CNNs with pooling in Figure 1).
3. Learning rate: $10^{-k} \times 2^{16} / (\text{width} \times q^*)$ for k from 5 to 9, where width is substituted with the number of filters for CNNs and q^* is substituted with σ_b^2 for ReLU networks. "Small learning rate" in Table 1 refers to $k \in \{8, 9\}$.
4. Weight decay: 0 and 10^{-k} for k from 0 to 5.

For NNs, all models are filtered to only 100%-accurate ones on the training set (expect for when the value in parentheses in Table 1) is reported). The reported values are then reported for models that achieve the best validation accuracy.

A.6.6 Model comparison on CIFAR10

Relevant Table 2.

We use the complete CIFAR10 dataset as described in §A.6.5 and consider 8-layer ReLU models with weight and bias variances of $\sigma_w^2 = 2$ and $\sigma_b^2 = 0.01$. The number of filters / width is set to 2^5, 2^{10} and 2^{12} for LCN, CNN, and FCN respectively.

GPs are computed with 64-bit precision, and NNs are trained with 32-bit precision.

No padding ("valid") is used.

NN training proceeds for 2^{18} gradient updates with batch size 64, but aborts if no progress on training loss is observed for the last 10 epochs. If the training loss does not reduce by at least 10^{-4} for 2 epochs, the learning rate is divided by 10.

Values for NNs are reported for the best validation accuracy over different learning rates (10^{-k} for k from 2 to 12) and weight decay values (0 and 10^{-k} for k from 2 to 7).For GPs, validation accuracy is maximized over initial diagonal regularization terms applied to the training convariance matrix: $10^{-k} \times$ [mean of the diagonal] for k among 2, 4 and 9 (if the cholesky decompisition fails, the regularization term is increased by a factor of 10 until it succeeeds).

[9]Due to time and compute limitations certain large configurations could not be evaluated. We believe this did not impact the results of this work in a qualitative way.