Tensor Network-Constrained Kernel Machines as Gaussian Processes

Frederiek Wesel
Delft Center for Systems and Control
Delft University of Technology
The Netherlands

Kim Batselier
Delft Center for Systems and Control
Delft University of Technology
The Netherlands

Abstract

In this paper we establish a new connection between *Tensor Network* (TN)-constrained kernel machines and *Gaussian Processes* (GPs). We prove that the outputs of *Canonical Polyadic Decomposition* (CPD) and *Tensor Train* (TT)-constrained kernel machines converge in the limit of large ranks to the same product kernel GP which we fully characterize, when specifying appropriate i.i.d. priors across their components. We show that TT-constrained models convergence faster to the GP compared to their CPD counterparts for the same number of model parameters. The convergence to the GP occurs as the ranks tend to infinity, as opposed to the standard approach which introduces TNs as an additional constraint on the posterior. This implies that the newly established priors allow the models to learn features more freely as they necessitate infinitely more parameters to converge to a GP, which is characterized by a fixed learning representation and thus no feature learning. As a consequence, the newly derived priors yield more flexible models which can better fit the data, albeit at increased risk of overfitting. We demonstrate these considerations by means of two numerical experiments.

1 INTRODUCTION

Tensor Networks [TNs, Cichocki, 2014; Cichocki et al., 2016, 2017], a tool from multilinear algebra, extend the concept of rank from matrices to tensors allowing to represent an exponentially large object with a linear number of parameters. As such, TNs have been used to reduce the storage and computational complexities by compressing the model parameters of a range of models such as *Deep Neural Networks* (DNNs) [Novikov et al., 2015], *Convolutional Neural Networks* (CNNs) [Jaderberg et al., 2014; Lebedev et al., 2015], *Recurrent Neural Networks* (RNNs) [Ye et al., 2018], *Graph Neural Networks* (GNNs) [Hua et al., 2022] and transformers [Ma et al., 2019].

Similarly, TNs have also found application in the context of kernel machines for supervised learning [Stoudenmire and Schwab, 2016; Novikov et al., 2018; Wesel and Batselier, 2021] as an additional constraint on the model posterior. Such models learn a low-rank nonlinear data-dependent representation from an exponentially large number of fixed features by means of a restricted number of parameters, and are as such characterized by an implicit source of regularization. Furthermore, storage and the evaluation of the model and its gradient require a linear complexity in the number of parameters, rendering these methods promising candidates for applications requiring both good generalization and scalability. Because of their intrinsic nonlinearity which prohibits closed-form Bayesian inference, the training of these models is typically accomplished in the *maximum likelihood* (ML) and *maximum a posteriori* (MAP) framework where the low-rank TN assumption is introduced as an additional nonlinear constraint in the optimization problem. In this setting the ensuing estimator recovers the solution that would be obtained without the TN constraint when the low-rank assumption is satisfied *exactly*, i.e. for *finite rank*.

In contrast, *Gaussian Processes* [GPs, Rasmussen and Williams, 2006] are an established framework for modeling functions which naturally allows the practitioner to incorporate prior knowledge. When considering i.i.d. observations and Gaussian likelihoods, GPs allow for the determination of the posterior in *closed-form*, which considerably facilitates tasks such as inference, sampling and the construction of sparse approximations among many others.

In this paper we establish a direct connection between

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

TN-constrained kernel machines and GPs, thus solving an open question considered by Wesel and Batselier [2021, 2023]. We prove that the outputs of *Canonical Polyadic Decomposition* (CPD) and *Tensor Train* (TT)-constrained kernel machines converge in the limit of large ranks to the same product kernel GP which we fully characterize, when specifying appropriate i.i.d. priors across their components. This result allows us to derive that when placing such priors on their parameters, TT-constrained models achieve faster convergence to the GP compared to their CPD counterparts for the same number of model parameters. The convergence to the GP occurs as the ranks tend to infinity, as opposed to the standard approach which introduces TNs as an additional constraint on the model posterior. This implies that the newly established priors allow the models to more autonomously learn features as they necessitate infinitely more parameters to converge to a GP, which is characterized by a *fixed* learning representation and thus no feature learning. Consequently, the newly derived priors yield more flexible models which can better fit the data and have a higher chance of overfitting. We showcase the convergence properties of both newly derived priors and their effect on MAP estimation by means of numerical experiments.

The rest of this paper is organized as follows. In section 2 we provide a brief introduction to GPs and their approximations, TNs and TN-constrained kernel machines. In section 3 we present our main result, i.e. the equivalence in the limit between TN-constrained kernel machines and product kernel GPs. In section 4 we showcase the different convergence rates to the GP of CPD and TT and their effect on MAP estimation. We then provide a review of related work (section 5) and a conclusion (section 6). We discuss the notation used throughout the paper in appendix A.1.

2 BACKGROUND

GPs are a collection of random variables such that any finite subset has a joint Gaussian distribution [Rasmussen and Williams, 2006]. They provide a flexible formalism for modeling functions which inherently allows for the incorporation of prior knowledge and the production of uncertainty estimates in the form of a predictive distribution. More specifically, a GP is fully specified by a mean function $\mu(\cdot) : \mathbb{R}^D \to \mathbb{R}$, typically chosen as zero, and a covariance or kernel function $k(\cdot, \cdot) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$:

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \cdot)).$$

Given a labeled dataset $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ consisting of N inputs $\boldsymbol{x}_n \in \mathbb{R}^D$ and i.i.d. noisy observations $y_n \in \mathbb{R}$, GPs can be used for modeling the underlying function f in classification or regression tasks by specifying a likelihood function. For example the likelihood

$$p(y_n \mid f(\boldsymbol{x}_n)) = \mathcal{N}(f(\boldsymbol{x}_n), \sigma^2), \tag{1}$$

yields a GP posterior which can be obtained in closed-form by conditioning the prior GP on the noisy observations. Calculating the mean and covariance of such a posterior crucially requires instantiating and formally inverting the kernel matrix $\boldsymbol{K}$ such that $k_{n,m} \coloneqq k(\boldsymbol{x}_n, \boldsymbol{x}_m)$. These operations respectively incur a computational cost of $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$ and therefore prohibit the processing of large-sampled datasets.

2.1 Basis Function Approximation

Aside from variational inference [Titsias, 2009; Hensman et al., 2013] and iterative methods Wilson and Nickisch [2015], a common approach in literature to circumvent the $\mathcal{O}(N^3)$ computational bottleneck is to project the GP onto a finite number of *Basis Functions* (BFs) [e.g., Rasmussen and Williams, 2006; Quiñonero-Candela and Rasmussen, 2005]. This is achieved by approximating the kernel as

$$k(\boldsymbol{x}, \boldsymbol{x'}) \approx \boldsymbol{\varphi}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}\ \boldsymbol{\varphi}(\boldsymbol{x'}), \tag{2}$$

where here $\boldsymbol{\varphi}(\boldsymbol{x}) : \mathbb{R}^D \to \mathbb{R}^M$ are (nonlinear) basis functions and $\boldsymbol{\Lambda} \in \mathbb{R}^{M \times M}$ are the BF weights. This finite-dimensional kernel approximation ensures a *degenerate* kernel [Rasmussen and Williams, 2006], as it characterized by a finite number of non-zero eigenvalues. Its associated GP can be characterized equivalently as

$$f(\boldsymbol{x}) = \langle \boldsymbol{\varphi}(\boldsymbol{x}), \boldsymbol{w} \rangle, \quad \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Lambda}), \tag{3}$$

wherein $\boldsymbol{w} \in \mathbb{R}^M$ are the model weights and $\boldsymbol{\Lambda}$ is the associated prior covariance. Once more considering a Gaussian likelihood (equation (1)) yields a closed-form posterior GP whose mean and covariance require only the posterior covariance matrix $(\sum_{n=1}^N \boldsymbol{\varphi}(\boldsymbol{x}_n)\boldsymbol{\varphi}(\boldsymbol{x}_n)^{\mathrm{T}} + \boldsymbol{\Lambda}^{-1})^{-1}$. This yields a computational complexity of $\mathcal{O}(NM^2 + M^3)$, which allows to tackle large-sampled data when $N \gg M$.

2.2 Product Kernels

In the remainder of this paper we consider GPs with product kernels.

Definition 2.1 (Product kernel [Rasmussen and Williams, 2006])**.** A kernel $k(\boldsymbol{x}, \boldsymbol{x'})$ is a product kernel if

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \prod_{q=1}^{Q} k^{(q)}(\boldsymbol{x}, \boldsymbol{x'}), \tag{4}$$

where each $k^{(q)}(\cdot, \cdot) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ is a valid kernel.

While many commonly used kernels are product kernels e.g. the Gaussian kernel and the polynomial kernel, product kernels provide a straightforward strategy to extend one-dimensional kernels to the higher-dimensional case [Rasmussen and Williams, 2006; Hensman et al., 2017]. The basis functions and prior covariance of product kernels can then be determined based on the basis function expansion of their constituents as follows.

Lemma 2.2 (Basis functions and prior covariances of product kernels)**.** *Consider the product kernel of definition 2.1. Denote the basis functions and prior covariance of each factor $k^{(q)}(\boldsymbol{x}, \boldsymbol{x'})$ as $\boldsymbol{\varphi}^{(q)}(\boldsymbol{x}) \in \mathbb{R}^{M_q}$ and $\boldsymbol{\Lambda}^{(q)} \in \mathbb{R}^{M_q \times M_q}$ respectively, then the basis functions and prior covariance of $k(\boldsymbol{x}, \boldsymbol{x'})$ are*

$$\boldsymbol{\varphi}(\boldsymbol{x}) = \otimes_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x}), \tag{5}$$

and

$$\boldsymbol{\Lambda} = \otimes_{q=1}^{Q} \boldsymbol{\Lambda}^{(q)}, \tag{6}$$

The inherent challenge in this approach stems from the exponential increase of the number of basis functions M and thus of model parameters as a function of the dimensionality of the input data, thereby restricting their utility to low-dimensional datasets.

Such structure arises for instance when dealing with Mercer expansions of product kernels, in the structured kernel interpolation framework [Wilson and Nickisch, 2015; Yadav et al., 2021] variational Fourier features framework [Hensman et al., 2017] and Hilbert-GP framework [Solin and Särkkä, 2020]. Alternative important approximation strategies which avoid this exponential scaling are random features [Rahimi and Recht, 2007; Lázaro-Gredilla et al., 2010], inducing features [Csató and Opper, 2002; Seeger et al., 2003; Quiñonero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2006; Hensman et al., 2013, 2015] and additive GPs [Duvenaud et al., 2011; Lu et al., 2022] which circumvent the outlined computational issue. All those approaches can be interpreted as projecting the GP on a set of BFs.

The performance of these methods however tends to deteriorate in higher dimensions, as they need to cover an exponentially large domain with a linear number of random samples or inducing points. These issues are some of the computational aspects of the *curse of dimensionality*, which renders it difficult to operate in high-dimensional feature spaces [Hastie et al., 2001].

2.3 Tensor Networks

A recent alternative approach to remedy said curse of dimensionality affecting the exponentially increasing weights of the linear model in equation (3) consists in constraining the tensorized models weights $\texttt{ten}(\boldsymbol{w})$ to be a low-rank tensor network. TNs express a Q-dimensional tensor $\boldsymbol{\mathcal{W}}$ as a multi-linear function of a number of *core* tensors. Two commonly used TNs are the CPD and TT, defined as follows.

Definition 2.3 (CPD [Hitchcock, 1927])**.** A Q-dimensional tensor $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q}$ has a rank-R CPD if

$$w_{m_1, m_2, \ldots, m_Q} = \sum_{r=1}^{R} \prod_{q=1}^{Q} {w^{(q)}}_{m_q, r}. \tag{7}$$

The cores of a CPD are the matrices $\boldsymbol{W}^{(q)} \in \mathbb{R}^{M_q \times R}$. Since a CPD tensor can be expressed solely in terms of its cores, its storage requires $P_{\texttt{CPD}} = R \sum_{q=1}^{Q} M_q$ parameters as opposed to $\prod_{q=1}^{Q} M_q$.

Definition 2.4 (TT [Oseledets, 2011])**.** A Q-dimensional tensor $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q}$ admits a rank-$(R_0 \coloneqq 1, R_1, \ldots, R_Q \coloneqq 1)$ tensor train if

$$w_{m_1, m_2, \ldots, m_Q} = \sum_{r_0=1}^{R_0} \sum_{r_1=1}^{R_1} \cdots \sum_{r_Q=1}^{R_Q} \prod_{q=1}^{Q} {w^{(q)}}_{r_{q-1}, m_q, r_q}. \tag{8}$$

The cores of a tensor train are Q 3-dimensional tensors $\boldsymbol{\mathcal{W}}^{(q)} \in \mathbb{R}^{R_{q-1} \times M_q \times R_q}$ which yield $P_{\texttt{TT}} = \sum_{q=1}^{Q} R_{Q-1} M_Q R_Q$ parameters.

In the following we denote by $\texttt{TN}(\boldsymbol{\mathcal{W}})$ a tensor which admits a general TN format, by $\texttt{CPD}(\boldsymbol{\mathcal{W}})$ a tensor which is a rank-R CPD form and by $\texttt{TT}(\boldsymbol{\mathcal{W}})$ a tensor in rank-$(R_0 \coloneqq 1, R_1, \ldots, R_Q \coloneqq 1)$ TT form. Lastly, we denote by $\texttt{R}_1(\boldsymbol{\mathcal{W}})$ a tensor which is rank-1 CPD form or rank-$(1, 1, \ldots, 1)$ TT, as both are equivalent. Importantly, we refer to a tensor in general TN format $\texttt{TN}(\boldsymbol{\mathcal{W}}) \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q}$ as *underparametrized* if its rank hyperparameters, e.g. R in case of CPD, are chosen such that its storage cost is less than $\prod_{q=1}^{Q} M_q$. This is crucial in order to obtain storage and computational benefits.

2.4 Tensor Network-Constrained Kernel Machines

TNs have been used to reduce the number of model parameters in kernel machines (equation (3)) by tensorizing the BFs $\boldsymbol{\varphi}(\cdot)$ and model weights $\boldsymbol{w}$ and by constraining both to be underparameterized TNs. This approach lays its foundations on the fact that the Frobenius inner product of a tensorized vector is isometric with respect to the Euclidean inner product, i.e.

$$f(\boldsymbol{x}) = \langle \boldsymbol{\varphi}(\boldsymbol{x}), \boldsymbol{w} \rangle = \langle \texttt{ten}(\boldsymbol{\varphi}(\boldsymbol{x})), \texttt{ten}(\boldsymbol{w}) \rangle_{\mathrm{F}}. \tag{9}$$

This isometry allows then to constrain the BFs and the model weights to be an underparameterized TN. Since product kernels yield an expansion in terms of

Kronecker-product BFs (equation (5)), they are a rank-1 TN by definition after tensorization. Embedding these relations yields an approximate model

$$f(\boldsymbol{x}) \approx f_{\texttt{TN}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\,(\boldsymbol{\varphi}(\boldsymbol{x}))), \texttt{TN}(\texttt{ten}\,(\boldsymbol{w}))\rangle_{\mathrm{F}}, \tag{10}$$

characterized by lower storage and computational complexities. This approach has been proposed mostly for weights modeled as CPD [Kargas and Sidiropoulos, 2021; Wesel and Batselier, 2021, 2023] or TT [Wahls et al., 2014; Stoudenmire and Schwab, 2016; Batselier et al., 2017; Novikov et al., 2018; Chen et al., 2018] as they arguably introduce fewer rank hyperparameters (only one in case of CPD) and thus are in practice easier to work with compared to other TNs such as the *Multi-Scale Entanglement Renormalization Ansatz* (MERA) [Reyes and Stoudenmire, 2021].

We define such models as we will need them in detail in the next section, where we present our main contribution.

Definition 2.5 (CPD-constrained kernel machine)**.** The CPD-constrained kernel machine is defined as

$$f_{\texttt{CPD}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\,(\boldsymbol{\varphi}(\boldsymbol{x}))), \texttt{CPD}(\texttt{ten}\,(\boldsymbol{w}))\rangle_{\mathrm{F}} \tag{11}$$

$$= \sum_{r=1}^{R} h_r(\boldsymbol{x}), \tag{12}$$

where the intermediate variables $h_r \in \mathbb{R}$ are defined as

$$h_r(\boldsymbol{x}) \coloneqq \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{w}^{(q)}{}_{:,r}. \tag{13}$$

Similarly, we provide a definition for the TT-constrained kernel machine.

Definition 2.6 (TT-constrained kernel machine)**.** The TT-constrained kernel machine is defined as

$$f_{\texttt{TT}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\,(\boldsymbol{\varphi}(\boldsymbol{x}))), \texttt{TT}(\texttt{ten}\,(\boldsymbol{w}))\rangle_{\mathrm{F}} \tag{14}$$

$$= \sum_{r_Q=1}^{R_Q} \sum_{r_{Q-1}=1}^{R_{Q-1}} \cdots \sum_{r_0=1}^{R_0} \prod_{q=1}^{Q} z^{(q)}_{r_{q-1},r_q}(\boldsymbol{x}), \tag{15}$$

where the intermediate variables $\boldsymbol{Z}^{(q)} \in \mathbb{R}^{R_{q-1}\times R_q}$ are defined element-wise as

$$z^{(q)}_{r_{q-1},r_q}(\boldsymbol{x}) \coloneqq \sum_{m_q=1}^{M_q} \varphi^{(q)}_{m_q}(\boldsymbol{x}) w^{(q)}_{r_{q-1},m_q,r_q}. \tag{16}$$

Evaluating CPD and TT-constrained kernel machines (equation (11), equation (14)) and their gradients can be accomplished with $\mathcal{O}(P_{\texttt{CPD}})$ and $\mathcal{O}(P_{\texttt{TT}})$ computations, respectively. This allows the practitioner to tune the rank hyperparameter in order to achieve a model that fits in the computational budget at hand and that learns from the specified BFs.

From an optimization point-of-view, models in the form of equation (10) are considered in the ML [Stoudenmire and Schwab, 2016; Batselier et al., 2017] and in the MAP setting [Wahls et al., 2014; Novikov et al., 2018; Chen et al., 2018; Kargas and Sidiropoulos, 2021; Wesel and Batselier, 2021, 2023] and in the context of GP variational inference [Izmailov et al., 2018] where TTs are used to parameterize the variational distribution. In all these scenarios, TNs appear as *an additional constraint to the optimization problem*, and do hence not *define* a probabilistic model but merely approximate the chosen estimator (ML, MAP, etc.).

In the following section we present the main contribution of our work: we show how when placing i.i.d. priors on the cores of these CPD and TT-constrained model, they converge to a GP which we fully characterize. As we will see, beside connecting the TN-constrained kernel machines with GPs, this probabilistic characterization defines a different and less stringent type of regularization for such models.

3 TN-CONSTRAINED KERNEL MACHINES AS GPs

We commence to outline the correspondence between TN-constrained kernel machine and GPs, which makes use of the *Central Limit Theorem* (CLT). We begin by elucidating the simplest case, i.e. the CPD.

Theorem 3.1 (CPD-constrained kernel machine as GP)**.** *Consider the CPD-constrained kernel machine*

$$f_{\texttt{CPD}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\,(\boldsymbol{\varphi}(\boldsymbol{x}))), \texttt{CPD}(\texttt{ten}\,(\boldsymbol{w}))\rangle_{\mathrm{F}}.$$

If each of the R columns $\boldsymbol{w}^{(q)}{}_{:,r} \in \mathbb{R}^{M_q}$ of each CPD core is an i.i.d. random variable such that

$$\mathbb{E}\left[\boldsymbol{w}^{(q)}_{:,r}\right] = \boldsymbol{0},$$
$$\mathbb{E}\left[\boldsymbol{w}^{(q)}_{:,r}\boldsymbol{w}^{(q)\mathrm{T}}_{:,r}\right] = R^{-\frac{1}{Q}}\boldsymbol{\Lambda}^{(q)},$$

then $f_{\texttt{CPD}}(\boldsymbol{x})$ converges in distribution as $R \to \infty$ to the GP

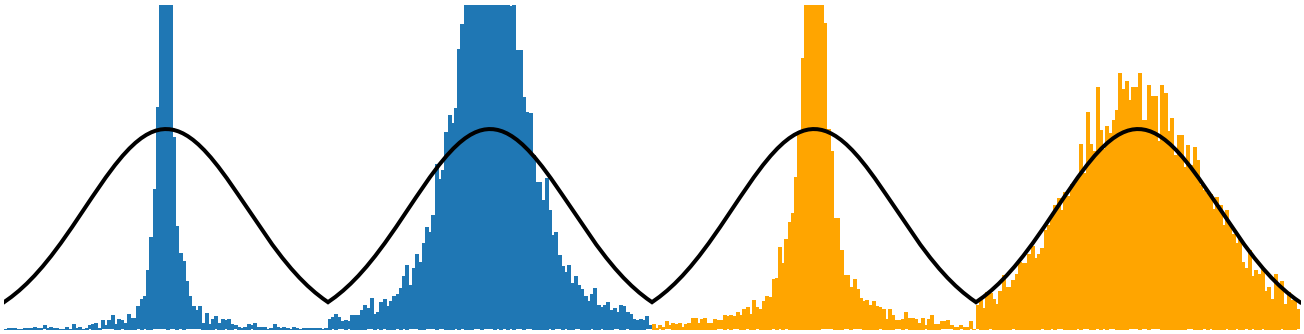
$$f_{\texttt{CPD}}(\boldsymbol{x}) \sim \mathcal{GP}\left(0, \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\cdot)\right).$$

Proof. See appendix B.1. □

A similar result can be constructed for the TT case.

Theorem 3.2 (TT-constrained kernel machine as GP)**.** *Consider the TT-constrained kernel machine*

$$f_{\texttt{TT}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\,(\boldsymbol{\varphi}(\boldsymbol{x}))), \texttt{TT}(\texttt{ten}\,(\boldsymbol{w}))\rangle_{\mathrm{F}}$$



(a) $P_{\texttt{CPD}} = 1600$ (b) $P_{\texttt{CPD}} = 1\,600\,000$ (c) $P_{\texttt{TT}} = 1320$ (d) $P_{\texttt{TT}} = 1\,605\,000$

Figure 1: Histograms of the empirical PDF of CPD (blue) and TT (orange) models specified in theorem 3.1 and 3.2 evaluated at a random point as a function of model parameters P for $Q = 16$. The black line is the PDF of the GP. Notice how TT converges faster to the GP for the same number of model parameters P.

If each of the $R_{q-1}R_q$ fibers $\boldsymbol{W}^{(q)}{}_{r_{q-1},:,r_q} \in \mathbb{R}^{M_q}$ of each TT core is an i.i.d. random variable such that

$$\mathbb{E}\left[\boldsymbol{W}^{(q)}_{r_{q-1},:,r_q}\right] = \mathbf{0},$$
$$\mathbb{E}\left[\boldsymbol{W}^{(q)}_{r_{q-1},:,r_q}\boldsymbol{W}^{(q)}_{r_{q-1},:,r_q}{}^{\mathrm{T}}\right] = \frac{1}{\sqrt{R_{q-1}R_q}}\boldsymbol{\Lambda}^{(q)},$$

then $f_{\texttt{TT}}(\boldsymbol{x})$ converges in distribution as sequentially $R_1 \to \infty$, $R_2 \to \infty$, ..., $R_{Q-1} \to \infty$ to the Gaussian process

$$f_{\texttt{TT}}(\boldsymbol{x}) \sim \mathcal{GP}\left(0, \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{\Lambda}^{(q)}\boldsymbol{\varphi}^{(q)}(\cdot)\right).$$

Proof. See appendix B.2. □

Theorem 3.2 guarantees the convergence in distribution of $f_{\texttt{TT}}(\boldsymbol{x})$ to the GP of equation (3) by taking successive limits of each TT rank. Importantly, the same convergence results also holds true if the TT ranks grow simultaneously, see appendix B.3.

Both theorem 3.1 and 3.2 are remarkable, as they imply that a GP which can be defined in terms of a finite number of $\prod_{q=1}^{Q} M_q$ weights $\boldsymbol{w}$ can be also obtained with an *infinite* number of model parameters P using the CPD-constrained model of definition 2.5 or the TT-constrained model of definition 2.6. Furthermore, theorem 3.1 and 3.2 suggest that when the priors of theorem 3.1 and 3.2 are placed on the model weights, CPD and TT-based models exhibit GP behavior in the overparameterized regime as their ranks tend towards infinity. GP behavior is characterized by a fixed learning representation $\boldsymbol{\varphi}(\cdot)$, which in case of the kernel in theorem 3.1 and 3.2 is fully defined by the BFs and is hence data-independent. On the contrary, as we will see, in the finite rank regime both CPD and TT models are able to craft nonlinear features from the provided BFs, learning nonlinear latent patterns in the mapped data.

3.1 Convergence Rate to the GP

While both theorem 3.1 and theorem 3.2 guarantee convergence in distribution to the GP of equation (3), they do so at rates that differ in terms of the number of model parameters. Let us assume, for simplicity, that the number of basis functions is the same along each dimension, i.e., M, and that the $Q-1$ TT ranks equal R. It follows then that the number of CPD model parameters $P_{\texttt{CPD}} = MQR_{\texttt{CPD}}$ and the number of TT model parameters $P_{\texttt{TT}} = M(Q-2)R^2_{\texttt{TT}} + 2MR_{\texttt{TT}} = \mathcal{O}(MQR^2_{\texttt{TT}})$. Given the convergence rate of the CLT for the expression in equation (11) to the GP in equation (3), denoted as $\mathcal{O}(1/\sqrt{R_{\texttt{CPD}}})$ with respect to the variable $P_{\texttt{CPD}}$, we can establish the following corollary by substituting $R_{\texttt{CPD}}$ as a function of $P_{\texttt{CPD}}$.

Corollary 3.3 (Convergence rate for CPD)**.** *Under the conditions of theorem 3.1, the function $f_{\texttt{CPD}}(\boldsymbol{x})$ converges in distribution to the GP defined by equation* (3)*. The convergence rate is given by:*

$$f_{\texttt{CPD}}(\boldsymbol{x}) \to \mathcal{O}\left(\left(\frac{MQ}{P_{\texttt{CPD}}}\right)^{\frac{1}{2}}\right).$$

Due to their hierarchical structure, TT models are a composition of $R_{\texttt{TT}}^{Q-1}$ variables, but can be represented in a quadratic number of model parameters in $R_{\texttt{TT}}$, since $P_{\texttt{TT}} = \mathcal{O}(MQR^2_{\texttt{TT}})$. Expressing then the CLT convergence rate of $\mathcal{O}(1/\sqrt{R_{\texttt{TT}}^{Q-1}})$ as a function of $P_{\texttt{TT}}$ yields the following corollary.

Corollary 3.4 (Convergence rate for TT)**.** *Under the conditions of theorem 3.2, the function $f_{\texttt{TT}}(\boldsymbol{x})$ converges in distribution to the GP defined by equation* (3)*. The*

convergence rate is given by:

$$f_{\texttt{TT}}(\boldsymbol{x}) \to \mathcal{O}\left(\left(\frac{MQ}{P_{\texttt{TT}}}\right)^{\frac{Q-1}{4}}\right).$$

Therefore, when dealing with identical models in terms of the number of basis functions (M), product kernel terms (Q), and the number of model parameters ($P_{\texttt{CPD}} = P_{\texttt{TT}}$), $f_{\texttt{TT}}(\boldsymbol{x})$ will converge at a polynomially faster rate than $f_{\texttt{CPD}}(\boldsymbol{x})$, thus exhibiting GP behavior with a reduced number of model parameters. In particular, based on corollaries 3.3 and 3.4 we expect the GP convergence rate of TT models to be faster for $Q \geq 3$.

These insights are relevant for practitioners engaged with TN-constrained kernel machines, as they shed light on the balance between the GP and (deep) neural network behavior inherent in these models. Notably, when using the priors of theorem 3.1 and 3.2 CPD and TT-constrained models, akin to shallow and DNNs respectively, have the capacity to craft additional nonlinearities beyond the provided basis functions. This characteristic can result in more expressive model when dealing with a limited number of parameters. However, as the parameter count increases, we expect these models to transition towards GP behavior, characterized by a fixed feature representation and static in comparison.

3.2 Consequences for MAP Estimation

As discussed in section 2.4, TN-constrained kernel machines are typically trained in the ML or MAP framework by constraining the weights $\boldsymbol{w}$ in the log-likelihood or log-posterior to be a TN. In said MAP context, and e.g. when specifying a normal prior on the model weights $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Lambda})$, the resulting regularization term Ω (log-prior) is approximated by $\Omega_{\texttt{TN}}$ as

$$\Omega \coloneqq \left|\left|\boldsymbol{\Lambda}^{-\frac{1}{2}}\boldsymbol{w}\right|\right|_{\text{F}}^2 \approx \Omega_{\texttt{TN}} \coloneqq \left|\left|\texttt{TN}(\texttt{ten}\left(\boldsymbol{\Lambda}^{-\frac{1}{2}}\boldsymbol{w}\right))\right|\right|_{\text{F}}^2,$$

where $\boldsymbol{\Lambda} = \otimes_{q=1}^{Q}\boldsymbol{\Lambda}^{(q)}$. For example, in case of CPD-constrained models we have

$$\Omega_{\texttt{CPD}} = \left|\left|\odot_{q=1}^{Q}\left(\boldsymbol{W}^{(q)^{\text{T}}}\boldsymbol{\Lambda}^{(q)^{-1}}\boldsymbol{W}^{(q)}\right)\right|\right|_{\text{F}}^2. \quad (17)$$

This form of regularization is considered for TT by Wahls et al. [2014]; Novikov et al. [2018]; Chen et al. [2018] and for CPD by Wesel and Batselier [2021, 2023]. It provides a Frobenius norm approximation of the regularization term which recovers the original MAP estimate as the hyperparameters of $\texttt{TN}(\texttt{ten}\,(\Lambda\boldsymbol{w}))$ are chosen such that $\texttt{TN}(\texttt{ten}\,(\boldsymbol{\Lambda}\boldsymbol{w})) = \texttt{ten}\,(\boldsymbol{\Lambda}\boldsymbol{w})$. If we now consider the log-posterior of theorem 3.1 we end up with

$$\Omega_{\texttt{CPD}} \coloneqq R^{\frac{1}{Q}}\sum_{q=1}^{Q}\left|\left|\boldsymbol{\Lambda}^{(q)^{-\frac{1}{2}}}\boldsymbol{W}^{(q)}\right|\right|_{\text{F}}^2. \quad (18)$$

This regularization has been employed without the scaling factor $R^{\frac{1}{Q}}$ and with $\boldsymbol{\Lambda}^{(q)} = \text{I}_{M_q}$ in the work of Kargas and Sidiropoulos [2021], who may not have been aware of the underlying connection with GPs at that time. Contrary to the regularization $\Omega_{\texttt{TN}}$ in equation (17), it provides an approximation which recovers the log-prior Ω and thus the MAP, which in combination with a Gaussian likelihood and squared-loss is equivalent to the GP posterior mean *in the limit* of large ranks. These considerations point to the fact that if the practitioner is interested only in a MAP estimate which recovers the GP posterior mean as faithfully as possible given the computational budget at hand, he might be more interested in the established regularization of equation (17). On the contrary, the choice of equation (18) in combination with squared-loss recovers the GP MAP *in the limit*, yielding models that can fit the data more closely albeit with an increased possibility of overfitting with respect to the associated GP baseline. Furthermore, sampling the priors in theorem 3.1 and 3.2 provide a sensible initial guess for gradient-based optimization which adjusts to the dimensionality of the inputs and the choice of rank hyperparameters [Barratt et al., 2021].

4 NUMERICAL EXPERIMENTS

We setup two numerical experiments in order to respectively empirically observe the claims in theorem 3.1 and 3.2 by evaluating the convergence to the prior GP in equation (3), and to evaluate the GP behavior of such models at prediction in the finite rank case. In all experiments we made use of the *Hilbert–space Gaussian Process* (HGP) Solin and Särkkä [2020] BFs which approximate stationary tensor product kernels of the form $\prod_{q=1}^{D} k(x_q, x_q')$, and opt for $M_q = 10$ basis functions per dimension. We ran all experiments on a Dell Inc. Latitude 7410 laptop computer with 16 GB of RAM. The Python implementation is available at `github.com/fwesel/tensorGP`.

4.1 GP Convergence

In order to empirically verify the convergence to the GP of equation (3) we sample 10 000 instances of the CPD and TT models specified in theorem 3.1 and 3.2 for increasing CPD and TT ranks yielding up to $P = 10\,000$ model parameters. Since the target distribution is Gaussian with known moments, we record the Cramér–von Mises statistic W^2 [D'Agostino and Stephens, 1986] which gives a metric of closeness between the target and our sampled empirical CDF. We repeat this for 10 randomly sampled data points and for $Q = 2, 4, 8, 16$ and report the mean and standard deviation of the results in figure 2. Therein it can be ob-

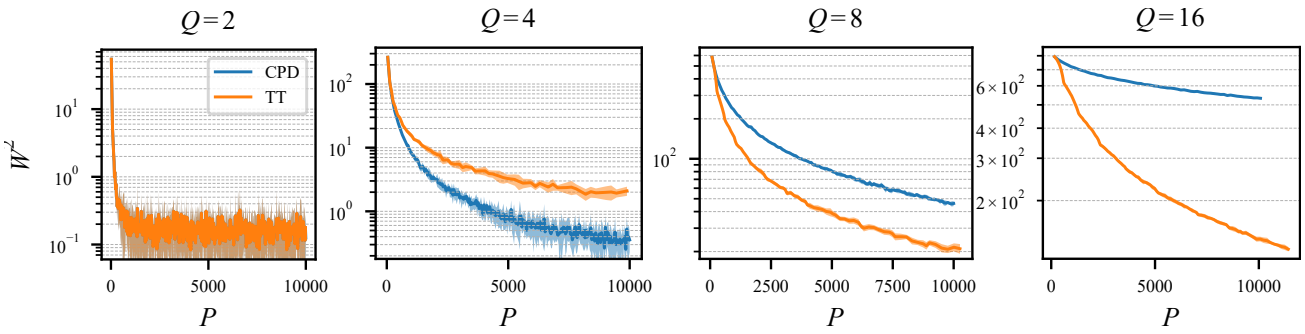


Figure 2: Mean and standard deviation of the Cramér–von Mises statistic W^2 evaluated between the empirical CDF of CPD and TT models specified in theorem 3.1 and 3.2 evaluated at $N = 10$ random points as a function of model parameters P for $Q = 2, 4, 8, 16$. The two models are equivalent for $Q = 2$. Notice how TT converges faster to the GP as Q increases.

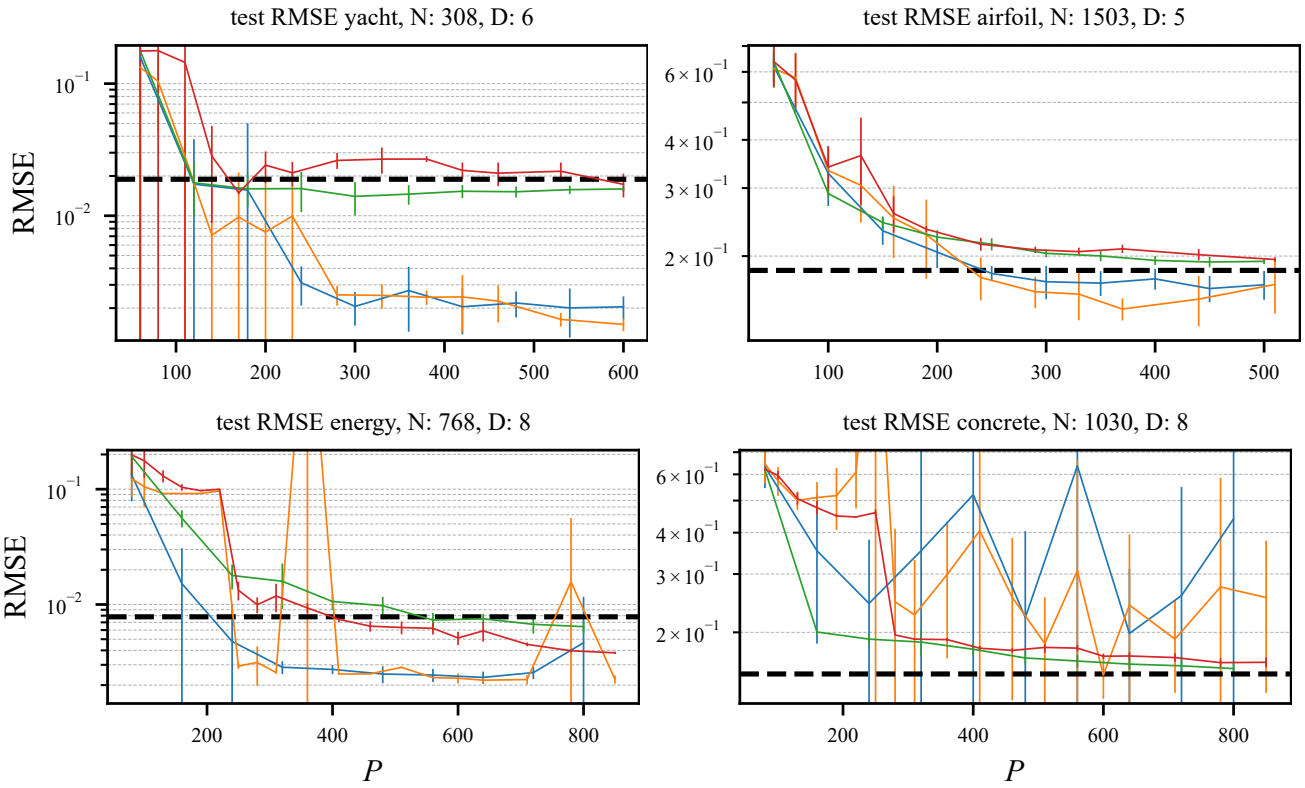


Figure 3: Mean and standard deviation of the test RMSE of CPD and TT models for regularization equations (17) and (18) (green and red curves respectively) as a function of model parameters P as well as their target KRR (dotted line). In the plots, the probabilistic regularization of equation (18) and its TT counterpart are denoted by a blue and orange line respectively. The dotted line corresponds to the KRR (GP posterior mean) baseline. The proposed regularization which stems from theorem 3.1 and 3.2 achieves lower test RMSE with fewer parameters, with the notable exception of the concrete datasets where it leads to overfitting.

served that for the same number of model parameters, TT converges more rapidly than CPD as the dimensionality of the inputs grows. Both approaches however need exponentially more parameters to converge at the same rate for increasing dimensionality of the inputs. Note that for $Q = P = 4$ CPD, contrary to what stated in section 3.1 still converges faster due to the approximation made when considering $P_{\text{TT}} = QMR^2$. Histograms of the empirical CDF for one datapoint are shown in figure 1. This behavior stems from the fact that for a fixed combination of Q, M and P, TT captures an exponential R^{Q-1} range of model interactions in contrast to the R linear interactions exhibited by CPD.

4.2 GP Behavior at Prediction

To investigate whether CPD and TT-constrained kernel machines trained with the priors of theorem 3.1 and 3.2 indeed exhibit less GP behavior compared to the standard CPD and TT-constrained prior we tackle four small *University of California, Irvine* (UCI) regression problems [Dua and Graff, 2017]. We consider 70 % of the data for training and the remaining for test and train a KRR model (equivalent to the GP posterior mean) on the training data and chose its kernel and regularization hyperparameters by 3-fold cross-validation. With the found KRR hyperparameters we then train two CPD-constrained kernel machines with *Alternating Least-Squares* (ALS) for an increasing number of ranks and thus of parameters, one such model with the standard regularization (equation (17)) and one with the regularization that follows from theorem 3.1 (equation (18)). We repeat the same procedure for TT-constrained kernel machines. We report the RMSE on test data in figure 3 and on train in figure 4 in the appendix. In figure 3 one can observe that on all datasets the predictions on unseen data of both CPD and TT models trained with the standard regularization (green and red curves respectively) converge to the KRR baseline (dotted line) for $P \ll N$. On the contrary, the CPD and TT models trained with the regularization term of equation (18) (blue and orange respectively) with the exception of the concrete dataset fare better in terms of test error, as they have been trained with a regularization that recover the KRR baseline in the limit. Plots of the training errors can be found in figure 4 in appendix C.1, where it can be seen that the regularization enforced by theorem 3.1 and 3.2 yields overall models that fit the data better and, with the exception of the concrete dataset, generalize better.

5 RELATED WORK

Our contribution is closely tied to the links between Bayesian neural networks and GPs, first established for single-layer single-output neural networks [Neal, 1996a,b] having sigmoidal [Williams, 1996], Gaussian [Williams, 1997] and rectified linear unit [Cho and Saul, 2009] as activation function. This idea was extended to DNNs by Lee et al. [2018] and Matthews et al. [2018] for the most common activation functions. Further extensions have been proposed to CNNs where the number of channels tends to infinity [Novak et al., 2018; Garriga-Alonso et al., 2018], to RNNs [Sun et al., 2022] and to DNNs having low-rank constraints on the weight matrices [Nait-Saada et al., 2023]. In particular theorem 3.1 resembles the results of Neal [1996a,b]; Williams [1997] which relate infinite-width single layer neural networks to GPs. The CPD rank corresponds exactly to the width of the neural network. The crucial difference lies however in the Kronecker product structure, which is not present in neural networks and introduces a nonlinearity of different kind than the activation function. TTs on the other hand resemble DNNs as they map the output of each core to the next one. However, in contrast to DNNs, the inputs are processed over the depth of the network. For a more in depth discussion we refer the reader to [Cohen et al., 2016]. Likewise theorem 3.2 is the TN counterpart to the works of Lee et al. [2018]; Matthews et al. [2018] which relate finite depth neural networks to GPs. The priors we propose are also used in practice as a sensible initial guess for gradient-based optimization of TN-constrained models [Barratt et al., 2021]. The results related to TT-constrained kernel machines in theorem 3.2 were also derived by Guo and Draper [2021a,b] from a quantum mechanical perspective, though a theoretical and experimental comparison with CPD-constrained kernel machines was not provided.

6 CONCLUSION

In this paper we proved that CPD and TT-constrained kernel machines are product kernel GPs in the limit of large TN ranks when placing suitable priors on their parameters. We characterized the target GP and showed that compared to CPD, TT-based models converge faster to the GP when dealing with higher-dimensional inputs. The proposed priors can be used in case of finite rank to train more flexible models that better fit the data compared to the standard approach which seeks instead to approximate the posterior with the addition of a TN constraint. One important limitation is that the ensuing models are more susceptible to overfitting and have thus to be tuned with more care. We empirically demonstrated these observations by means of numerical

experiments.

Acknowledgments

We would like to thank the anonymous reviewers for their numerous suggestions and improvements which have greatly improved the quality of this paper. Frederiek Wesel, and thereby this work, is supported by the Delft University of Technology AI Labs program. This publication is part of the project Sustainable learning for Artificial Intelligence from noisy large- scale data (with project number VI.Vidi.213.017) which is financed by the Dutch Research Council (NWO). The authors declare no competing interests.

References

F. Barratt, J. Dborin, and L. Wright. Improvements to Gradient Descent Methods for Quantum Tensor Network Machine Learning. In *Second Workshop on Quantum Tensor Networks in Machine Learning*, May 2021.

K. Batselier, Z. Chen, and N. Wong. Tensor Network alternating linear scheme for MIMO Volterra system identification. *Automatica*, 84:26–35, Oct. 2017.

Z. Chen, K. Batselier, J. A. K. Suykens, and N. Wong. Parallelized Tensor Train Learning of Polynomial Classifiers. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10):4621–4632, Oct. 2018.

Y. Cho and L. Saul. Kernel Methods for Deep Learning. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009.

A. Cichocki. Era of Big Data Processing: A New Approach via Tensor Networks and Tensor Decompositions. *arXiv:1403.2048 [cs]*, Aug. 2014.

A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, and D. P. Mandic. Tensor Networks for Dimensionality Reduction and Large-Scale Optimization: Part 1 Low-Rank Tensor Decompositions. *Foundations and Trends® in Machine Learning*, 9(4-5):249–429, 2016.

A. Cichocki, A.-H. Phan, Q. Zhao, N. Lee, I. V. Oseledets, M. Sugiyama, and D. Mandic. Tensor Networks for Dimensionality Reduction and Large-Scale Optimizations. Part 2 Applications and Future Perspectives. *Foundations and Trends® in Machine Learning*, 9(6):249–429, 2017.

N. Cohen, O. Sharir, and A. Shashua. On the Expressive Power of Deep Learning: A Tensor Analysis. In *Conference on Learning Theory*, pages 698–728. PMLR, June 2016.

L. Csató and M. Opper. Sparse On-Line Gaussian Processes. *Neural Computation*, 14(3):641–668, Mar. 2002.

R. B. D'Agostino and M. A. Stephens. *Goodness-of-Fit Techniques*. CRC Press, Jan. 1986.

D. Dua and C. Graff. UCI Machine Learning Repository, 2017.

D. K. Duvenaud, H. Nickisch, and C. Rasmussen. Additive Gaussian Processes. *Advances in Neural Information Processing Systems*, 24:226–234, 2011.

A. Garriga-Alonso, C. E. Rasmussen, and L. Aitchison. Deep Convolutional Networks as shallow Gaussian Processes. In *International Conference on Learning Representations*, Sept. 2018.

E. Guo and D. Draper. Infinitely Wide Tensor Networks as Gaussian Process, Jan. 2021a.

E. Guo and D. Draper. Neural Tangent Kernel of Matrix Product States: Convergence and Applications, Nov. 2021b.

T. Hastie, J. Friedman, and R. Tibshirani. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, NY, 2001.

J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for Big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, pages 282–290. AUAI Press, Aug. 2013.

J. Hensman, A. Matthews, and Z. Ghahramani. Scalable Variational Gaussian Process Classification. In *Artificial Intelligence and Statistics*, pages 351–360. PMLR, Feb. 2015.

J. Hensman, N. Durrande, and A. Solin. Variational Fourier features for Gaussian processes. *The Journal of Machine Learning Research*, 18(1):5537–5588, Jan. 2017.

F. L. Hitchcock. The Expression of a Tensor or a Polyadic as a Sum of Products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927.

C. Hua, G. Rabusseau, and J. Tang. High-Order Pooling for Graph Neural Networks with Tensor Decomposition. *Advances in Neural Information Processing Systems*, 35:6021–6033, Dec. 2022.

P. Izmailov, A. Novikov, and D. Kropotov. Scalable Gaussian Processes with Billions of Inducing Inputs via Tensor Train Decomposition. In *International Conference on Artificial Intelligence and Statistics*, pages 726–735. PMLR, Mar. 2018.

M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up Convolutional Neural Networks with Low Rank Expansions. *Proceedings of the British Machine Vision Conference 2014*, 2014.

N. Kargas and N. D. Sidiropoulos. Supervised Learning and Canonical Decomposition of Multivariate Functions. *IEEE Transactions on Signal Processing*, pages 1–1, 2021.

M. Lázaro-Gredilla, J. Quiñnero-Candela, C. E. Rasmussen, and b. R. Figueiras-Vidal. Sparse Spectrum Gaussian Process Regression. *Journal of Machine Learning Research*, 11(63):1865–1881, 2010.

V. Lebedev, Y. Ganin, M. Rakhuba, I. V. Oseledets, and V. S. Lempitsky. Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition. In *International Conference on Learning Representations*, Jan. 2015.

J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep Neural Networks as Gaussian Processes. In *International Conference on Learning Representations*, Feb. 2018.

X. Lu, A. Boukouvalas, and J. Hensman. Additive Gaussian Processes Revisited. In *Proceedings of the 39th International Conference on Machine Learning*, pages 14358–14383. PMLR, June 2022.

X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song. A Tensorized Transformer for Language Modeling. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani. Gaussian Process Behaviour in Wide Deep Neural Networks, Aug. 2018.

T. Nait-Saada, A. Naderi, and J. Tanner. Beyond IID weights: Sparse and low-rank deep Neural Networks are also Gaussian Processes. In *The Twelfth International Conference on Learning Representations*, Oct. 2023.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer Science & Business Media, Jan. 1996a.

R. M. Neal. Priors for Infinite Networks. In R. M. Neal, editor, *Bayesian Learning for Neural Networks*, Lecture Notes in Statistics, pages 29–53. Springer, New York, NY, 1996b.

R. Novak, L. Xiao, Y. Bahri, J. Lee, G. Yang, J. Hron, D. A. Abolafia, J. Pennington, and J. Sohl-dickstein. Bayesian Deep Convolutional Networks with Many Channels are Gaussian Processes. In *International Conference on Learning Representations*, Sept. 2018.

A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. Tensorizing neural networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

A. Novikov, I. Oseledets, and M. Trofimov. Exponential machines. *Bulletin of the Polish Academy of Sciences: Technical Sciences; 2018; 66; No 6 (Special Section on Deep Learning: Theory and Practice); 789-797*, 2018.

I. V. Oseledets. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, Jan. 2011.

J. Quiñonero-Candela and C. E. Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *Journal of Machine Learning Research*, 6(65):1939–1959, 2005.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, pages 1177–1184. Curran Associates Inc., Dec. 2007.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 2006.

J. A. Reyes and E. M. Stoudenmire. Multi-scale tensor network architecture for machine learning. *Machine Learning: Science and Technology*, 2(3):035036, July 2021.

M. W. Seeger, C. K. I. Williams, and N. D. Lawrence. Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In *International Workshop on Artificial Intelligence and Statistics*, pages 254–261. PMLR, Jan. 2003.

E. Snelson and Z. Ghahramani. Sparse Gaussian Processes using Pseudo-inputs. In *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2006.

A. Solin and S. Särkkä. Hilbert space methods for reduced-rank Gaussian process regression. *Statistics and Computing*, 30(2):419–446, Mar. 2020.

E. M. Stoudenmire and D. J. Schwab. Supervised learning with tensor networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4806–4814. Curran Associates Inc., Dec. 2016.

X. Sun, S. Kim, and J.-I. Choi. Recurrent neural network-induced Gaussian process. *Neurocomputing*, 509:75–84, Oct. 2022.

M. Titsias. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, pages 567–574. PMLR, Apr. 2009.

S. Wahls, V. Koivunen, H. V. Poor, and M. Verhaegen. Learning multidimensional Fourier series with tensor trains. In *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 394–398, Dec. 2014.

F. Wesel and K. Batselier. Large-Scale Learning with Fourier Features and Tensor Decompositions. In

Advances in Neural Information Processing Systems, volume 34, pages 17543–17554. Curran Associates, Inc., 2021.

F. Wesel and K. Batselier. Tensor-based Kernel Machines with Structured Inducing Points for Large and High-Dimensional Data. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 8308–8320. PMLR, Apr. 2023.

C. Williams. Computing with Infinite Networks. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996.

C. Williams. Computing with Infinite Networks. *Advances in Neural Information Processing Systems 9*, pages 295–301, 1997.

A. Wilson and H. Nickisch. Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP). In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1775–1784. PMLR, June 2015.

M. Yadav, D. Sheldon, and C. Musco. Faster Kernel Interpolation for Gaussian Processes. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 2971–2979. PMLR, Mar. 2021.

J. Ye, L. Wang, G. Li, D. Chen, S. Zhe, X. Chu, and Z. Xu. Learning Compact Recurrent Neural Networks With Block-Term Tensor Decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9378–9387, 2018.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] we define the setting (model) in the introduction and in the main section the theorems clearly state the assumptions and consequences.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Not Applicable] We do not propose a new algorithm but a theoretical connection between two approaches (TNKM and GPs) in terms of the CLT that was not discussed before.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes], the link is included in the paper.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes] all assumptions are clearly stated in the theorems.
   (b) Complete proofs of all theoretical results. [Yes] in both main text and appendix.
   (c) Clear explanations of any assumptions. [Yes] the assumptions are encapsulated in the theorems.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [No]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A Introduction

A.1 Notation

Throughout this paper we denote scalars in both capital and non-capital italics w, W, vectors in non-capital bold $\boldsymbol{w}$, matrices in capital bold $\boldsymbol{W}$ and tensors in capital italic bold font $\boldsymbol{\mathcal{W}}$. The m-th entry of a vector $\boldsymbol{w} \in \mathbb{R}^M$ is indicated as w_m and the $m_1 m_2 \dots m_Q$-th entry of a Q-dimensional tensor $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q}$ as $w_{m_1, m_2, \dots, m_Q}$. We employ the column notation to indicate a set of elements of tensor given a set of indices, e.g. $\boldsymbol{W}_{m_1,:,m_2}$ and $\boldsymbol{W}_{m_1,1:3,m_2}$ represent respectively all elements and the first three elements along the second dimension of tensor $\boldsymbol{\mathcal{W}}$ with fixed indices m_1 and m_2. The Kronecker product is denoted by $\otimes$ and the Hadamard (elementwise) by $\odot$. We employ one-based indexing for all tensors. The Frobenius inner product between two Q-dimensional tensors $\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{W}} \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q}$ is

$$\langle \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{W}} \rangle_{\mathrm{F}} \coloneqq \sum_{m_1=1}^{M_1} \sum_{m_2=1}^{M_2} \cdots \sum_{m_Q=1}^{M_Q} v_{m_1, m_2, \dots, m_Q} w_{m_1, m_2, \dots, m_Q},$$

and the Frobenius norm of $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q}$ is denoted and defined as

$$||\boldsymbol{\mathcal{W}}||_{\mathrm{F}}^2 \coloneqq \langle \boldsymbol{\mathcal{W}}, \boldsymbol{\mathcal{W}} \rangle_{\mathrm{F}}.$$

We define the vectorization operator as $\texttt{vec}\,(\cdot) : \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_Q} \to \mathbb{R}^{M_1 M_2 \cdots M_Q}$ such that

$$\texttt{vec}\,(\boldsymbol{\mathcal{W}})_m = w_{m_1, m_2, \dots, m_Q},$$

with $m = m_1 + \sum_{q=2}^{Q} (m_q - 1) \prod_{k=1}^{q-1} M_k$. Likewise, its inverse, the tensorization operator $\texttt{ten}\,(\cdot) : \mathbb{R}^{M_1 M_2 \cdots M_Q} \to \mathbb{C}^{M_1 \times M_2 \times \dots M_Q}$ is defined such that

$$\texttt{ten}\,(\boldsymbol{w})_{m_1, m_2, \cdots, m_Q} = w_m.$$

B Tn-Constrained Kernel Machines as GPs

B.1 GP of CPD-Constrained Kernel Machine

Theorem 3.1 (CPD-constrained kernel machine as GP)**.** *Consider the CPD-constrained kernel machine*

$$f_{\texttt{CPD}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\,(\boldsymbol{\varphi}(\boldsymbol{x}))), \texttt{CPD}(\texttt{ten}\,(\boldsymbol{w})) \rangle_{\mathrm{F}}.$$

If each of the R columns $\boldsymbol{w}^{(q)}{}_{:,r} \in \mathbb{R}^{M_q}$ of each CPD core is an i.i.d. random variable such that

$$\mathbb{E}\left[\boldsymbol{w}^{(q)}_{:,r}\right] = \mathbf{0},$$
$$\mathbb{E}\left[\boldsymbol{w}^{(q)}_{:,r} \boldsymbol{w}^{(q)\mathrm{T}}_{:,r}\right] = R^{-\frac{1}{Q}} \boldsymbol{\Lambda}^{(q)},$$

then $f_{\texttt{CPD}}(\boldsymbol{x})$ converges in distribution as $R \to \infty$ to the GP

$$f_{\texttt{CPD}}(\boldsymbol{x}) \sim \mathcal{GP}\left(0, \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\cdot)\right).$$

Proof. Consider the R intermediate functions h_r of equation (13) which constitute the CPD-constrained model of equation (11). Due to the i.i.d. assumption on $\boldsymbol{w}^{(q)}{}_{:,r}$ each addend is the same function of i.i.d. random variables and thus is itself i.i.d.. The mean of each addend is

$$\mathbb{E}\left[h_r(\boldsymbol{x})\right] = \mathbb{E}\left[\prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{w}^{(q)}{}_{:,r}\right] = 0, \tag{19}$$

due to the i.i.d assumption and the linearity of expectation. Its covariance is

$$\mathbb{E}\left[h_r(\boldsymbol{x})h_r(\boldsymbol{x'})\right] \tag{20a}$$

$$= \mathbb{E}\left[\prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{w}^{(q)}{}_{:,r} \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'})^{\mathrm{T}} \boldsymbol{w}^{(q)}{}_{:,r}\right] \tag{20b}$$

$$= \mathbb{E}\left[\prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{w}^{(q)}{}_{:,r} \boldsymbol{w}^{(q)}{}_{:,r}{}^{\mathrm{T}} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'})\right] \tag{20c}$$

$$= \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \mathbb{E}\left[\boldsymbol{w}^{(q)}{}_{:,r} \boldsymbol{w}^{(q)}{}_{:,r}{}^{\mathrm{T}}\right] \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'}) \tag{20d}$$

$$= \frac{1}{R} \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'}).$$

Here the step from equation (20b) to equation (20c) exploits the fact that the transpose of a scalar is equal to itself, the step from equation (20c) to equation (20d) is due to the linearity of expectation. As the variances of each intermediate function h_r are appropriately scaled, by the CLT the partial sum $f_{\texttt{CPD}}(\boldsymbol{x})$ converges in distribution to a multivariate normal distribution, which is fully specified by its first two moments

$$\mathbb{E}\left[f_{\texttt{CPD}}(\boldsymbol{x})\right] = 0,$$
$$\mathbb{E}\left[f_{\texttt{CPD}}(\boldsymbol{x})f_{\texttt{CPD}}(\boldsymbol{x'})\right] = \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'}).$$

Since any finite collection of $\{f_{\texttt{CPD}}(\boldsymbol{x}), \ldots, f_{\texttt{CPD}}(\boldsymbol{x'})\}$ will have a joint multivariate normal distribution with the aforementioned first two moments, we conclude that $f_{\texttt{CPD}}(\boldsymbol{x})$ is the Gaussian process

$$f_{\texttt{CPD}}(\boldsymbol{x}) \sim \mathcal{GP}\left(0, \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\cdot)\right).$$

□

B.2 GP of TT-Constrained Kernel Machine in the Sequential Limit of the TT Ranks

Theorem 3.2 (TT-constrained kernel machine as GP)**.** *Consider the TT-constrained kernel machine*

$$f_{\texttt{TT}}(\boldsymbol{x}) \coloneqq \langle \texttt{R}_1(\texttt{ten}\left(\boldsymbol{\varphi}(\boldsymbol{x})\right)), \texttt{TT}(\texttt{ten}\left(\boldsymbol{w}\right))\rangle_{\mathrm{F}}$$

If each of the $R_{q-1}R_q$ fibers $\boldsymbol{W}^{(q)}{}_{r_{q-1},:,r_q} \in \mathbb{R}^{M_q}$ of each TT core is an i.i.d. random variable such that

$$\mathbb{E}\left[\boldsymbol{W}^{(q)}_{r_{q-1},:,r_q}\right] = \boldsymbol{0},$$
$$\mathbb{E}\left[\boldsymbol{W}^{(q)}_{r_{q-1},:,r_q} \boldsymbol{W}^{(q)}_{r_{q-1},:,r_q}{}^{\mathrm{T}}\right] = \frac{1}{\sqrt{R_{q-1}R_q}} \boldsymbol{\Lambda}^{(q)},$$

then $f_{\texttt{TT}}(\boldsymbol{x})$ converges in distribution as sequentially $R_1 \to \infty$, $R_2 \to \infty$, …, $R_{Q-1} \to \infty$ to the Gaussian process

$$f_{\texttt{TT}}(\boldsymbol{x}) \sim \mathcal{GP}\left(0, \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\cdot)\right).$$

Proof. Define the vector of intermediate function $\boldsymbol{h}^{(q+1)} \in \mathbb{R}^{R_{q+1}}$ recursively as

$$h^{(q+1)}_{r_{q+1}} \coloneqq \sum_{r_q=1}^{R_q} z^{(q+1)}_{r_q,r_{q+1}}(x_{q+1}) h^{(q)}_{r_q},$$

with $h^{(0)} \coloneqq 1$. Note that the first two moments of intermediate variable $z^{(q+1)}_{r_q,r_{q+1}}(x_{q+1})$ are

$$\begin{aligned}
&\mathbb{E}\left[z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x})\right] = 0,\\
&\mathbb{E}\left[z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x}) z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x'})\right]\\
&= \frac{1}{\sqrt{R_q R_{q+1}}} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'}).
\end{aligned}$$

We proceed by induction. For the induction step suppose that $h^{(q)}_{r_q}$ is a GP, identical and independent for every r_q such that

$$h^{(q)}_{r_q} \sim \mathcal{GP}\left(0, \frac{1}{\sqrt{R_q}} \prod_{p=1}^{q} \boldsymbol{\varphi}^{(p)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(\cdot)\right).$$

The scalar $h^{(q+1)}_{r_{q+1}}$ is the sum of R_q i.i.d. terms having mean

$$\mathbb{E}\left[h^{(q+1)}_{r_{q+1}}\right] = \mathbb{E}\left[z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x}) h^{(q)}_{r_q}\right] = 0,$$

and covariance

$$\begin{aligned}
&\mathbb{E}\left[h^{(q+1)}_{r_{q+1}} h^{(q+1)}_{r_{q+1}}\right]\\
=&\mathbb{E}\left[z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x}) h^{(q)}_{r_q} z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x'}) h^{(q)}_{r_q}\right]\\
=&\mathbb{E}\left[z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x}) z^{(q+1)}_{r_q,r_{q+1}}(\boldsymbol{x'})\right] \mathbb{E}\left[h^{(q)}_{r_q} h^{(q)}_{r_q}\right]\\
=&\frac{1}{\sqrt{R_{q+1}}} \prod_{p=1}^{q+1} \boldsymbol{\varphi}^{(p)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(\boldsymbol{x'}).
\end{aligned}$$

Since the assumptions of the CLT are satisfied the partial sum $h^{(q+1)}_{r_{q+1}}$ converges in distribution to the normal distribution, fully specified by the above mentioned first two moments. Since any finite collection of $\{h^{(q+1)}_{r_{q+1}}(\boldsymbol{x}_{1:q+1}), \ldots, h^{(q+1)}_{r_{q+1}}(\boldsymbol{x'}_{1:q+1})\}$ will have a joint multivariate normal distribution with the aforementioned first two moments, we conclude that $h^{(q+1)}_{r_{q+1}}(\boldsymbol{x}_{1:q+1})$ is the GP

$$h^{(q+1)}_{r_{q+1}} \sim \mathcal{GP}\left(0, \frac{1}{\sqrt{R_{q+1}}} \prod_{p=1}^{q+1} \boldsymbol{\varphi}^{(p)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(\cdot)\right).$$

For the base case, consider the R_1 outputs of the first hidden function $h^{(1)}_{r_1}$. They are i.i.d. with mean

$$\mathbb{E}\left[h^{(1)}_{r_1}(\boldsymbol{x})\right] = 0.$$

and covariance

$$\mathbb{E}\left[h^{(1)}_{r_1}(\boldsymbol{x}) h^{(1)}_{r_1}(\boldsymbol{x'})\right] = \frac{1}{\sqrt{R_1}} \boldsymbol{\varphi}^{(1)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(1)} \boldsymbol{\varphi}^{(1)}(\boldsymbol{x'}).$$

We now consider the R_2 outputs of the second hidden function $h^{(2)}_{r_2}$

$$h^{(2)}_{r_2} = \sum_{r_1=1}^{R_1} z^{(2)}_{r_1,r_2}(\boldsymbol{x}) h^{(1)}_{r_1},$$

which are i.i.d. as they are the same function of the R_1 i.i.d. outputs of $h^{(1)}_{r_1}(\boldsymbol{x})$. More specifically, their mean and covariance are

$$\begin{aligned}
&\mathbb{E}\left[h^{(2)}_{r_2}\right] = 0,\\
&\mathbb{E}\left[h^{(2)}_{r_2}(\boldsymbol{x}) h^{(2)}_{r_2}(\boldsymbol{x'})\right]\\
=&\frac{1}{\sqrt{R_2}} \prod_{q=1}^{2} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(d)}(\boldsymbol{x'}).
\end{aligned}$$

Once more by the CLT, the partial sum $h_{r_2}^{(2)}$ converges in distribution to the normal distribution with the above first two moments. Since any finite collection of $\{h_{r_2}^{(2)}(\boldsymbol{x}), \ldots, h_{r_2}^{(2)}(\boldsymbol{x'}\}$ will have a joint multivariate normal distribution with the aforementioned first two moments, we conclude that $h_2^{(2)}(\boldsymbol{x})$ is the GP

$$h_{r_2}^{(2)} \sim \mathcal{GP}\left(0, \frac{1}{\sqrt{R_2}} \prod_{q=1}^{2} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\cdot)\right),$$

which is our base case. Hence by induction $f_{\mathrm{TT}}(\boldsymbol{x}) = h^{(Q)}$ converges in distribution as $R_1 \to \infty$, $R_2 \to \infty$, …, $R_{Q-1} \to \infty$ to the GP

$$f_{\mathrm{TT}}(\boldsymbol{x}) \sim \mathcal{GP}\left(0, \prod_{q=1}^{Q} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\cdot)\right).$$

□

B.3 GP of TT-Constrained Kernel Machine in the Simultaneous Limit of the TT Ranks

In theorem 3.2 we prove by induction that the TT-constrained kernel machine converges to a GP by taking successive limits of the TT ranks. This result is analogous to the work of Lee et al. [2018], who prove that for the DNNs, taking sequentially the limit of each layer. A more practically useful result consists in the convergence in the *simultaneous* limit of TT ranks.

In deep learning Matthews et al. [2018, theorem 4] prove convergence in the context of DNNs over the widths of all layers simultaneously. Said theorem has been employed to prove GP convergence in the context of convolutional neural networks [Garriga-Alonso et al., 2018] and in the context of DNNs where each weight matrix is of low rank [Nait-Saada et al., 2023].

Seeing the similarity between TT-constrained kernel machines (equation (14)) and DNNs and the technicality of the proof, similarly to [Garriga-Alonso et al., 2018; Nait-Saada et al., 2023] we draw a one-to-one map between the TT-constrained kernel machines and the DNNs considered in Matthews et al. [2018, theorem 4]. Convergence in the simultaneous limit is then guaranteed by Matthews et al. [2018, theorem 4].

We begin by restating the definitions of linear envelope property, DNNs, linear envelope property and normal recursion as found in Matthews et al. [2018]. To make the comparison easier for the reader, we change the indexing notation to match the one in this paper.

Definition B.1 (Linear envelope property for nonlinearities [Matthews et al., 2018])**.** A nonlinearity $t : \mathbb{R} \to \mathbb{R}$ is said to obey the linear envelope property if there exist $c, l \geq 0$ such that the following inequality holds

$$|t(u)| < c + l|u| \ \forall u \in \mathbb{R}. \tag{25}$$

Definition B.2 (Fully connected DNN [Matthews et al., 2018])**.** A fully connected deep neural with one-dimensional output and inputs $\boldsymbol{x} \in \mathbb{R}^{R_0}$ is defined recursively such that the initial step is

$$h_{r_1}^{(1)}(\boldsymbol{x}) = \sum_{r_0=1}^{R_0} z_{r_1,r_0}^{(1)} x_{r_0} + b_{r_1}^{(1)}, \tag{26}$$

the activation step by nonlinear activation function t is given by

$$g_{r_q}^{(q)} = t(f_{r_q}^{(q)}), \tag{27}$$

and the subsequent layers are defined by the recursion

$$h_{r_{q+1}}^{(q+1)} = \sum_{r_q=1}^{R_q} z_{r_{q+1},r_d}^{(q+1)} g_{r_q}^{(q)} + b_{r_{q+1}}^{q+1}, \tag{28}$$

so that $h^{(Q)}$ is the output of the network. In the above, $\boldsymbol{Z}^{(q)} \in \mathbb{R}^{R_{q-1} \times R_q}$ and $\boldsymbol{b}^{(q)} \in \mathbb{R}^{R_q}$ are respectively the weights and biases of the q-th layer.

Definition B.3 (Width function Matthews et al. [2018])**.** For a given fixed input $n \in \mathbb{N}$, a width function $v^{(q)} : \mathbb{N} \to \mathbb{N}$ at depth q specifies the number of hidden units R_q at depth q.

Lemma B.4 (Normal recursion [Matthews et al., 2018])**.** *Consider* $z^{(q)}_{r_{q-1},r_q} \sim \mathcal{N}(0, C^{(q)}_w)$ *and* $b^{(q)}_{r_q} \sim \mathcal{N}(0, C^{(q)}_b)$. *If the activations of the q-th layer are normally distributed with moments*

$$\mathbb{E}\left[h^{(q)}_{r_q}\right] = 0 \tag{29}$$

$$\mathbb{E}\left[h^{(q)}_{r_q} h^{(q)}_{r_q}\right] = K(x, x'), \tag{30}$$

then under recursion equations (27) *and* (28)*, as* $R_{q-1} \to \infty$*, the activations of the next layer converge in distribution to a normal distribution with moments*

$$\mathbb{E}\left[h^{(q+1)}_{r_{q+1}}\right] = 0 \tag{31}$$

$$\mathbb{E}\left[h^{(q+1)}_{r_{q+1}} h^{(q+1)}_{r_{q+1}}\right] = C^{(q+1)}_w \, \mathbb{E}_{(\epsilon_1,\epsilon_2)\sim\mathcal{N}(0,K)}\left[t(\epsilon_1)t(\epsilon_2)\right] + C^{(q+1)}_b. \tag{32}$$

We can now state the major result in Matthews et al. [2018].

Theorem B.5 (GP in the simultaneous limit of fully connected DNNs [Matthews et al., 2018])**.** *Consider a random* DNN *of the form of definition B.2 obeying the linear envelope condition of definition B.1. Then for all sets of strictly increasing width functions* $v^{(q)}$ *and for any countable input set* $\{\boldsymbol{x}, \ldots, \boldsymbol{x'}\}$*, the distribution of the output of the network converges in distribution to a* GP *as* $n \to \infty$*. The* GP *has mean and covariance functions given by the recursion in lemma B.4.*

Corollary B.6 (GP in the simultaneous limit of TT-constrained kernel machines)**.** *Consider a random* TT*-constrained kernel machine of the form of definition 2.6 obeying the linear envelope condition of definition B.1. Then for all sets of strictly increasing width functions* $v^{(q)}$ *and for any countable input set* $\{\boldsymbol{x}, \ldots, \boldsymbol{x'}\}$*, the distribution of the output of the network converges in distribution to a* GP *as* $P \to \infty$*. The* GP *has mean and covariance functions given by the recursion in lemma B.4 and stated in theorem 3.2.*

Proof. When examining definition B.2 and comparing it with definition 2.6 it becomes clear that both models are similar. In the special case of involving linear activation function and zero biases, the models are structurally identical if one considers unit inputs $x = 1$ in equation (26). The normal recursion in lemma B.4 is satisfied by TT-constrained kernel machines, as we have that

$$\begin{aligned}
t(u) &:= u \;\forall u \in \mathbb{R}, \\
C^{(q+1)}_b &:= 0, \\
C^{(q+1)} &:= \frac{1}{\sqrt{R_q R_{q+1}}} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(q)} \boldsymbol{\varphi}^{(q)}(\boldsymbol{x'}), \\
K &:= \frac{1}{\sqrt{R_q}} \prod_{p=1}^{q} \boldsymbol{\varphi}^{(p)}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(\boldsymbol{x'}) \\
\mathbb{E}_{(\epsilon_1,\epsilon_2)\sim\mathcal{N}(0,K)}\left[t(\epsilon_1)t(\epsilon_2)\right] &:= K.
\end{aligned}$$

Hence by theorem B.5, for all sets of strictly increasing width functions $v^{(q)}$ and for any countable input set $\{\boldsymbol{x}, \ldots, \boldsymbol{x'}\}$, the distribution of the output of the network converges in distribution to a GP, fully specified by the output of the normal recursion in lemma B.4, which equals the GP in theorem 3.2. □

C Numerical Experiments

C.1 GP Behavior at Prediction

We provide the training RMSE related to section 4.2 in figure 4, where it can be seen that the new priors yield model that provide a better fit on all datasets.

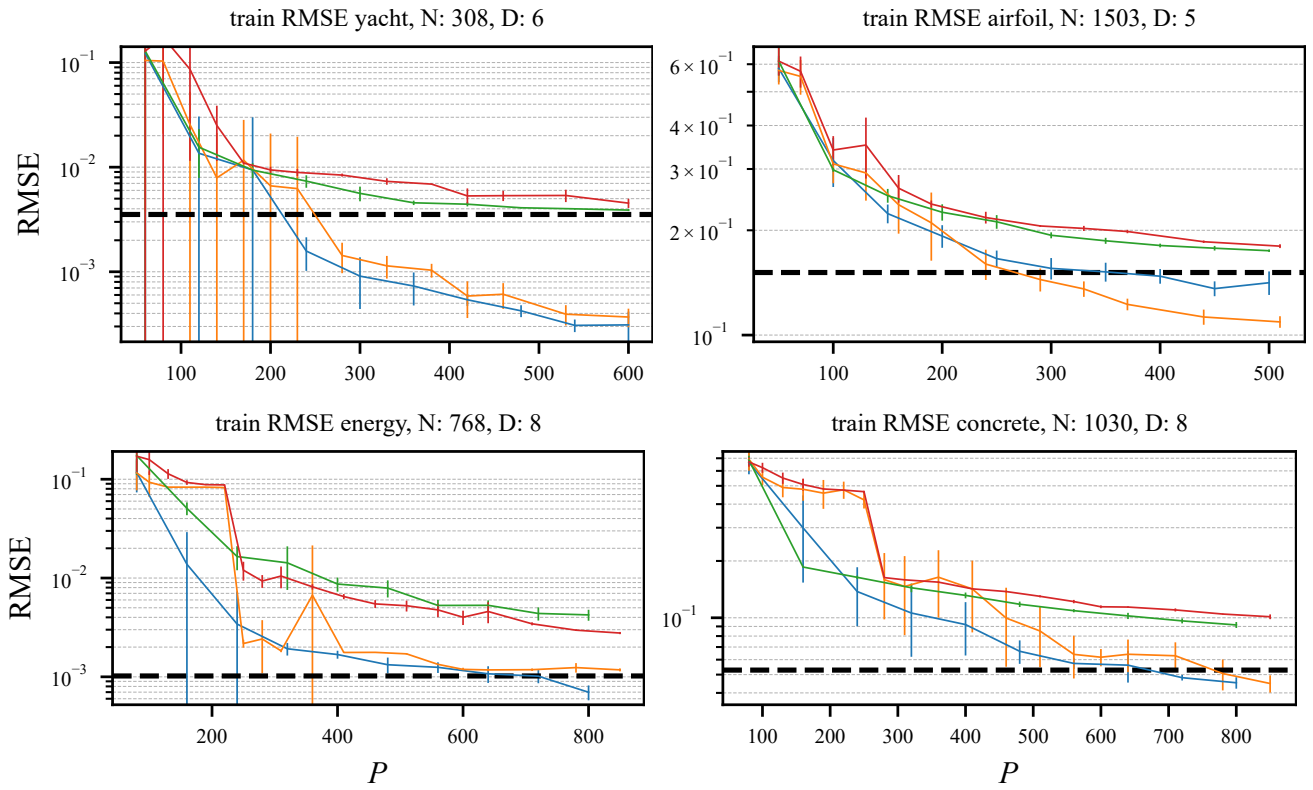


Figure 4: Mean and standard deviation of the training RMSE of CPD and TT models for regularization equations (17) and (18) (green and red curves respectively) as a function of model parameters P as well as their target KRR (dotted line). In the plots, the probabilistic regularization of equation (18) and its TT counterpart are denoted by a blue and orange line respectively. The dotted line corresponds to the KRR (GP posterior mean) baseline. The proposed regularization which stems from theorem 3.1 and 3.2 achieves lower test RMSE with fewer parameters, with the notable exception of the concrete datasets where it leads to overfitting.