DECOUPLED-VALUE ATTENTION FOR PRIOR-DATA FITTED NETWORKS: GP INFERENCE FOR PHYSICAL EQUATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Prior-data fitted networks (PFNs) are a promising alternative to time-consuming Gaussian process (GP) inference for creating fast surrogates of physical systems. PFN reduces the computational burden of GP-training by replacing Bayesian inference in GP with a single forward pass of a learned prediction model. However, with standard Transformer attention, PFNs show limited effectiveness on high-dimensional regression tasks. We introduce Decoupled-Value Attention (DVA)–motivated by the GP property that the function space is fully characterized by the kernel over inputs and the predictive mean is a weighted sum of training targets. DVA computes similarities from inputs only and propagates labels solely through values. Thus, the proposed DVA mirrors the GP update while remaining kernel-free. We demonstrate that the crucial factor for scaling PFNs is the attention rule rather than the architecture itself. Specifically, our results demonstrate that (a) localized attention consistently reduces out-of-sample validation loss in PFNs across different dimensional settings, with validation loss reduced by more than 50% in five- and ten-dimensional cases, and (b) the role of attention is more decisive than the choice of backbone architecture, showing that CNN-based PFNs can perform at par with their Transformer-based counterparts. The proposed PFNs provide 64-dimensional power flow equation approximations with a mean absolute error of the order of 10^{-3}, while being over $80\times$ faster than exact GP inference.

1 INTRODUCTION

Bayesian inference provides a powerful framework for reasoning under uncertainty, with methods like Gaussian processes (GPs) offering well-calibrated predictions and principled uncertainty estimates (Williams & Rasmussen, 2006). However, the practical application of these methods is often hindered by the heavy computational burden of learning kernel hyperparameters. For example, exact GP inference scales cubically with the number of data points, making its deployment infeasible for large datasets or problems requiring repeated training. Consider a physical system where a surrogate GP is chosen due to its uncertainty estimates and differentiable closed-form expressions. However, the underlying input dataset and configuration changes frequently, and the surrogate is supposed to work for these new, previously unseen variations. For example, changes in underlying physical networks for power grids Tan et al. (2025). In such conditions, GP needs to be trained repeatedly, incurring significant computing cost, each time the dataset changes.

To address this, Prior-Data Fitted Networks (PFNs) have emerged as a method (Müller et al., 2022) that uses large-scale pre-training to approximate the Bayesian posterior predictive in a single forward pass. Note that unlike sparse GP approximations Daskalakis et al. (2022), PFNs eliminate kernel parameter training step. Although Low-rank approximations reduce GP cost to $\mathcal{O}(nm^2)$, where m is a user-defined parameter, PFNs need only a forward pass at deployment. This advantage grows when multiple GPs must be learned, as training K GPs scales to $\mathcal{O}(Knm^2)$, with each m requiring tuning. PFNs avoid these issues by directly predicting the posterior distribution in one step in a forward pass of the trained network. However, PFNs face scaling and bias issues in problems with high input dimensions due to their joint input–output embedding strategy Müller et al. (2022); Hollmann et al. (2025); Wang et al. (2025); Nagler (2023). Attention over concatenated $(\mathbf{x}, \mathbf{y})$ embeddings, as done in PFNs, degrade locality and similarity measures as input dimension grows

Further, they are almost exclusively built using Transformer architectures, which have high memory requirements. These challenges in existing PFFs motivate this work.

In this work, we propose **Decoupled-Value Attention (DVA)**, an input localized attention mechanism to scale PFNs with different architectures. We provide evidence that the attention mechanism is the primary driver of PFN performance, and it can be built using different architectures Convolution Neural Networks (CNNs) as well along with Transformers. The proposed DVA computes attention affinities (queries and keys) purely from the input space, while propagating information from the output space exclusively through the values. This aligns directly with the functional-space view of a GP, where the influence of training outputs y_i on a test prediction is weighted by the similarity of their corresponding inputs x_i Williams & Rasmussen (2006). This is a significant deviation from the standard attention mechanism applied in existing PFN works where affinities are calculated from a concatenated input-output vector Müller et al. (2022); Hollmann et al. (2022). This, combining inputs and outputs, increases the computational load, reducing PFNs ability to learn when the dimensions of input space grow. We note the observation made by Nagler (2023) that the convergence of PFNs is due to the attention mechanism, while bias[1] is a function of architecture choice. More importantly, it argues that a *post-hoc localization mechanism* is needed to reduce bias. We show that using the proposed localized attention, the reduction in PFN validation loss is consistent across architectures and exceeds the bias variation caused by the architectures themselves. Experimental studies show that DVA performs better than standard Vanilla Attention (VA) used in PFN literature, across dimensions and architectures. Our main contributions are:

- **A Localized Attention Mechanism for GP-PFNs:** We introduce DVA, which explicitly enforces input-only localization and reduces difference between predicted and true posterior distributions in PFN training by more than 50% for the inputs of 5D and 10D[2]. This design leads to substantially lower validation loss and improves predictive performance on high-dimensional regression tasks compared to standard PFN attention, without requiring additional data or compute resources.
- **Attention is More Important than Architecture:** We show that PFNs can also be constructed using CNN as backbone architecture, and with DVA, the choice of backbone architecture becomes secondary. This confirms that the attention mechanism is the primary driver of bias reduction. The proposed CNN-DVA based PFN achieves accuracy comparable to a Transformer-DVA based PFN across input dimensions upto 64D. Overall, changes in attention produce a more pronounced reduction in validation loss and predicted error than changes in the backbone architecture.
- **Scaling PFNs to High-Dimensional Learning Problems:** Standard PFNs with joint input–output attention fail to generalize beyond $\sim$10 input dimensions (10D), saturating at high validation loss. In contrast, DVA enables successful inference up to 64D and on power flow learning task in 64D, CNN+DVA achieves Mean Squared Error of order 10^{-5} even with 50% load uncertainty levels, and Mean Absolute Error on the order of 10^{-3} – at $\mathbf{80}\times$ the speed of exact GP inference.

Positioning: We want to highlight that our goal is not to claim a novel, general-purpose attention mechanism. Rather, DVA is a specialized design intended to create scalable and robust PFNs via localization and emulation of GP inference. We also note that there are many efficient attention mechanisms, including linearized kernels Katharopoulos et al. (2020), Nyström approximations Xiong et al. (2021), random feature expansions Choromanski et al. (2021), and cross-kernel attention Wang & Others (2025), which are kernel-based. These attentions are designed to incorporate GP and kernel advantages into Transformer-based language and vision models, along with scaling approximations like Peng et al. (2021); Bui et al. (2025). In contrast, the proposed DVA is designed to develop scalable PFNs Hollmann et al. (2022) that can mimic GP inference for physical equations in particular. Further, our sole focus is not on scaling PFNs with Transformer-like Wang et al. (2025), instead a) highlight that attention is the critical component in PFNs over architectures and b) bias reduction in PFNs can be achieved via attention without post-hoc localization Nagler (2023). More importantly, DVA is intentionally designed to remain *kernel-free* because forcing a single kernel type can lead to significant model mismatch for physics problems. For instance, the functions

[1]Here, we use the definition of bias as the difference between the parametrized PPD and the true PPD, which converges to zero as the number of samples increases Nagler (2023). Consequently, a decrease in NLL reflects a reduction in this bias.

[2]We use D to indicate dimension; for example, ND means N-dimensional.

governing AC power flow are best modeled by specialized kernels distinct from standard choices (Liu & Srikantha, 2022), and the optimal kernel can even change with operating conditions (Pareek & Nguyen, 2021). By learning a data-driven similarity metric, DVA remains flexible and robust, avoiding the need for manual kernel selection and tuning.

1.1 Related Works

Prior-data Fitted Networks: There are several works on PFNs (Hollmann et al., 2025; Wang et al., 2025; Nagler, 2023; Adriaensen et al., 2023; Li et al., 2023), most of which rely on the Transformer architecture (Vaswani et al., 2017), applying self-attention over concatenated (x_i, y_i) embeddings. While this design has shown strong performance on certain tasks, it presents two key limitations that remain largely unaddressed. First, these works implicitly assume that the Transformer backbone is crucial to PFN success. Second—and more importantly—the standard attention mechanism does not scale well to high-dimensional problems: training becomes unstable, and performance deteriorates quickly as dimensionality increases Wang et al. (2025). Although Wang et al. (2025) introduced a Boosting-based method that splits the dataset into smaller subsets and trains an ensemble of PFNs, this was primarily intended to handle longer input sequences, not to address high-dimensional scaling issues or architectural dependence of PFNs.

Physical Equation Surrogates for Power Flow: Efficiently solving power flow equations is crucial for integrating renewable energy and electric vehicles Barry et al. (2022), a key area where machine learning can help mitigate climate change Rolnick et al. (2022). Faster analytical approximations of nonlinear alternating current power flow (ACPF) equations exist, but come at the cost of accuracy Molzahn et al. (2019). To address this, various ML models—including physics-informed methods—have been developed for ACPF learning and uncertainty quantification Chen et al. (2025). Among these, GPs have gained prominence for building explainable surrogates with closed-form predictions Tan et al. (2025). However, such modeling is extremely sensitive to GP kernels, as shown by Liu & Srikantha (2022) by showing that specialized kernels outperform standard options like squared-exponential or polynomial kernels Pareek & Nguyen (2021).

2 Background

2.1 Gaussian Processes (GP)

GP is a non-parametric, probabilistic framework for modeling functions from a functional space perspective. Given data (x_i, y_i), we assume $y_i = f(x_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2)$. For N inputs $x = (x_1, \ldots, x_N)$, the function values $\boldsymbol{f}(x) = [f(x_1), \ldots, f(x_N)]^\top$ follow a joint Gaussian distribution as $\boldsymbol{f}(x) \sim \mathcal{N}(\boldsymbol{m}(x), K(x, x'))$, with mean function $m(x)$ and covariance matrix $K(x, x')$. By definition, a GP is a collection of random variables such that any finite subset is jointly Gaussian, denoted $f \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot))$. The observation distribution is then $\mathbb{P}(y) \sim \mathcal{N}(m(x), K(x, x') + \sigma_\varepsilon^2 \mathrm{I})$, with I as identity matrix of appropriate size. Thus, given training data y at x, the predictive distribution of $f^\star$ at a new input $x^\star$ is Gaussian with closed-form mean and covariance. The choice of kernel (covariance) function $k(x, x')$ encodes prior assumptions about f, while hyperparameters are typically learned by maximizing the marginal log-likelihood. However, the closed-form of exact inference only works when the likelihood is Gaussian and inversion of kernel matrix presents training bottleneck. A key property of GPs, central to the design of DVA, is that the kernel $k(\cdot, \cdot)$ measures similarity solely between input data Williams & Rasmussen (2006).

2.2 Prior-Data Fitted Network (PFN)

PFNs (Müller et al., 2022) are neural predictors trained to approximate the *posterior predictive distribution* (PPD) of a Bayesian model in a single forward pass. Rather than fitting a single static dataset, a PFN is trained on multiple *synthetic datasets*– drawn from a prior over data-generating mechanisms. Given a prior distribution $p(\mathcal{D})$ over supervised learning tasks, PFNs repeatedly sample datasets $\mathcal{D}^k \cup \{(\mathbf{x}^k, \mathbf{y}^k)\} \sim p(\mathcal{D})$ for $k = 1 \ldots K$ and train the model to minimize

$$\text{Negative Log-Likelihood (NLL)} \quad \ell_\theta = \sum_{k=1}^{K} \big[-\log q_\theta(\mathbf{y}^k \mid \mathbf{x}^k, \mathcal{D}^k) \big]. \tag{1}$$

Here, $q_\theta(\cdot)$ represents the Transformer prediction. This procedure treats entire datasets $\mathcal{D}$'s as inputs and optimizes the model parameters θ to predict a held-out label conditioned on the remaining data. Thus, fitting the PPD without explicitly computing posteriors. Further, PFNs represent the output distribution using a discrete set of *buckets* (bins) for the target $\mathbf{y}$, essentially posing regression as a classification problem. After training, the PFN performs *amortized Bayesian inference*: given a new dataset $\mathcal{D}_{\text{train}}$ and query point x_{test}, it outputs $q_{\theta^\star}(y_{\text{test}} \mid x_{\text{test}}, \mathcal{D}_{\text{train}}) \approx p(y_{\text{test}} \mid x_{\text{test}}, \mathcal{D}_{\text{train}})$ in a single forward pass, where $\theta^\star$ is optimal Transformer parameters Müller et al. (2022).

2.3 Limitations of Existing PFN Architectures with Joint Attentions

PFNs offer a promising framework for amortized Bayesian inference, though their application to *high-dimensional regression* has so far been relatively limited Hollmann et al. (2025); Wang et al. (2025). The common recipe of using a Transformer backbone that performs self-attention over joint $(\mathbf{x}, \mathbf{y})$ embeddings has a scaling issue. The design choice of representing each training example (x, y) in PFNs as a joint embedding $\text{enc}(x) + \text{enc}(y)$ can be traced back to the way attention-based models have historically treated their basic units of computation. In natural language processing, the Transformer architecture Vaswani et al. (2017) encodes each token as a self-contained representation, i.e. "token as full carrier of information". In this lineage, PFNs adopt the same strategy by concatenating or summing input and output encodings to form a single token while removing positional encoding Müller et al. (2022). Below, we examine the this PFN recipe's structural limitations.

Firstly, attention computation based on the standard PFN initial embedding strategy, of joint representation $\mathbf{z}_i = \text{enc}(\mathbf{x}_i) + \text{enc}(\mathbf{y}_i)$, forces the model to measure across both input features and target output values. As the input dimension grows, pairwise distances concentrate, and the margin between true and spurious neighbors of the input shrinks (the "curse of dimensionality"). Thus, variation in output $\mathbf{y}_i$, unrelated to input proximity, can dominate similarity calculations. Empirically, we observe significant degradation beyond about 10D input in our experiments discussed in Sec. 4.

Further, we can also analyze this dimensionality limitation from a bias perspective as the joint embedding breaks *localization*. The Transformer computes similarity (via dot-product attention) between queries and these mixed embeddings in which the label $\mathbf{y}_i$ contributes equally. This conflicts with theoretical results Nagler (2023), which show that only local samples should influence posterior estimates. Consequently, incorporating joint input–output attention introduces additional bias, which becomes more pronounced in higher dimensions due to the concentration of pairwise distances. In view of these limitations, we propose a simple decoupled value attention (DVA) which keeps the localization intact.

3 Decoupled-Value Attention

We propose **Decoupled-Value Attention (DVA)**, an input-localized attention mechanism for training PFNs. The proposed DVA is structurally aligned with GP inference by treating input $\mathbf{x}$ and output $\mathbf{y}$ separately at the attention stage. We enforce a strict separation of roles: attention affinities (queries and keys) are computed solely from the inputs, while the aggregated information (values) comes from the corresponding outputs– during both PFN training and prediction. Below, we explain DVA mathematically along with comparative assessment against Vanilla Attention (VA) Müller et al. (2022) and a kernel-based attention Wang & Others (2025).

Consider a PFN training dataset $\mathcal{D} = \{X, \mathbf{y}\}$ where $X \in \mathbb{R}^{N \times d}$ and $\mathbf{y} \in \mathbb{R}^{N \times 1}$ with N input samples of dimension d. In DVA we calculate query Q, key K and value V as

$$Q = W_q\, \varphi_x(X), \quad K = W_k\, \varphi_x(X), \quad V = W_v\, \varphi_y(\mathbf{y}), \tag{2}$$

with encoders φ_x, φ_y and trainable linear maps $W_q \in \mathbb{R}^{d \times d_k}, W_k \in \mathbb{R}^{d \times d_k}, W_v \in \mathbb{R}^{d \times 1}$. Then, attention is then computed as $\text{Att}(Q, K, V) = \text{softmax}\big(QK^T/\sqrt{d_k}\big)\, V$. Now, via equation 2, proposed DVA enforces that similarity is calculated purely in input space, while labels flow only through values. This is unlike VA used in PFNs, which mixes inputs and outputs in a joint embedding.

Training: During training we simulate inference by masking one or more labels from the dataset Müller et al. (2022). The unmasked pairs $\mathcal{D}_{\text{cx}} = \{(x_i, y_i)\}_{i=1}^{N_{\text{context}}}$ form the *context set*, while the

Table 1: Comparison of Attention Mechanisms for PFNs

Component	Vanilla	Kernel-based	DVA (ours)
Input Emb.	$\text{enc}(x_i) + \text{enc}(y_i)$	x_i, y_i separately	x_i, y_i separately
Query	From z_i	$\phi(x_i)$	From $\text{enc}(x_i)$
Key	From z_j	$\phi(x_j)$	From $\text{enc}(x_j)$
Value	From z_j	From $\text{enc}(y_j)$	From $\text{enc}(y_j)$
Limitation	Unstable in high-D	Requires kernel choice	Absent output cues

Vanilla attention is taken from PFN literature Müller et al. (2022); Hollmann et al. (2022)

masked inputs $X_{\text{te}} = \{x_j\}_{j=1}^{M}$ form the *queries*. From the context we build

$$K_{\text{tr}} = W_k \varphi_x(X_{\text{cx}}), \quad V_{\text{tr}} = W_v \varphi_y(\mathbf{y}_{\text{cx}}), \tag{3}$$

where, X_{cx} and $\mathbf{y}_{\text{cx}}$ are matrix and vector forms of $\mathcal{D}_{\text{cx}}$ respectively. Further, from the test (masked) inputs query Q_{te} in matrix from and labels are predicted by attending H_{te} to the context as :

$$Q_{\text{te}} = W_q \varphi_x(X_{\text{te}}); \quad H_{\text{te}} = \text{softmax}\Big(\tfrac{Q_{\text{te}} K_{\text{tr}}^{\top}}{\sqrt{d_k}}\Big) V_{\text{tr}}. \tag{4}$$

A head $g(\cdot)$ maps H_{te} to a predictive distribution, and training minimizes the NLL (equation 1) of the true labels to learn parameters of the network as explained in Müller et al. (2022).

Inference: At test time, the mechanism is identical except that *training dataset* forms the *context set* and the "queries" are now the real unseen inputs i.e. we do not know the true output $\mathbf{y}$ for test inputs. Given a training dataset $\mathcal{D}_{\text{train}} \equiv \mathcal{D}_{\text{context}}$ for unseen function learning via GP, we obtain the predicted output with $Q_\star = W_q \varphi_x(X_\star)$ for test input $X_\star$ as

$$\hat{y}_{\text{test}} = g\Big(\text{softmax}\left(Q_\star K_{\text{tr}}^T / \sqrt{d_k}\right) V_{\text{tr}}\Big) \tag{5}$$

This ensures that the weight assigned to each context point's value $v(y_i)$ depends only on the similarity between the query input $\mathbf{x}_\star \in X_\star$ and the context input $\mathbf{x}_i \in \mathcal{D}_{train}$, mirroring the GP's use of an input-space kernel function, as discussed in the following subsection. The key differences between attention approaches are summarized in Table 1.

3.1 Localization Effect of DVA and Alignment with GP Inference

In DVA, the attention weights for a new test point $\mathbf{x}_\star$ are given by $\text{softmax}\big(\langle Q, K\rangle / \sqrt{d_k}\big)$, where $\langle \cdot, \cdot \rangle$ is the standard dot product. Explicitly, attention weights are

$$\alpha_i(\mathbf{x}_\star) = \frac{\exp\Big(\big\langle W_q \varphi_x(\mathbf{x}_\star),\, W_k \varphi_x(X_i)\big\rangle \Big/ \sqrt{d_k}\Big)}{\sum_{j=1}^{n} \exp\Big(\big\langle W_q \varphi_x(\mathbf{x}_\star),\, W_k \varphi_x(X_j)\big\rangle \Big/ \sqrt{d_k}\Big)} \tag{6}$$

From equation 6, it is clear that that affinities are determined entirely via relationship between the test input $\mathbf{x}_\star$ and context inputs X_i. Unlike joint embeddings $\phi(x, y)$ in VA, the labels y_i do not enter into the similarity measure and only appear downstream through the values as in equation 2. This separation implies that the weight placed on a output y_i depends solely on how well X_i aligns with $\mathbf{x}_\star$ in the projected input space. Thus, the softmax distribution $\alpha_i(\mathbf{x}_\star)$ concentrates mass on a neighborhood of $\mathbf{x}_\star$ because the projection matrices and encoders are trained to align nearby inputs with high value of inner product and push apart distant inputs. Consequently, altering labels (outputs) attached to distant inputs cannot affect the prediction asymptotically, which is exactly the localization property required in Theorem 5.4 of Nagler (2023). Thus, proposed DVA recovers input-space localization by construction, while still using the standard softmax form of attention.

We now discuss how DVA aligns with GP inference. As discussed in Section 2.1, GPs model all possible function realizations as zero-mean Gaussians with covariance defined by a kernel, i.e., $f \sim \mathcal{N}(\mathbf{0}, K(X, X'))$, where X is the input and $K(\cdot, \cdot)$ is the kernel matrix over input pairs. Note that parameterization of the possible function family only depends on the input. Further, for a given kernel hyperparameters, the mean prediction $\mu(\cdot)$ of GP is given as a weighted sum of training dataset outputs with weights solely depending upon inputs Williams & Rasmussen (2006):

$$\mu(x_\star) \;=\; \sum_{i=1}^{N_{\text{train}}} \beta_i(x_\star)\, y_i, \quad \text{where } \beta(x_\star) = k(x_\star, X)\left[K(X, X) + \sigma^2 I\right]^{-1}. \tag{7}$$

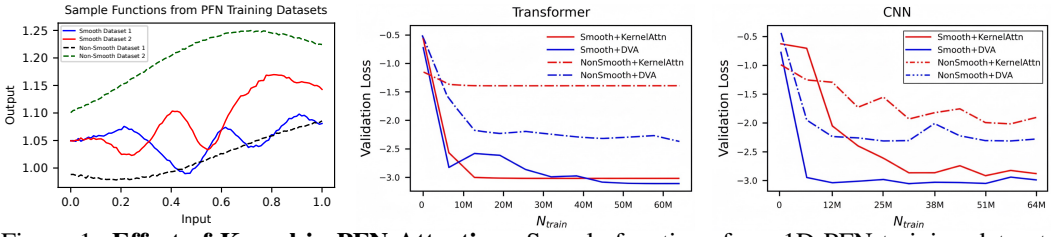

Figure 1: **Effect of Kernel in PFN Attention:** Sample functions from 1D PFN training datasets (Left). Validation loss for smooth and non-smooth functions with Kernel-based Attention and DVA with Transformer (Middle) and CNN (Right).

Following equation 5, 6 and 7, the attention weights in DVA can be interpreted as normalized kernel weights that depend only on the inputs. As in kernel smoothing Tsai et al. (2019), the exponential inner product in $\alpha(\cdot)$ of equation 6 acts as a positive kernel on the input space, with effective bandwidth governed by the scale of the projections and the $1/\sqrt{d_k}$ factor. Thus, similar to GP mean prediction, DVA predictions are obtained as weighted sums of training outputs where the weights are determined entirely by input similarity. Readers can refer to Tsai et al. (2019) for more discussion on the relationship between kernel and attention mechanism.

Here, we want to highlight that DVA's softmax produces non-negative, normalized weights ($\sum \alpha_i = 1$), whereas the GP coefficients $\beta_i(\cdot)$ have no positivity constraint. This limitation is mitigated by subsequent PFN layers (e.g., the final head $g(\cdot)$) and by encoding outputs in the value V, which together help adjust the DVA output toward the true GP posterior mean. This construction shows that DVA's architecture implements a predictor of the form "*input-only similarities produce weights, which combine label-dependent values*," precisely matching the dependency structure of a GP.

Another attention choice for PFNs can be kernel-inspired attentions, which relate GP mean weights $\beta(\cdot)$ and PFN attention weights $\alpha(\cdot)$ more closely– while maintaining input localization by decoupling input and output as in DVA. However, if the input affinities are forced through a fixed kernel function, the PFN will become kernel dependent. As discussed before, identifying the best performing kernel is non-trivial and often requires tailoring kernels to specific function classes Liu & Srikantha (2022). Therefore, it is not advisable to *hard-wire* a particular kernel in PFN design.

To test the effect of kernel dependence on PFN performance, we design a simple Gaussian kernel (radial basis function, RBF) similarity for attention Williams & Rasmussen (2006). We emphasize that this formulation is not equivalent to exact GP kernel regression but rather introduces RBF-style distance-based affinities in place of dot-product similarities Choromanski et al. (2022); Shen et al. (2021). The kernel-based attention assigned to a query input $\mathbf{x}_\star$ is then given by

$$\alpha_i(\mathbf{x}_\star) = \frac{\exp\left(-\gamma \left\| W_q \varphi_x(\mathbf{x}_\star) - W_k \varphi_x(X_i) \right\|^2\right)}{\sum_{j=1}^{n} \exp\left(-\gamma \left\| W_q \varphi_x(\mathbf{x}_\star) - W_k \varphi_x(X_j) \right\|^2\right)} \tag{8}$$

This distance-based attention in equation 8 is more aligned to the RBF kernel; however, it loses flexibility to learn input-localization via training. Thus, the model inherits kernel and γ dependence, which may not be suitable for a broader class of functions. To validate this limitation of the kernel-based attention, we test PFN performance with both DVA and this attention. We attempt to learn two classes of functions with different levels of smoothness as shown in Figure 1. The results demonstrate that while kernel-based attention can match DVA in effectively learning smooth functions aligned with the RBF kernel, it significantly underperforms on non-smooth functions generated using the linear-periodic kernel. More details on this experiment can be found in Appendix C.

4 Numerical Results and Discussion

In this section, we present numerical experiments demonstrating the behavior of PFNs equipped with the proposed DVA and with CNN backbone. The results show that with DVA, PFNs a) train with lower validation loss or residual bias, b) both CNN and Transformer perform comparably as architecture, underscoring that attention governs training behavior more than backbone architecture and c) remain scalable for learning in complex physical systems. These findings provide empirical support for the theoretical arguments in Section 3.1. Complete experimental details, including

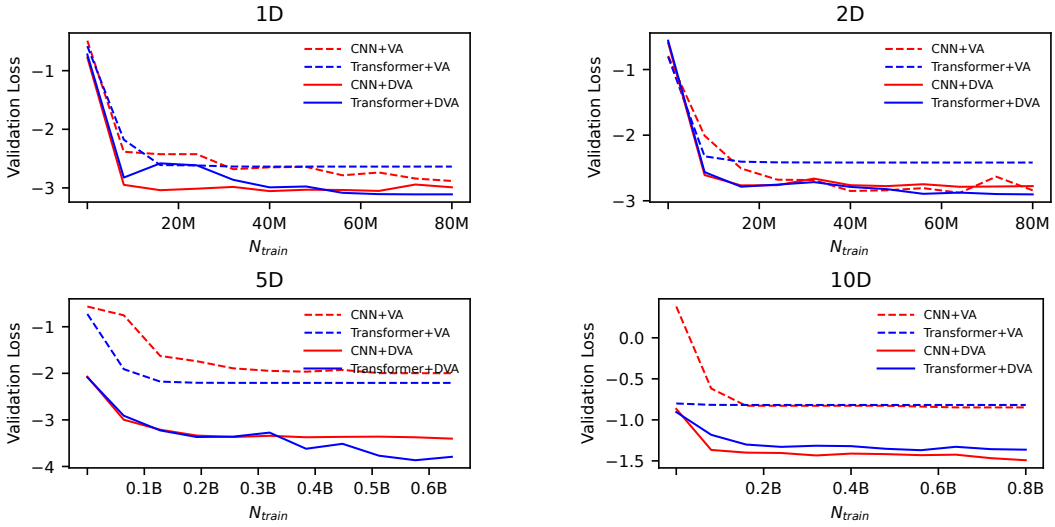

Figure 2: **Bias Reduction in PFN Training:** Validation loss (NLL) behavior with number of training points for various PFNs (Number of training points = epochs $\times$ steps per epoch $\times$ batch-size $\times$ dataset size. Dataset size is 100 for 1D/2D, 400 for 5D and 500 for 10D PFN). Validation loss was calculated on 64 out-of-sample datasets and Transformer + VA is taken from Müller et al. (2022).

architecture choices, hyperparameter selection, and data generation procedures, are provided in the Appendix B, while additional results are provided in Appendix D.

4.1 Bias Reduction and CNN PFNs

To assess the bias reduction capability of the proposed DVA, we perform PFN learning and testing for datasets of increasing input dimensionality (1D, 2D, 5D, and 10D). Figure 2 plots the validation loss as a function of the training set size N_{train} for CNN and Transformer backbones equipped with both VA and the proposed DVA.

Bias Reduction: Across all input dimensions, the curves with VA (dashed lines) saturate at visibly higher loss values, revealing a persistent residual bias that does not diminish even with large training data. In contrast, DVA-based PFNs (solid lines) consistently converge to lower loss plateaus, demonstrating that DVA mitigates this bias, with negligible increase in variance. The gap becomes especially pronounced in higher dimensions (5D and 10D), where VA-equipped models remain strongly biased, while DVA-equipped models continue to benefit from additional training samples. Further, In the 10D case, we observe an even more striking phenomenon: both CNN+VA and Transformer+VA curves flatten almost immediately after training begins, indicating that the models essentially stop learning. This rapid saturation at high validation loss reflects that VA-equipped PFNs become unable to adapt in higher-dimensional regimes, effectively collapsing to a biased estimator. In contrast, their DVA counterparts continue to decrease loss with additional training data, showing that DVA alleviates this high-dimensional learning obstruction. Another noteworthy trend is visible at the beginning of training. For low-dimensional tasks (1D and 2D), the initial validation loss is nearly identical across VA and DVA models, with improvements arising only as training progresses. However, in the higher-dimensional cases (5D and 10D), DVA-equipped PFNs already begin with a substantially lower validation loss compared to their VA counterparts, and this advantage compounds as more data are observed. This behavior suggests that DVA not only accelerates convergence but also reduces the asymptotic bias floor, thereby enabling PFNs to faithfully approximate the target physical mappings. To ensure robustness, the 10-dimensional (10D) models were trained multiple times. The corresponding results are provided in the Appendix D.

Comparative Discussion on CNN- and Transformer-Based PFNs: To analyze the effect of backbone architecture on PFN performance, we study 1D, 2D, 5D, and 10D inputs for two network architectures: Transformer Müller et al. (2022) and CNN. Performance is measured using mean squared error (**MSE**) and validation loss at convergence (**Final Val Loss**), summarized in Table 2. GP results are also included as a baseline for MSE. Each backbone is trained with both VA and the proposed DVA. The results show that attention choice has a larger effect than backbone choice.

Table 2: Mean squared error (MSE) and final validation loss across input dimensions.

	MSE					Final Val Loss			
	GP	VA		DVA		VA		DVA	
		CNN	**Tx**	**CNN**	**Tx**	**CNN**	**Tx**	**CNN**	**Tx**
1D	1.02e-04	1.07e-04	1.28e-04	1.37e-04	1.23e-04	−2.88	−2.63	−3.05	−3.11
2D	1.29e-04	1.23e-04	1.78e-04	2.26e-04	1.97e-04	−2.91	−2.41	−2.77	−2.91
5D	3.42e-06	7.59e-05	2.43e-04	5.04e-05	2.84e-05	−2.29	−2.04	−3.56	−4.05
10D	3.47e-04	3.55e-03	3.56e-03	5.49e-04	4.98e-04	−0.81	−0.81	−1.51	−1.37

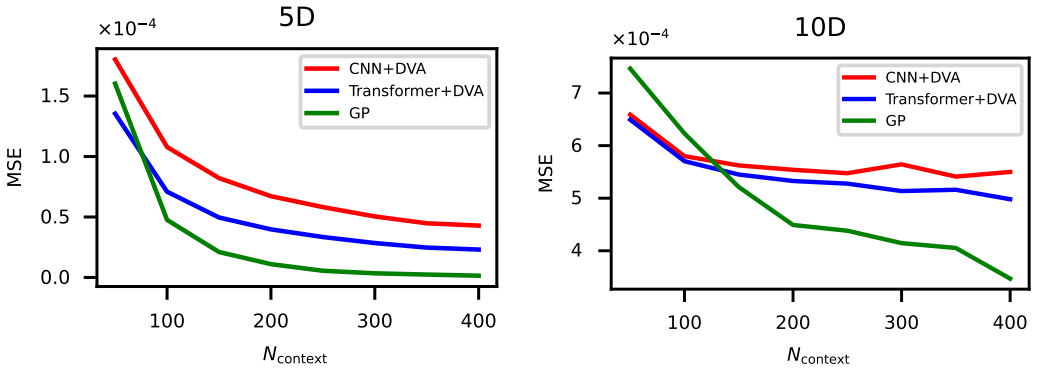

Figure 3: **Comparison with GP:** MSE for 5D and 10D PFNs as a function of context size. All models are tested using $n_{\text{test}} = 500$, for $N_{context}$. The results show that error consistently decreases with larger context sizes, and that CNN- and Transformer-based PFNs with DVA approach the performance of exact GP inference even in higher dimensions. Exact GP baselines were fit using `scikit-learn` with $N_{context}$ training samples.

For instance, at 5D, switching a Transformer from VA to DVA reduces MSE from 2.43×10^{-4} to 2.84×10^{-5}, closer to the GP baseline (3.42×10^{-6}). The validation loss also improves from -2.04 to -4.05—an absolute gain of 2.01 ($\approx 98.5\%$ relative improvement)—while the CNN-Transformer spread under VA is only 0.25 ($\approx 10.9\%$). Similarly, at 10D, CNN and Transformer MSEs drop by nearly an order of magnitude under DVA, far exceeding the architecture gap under VA. These results indicate that CNN- and Transformer-based PFNs perform comparably once the attention mechanism is specified, with DVA further pushing performance toward GP quality in higher dimensions.

Comparison with GP: In line with the observations made by authors in Müller et al. (2022), our experiments show that PFNs achieve performance comparable to exact GP inference. As seen in Table 2, PFNs with the proposed DVA consistently move closer to GP performance than those with VA—for instance, in the 10D setting, DVA reduces the MSE from 3.55e-03 (CNN-VA) to 5.49e-04, compared to the GP baseline of 3.47e-04. Importantly, the performance differences between architectures (CNN and Transformer) are relatively minor compared to the gains achieved by changing the attention mechanism, further reinforcing the hypothesis that the effect of attention on PFN performance is far greater than architecture.

We also evaluated the behavior of PFN inference as a function of N_{context}, i.e., *How PFN performance improves as the number of available samples increases at inference time?*. As shown in Figure 8, PFNs (both CNN and Transformer-based) with the proposed DVA exhibit a consistent decrease in error with increasing context, closely matching the performance of exact GPs in low-dimensional settings. In higher dimensions, GPs maintain a slight advantage, consistent with the trends observed in the training performance analysis. It is important to note that the observed performance gap between 5D and 10D (for both PFNs and GP) arises largely because of limited samples per dataset for 10D model (400 for 5D and 500 for 10D). Similar plots for 1D, 2D PFNs MSE, along with MAE and maximum error for all dimension PFNs are given in the Appendix D.

4.2 Physics Equation Learning

Rosenbrock Function: To benchmark our method against a well–established baseline, we conduct experiments on the 5-dimensional Rosenbrock function Rosenbrock (1960), a standard test problem

Table 3: **Voltage prediction on a 64D power-flow test bed:** Trained on 500 samples; evaluated on 4,500 test samples. The time results (t) are for evaluating on all 32 node voltages, and the MSE and MAE correspond to the maximum values across the buses.

	Exact GP			CNN + DVA			Transformer + DVA		
ΔLoad	**MSE**	**MAE**	t	**MSE**	**MAE**	t	**MSE**	**MAE**	t
5%	2.2e-7	0.0004	10.88	4.5e-7	0.0005	0.13	1.5e-6	0.001	0.17
10%	3.5e-7	0.0004	10.94	1.7e-6	0.001	0.13	2.8e-6	0.001	0.17
30%	3.2e-7	0.0005	11.61	1.5e-5	0.003	0.14	1.6e-5	0.003	0.17
50%	2.2e-7	0.0003	11.89	4.2e-5	0.005	0.13	4.4e-5	0.005	0.17

in optimization that is often interpreted as a nonlinear potential energy landscape with a curved valley structure Akian et al. (2022). GPs are a natural choice for such comparisons because they provide a flexible non-parametric model with uncertainty quantification, and they have been widely benchmarked on Rosenbrock and related test functions in the GP literature Xu et al. (2025). Results indicate that 5D PFN with Transformer+DVA shows MSE 6.8e-4 and CNN+DVA achieves MSE of 1.6e-3, without any retraining, see Table 8 in Appendix D for detailed results.

Power Flow Learning: In this experiment, we model the IEEE 33-bus distribution system by treating the real and reactive power demands at each of the 32 load buses as uncertain inputs (same experiment design as described in Pareek & Nguyen (2021); Liu & Srikantha (2022)). This results in a 64-dimensional input space (32 active + 32 reactive loads). Now the learning task is to predict the corresponding steady-state bus voltage magnitude— effectively learning the nonlinear AC power flow mapping from loads to voltages i.e. Voltage $= f(\text{Loads})$ (See equation 10 in Appendix A.2). Table 3 benchmarks power flow surrogates under varying load perturbations from 5% to 50%. Exact GP achieves the lowest MSE and MAE values across all cases, but requires training 32 times (one for each node), which becomes *infeasible for repeated queries under changing load conditions*. In contrast, both PFNs CNN+DVA and Transformer+DVA trade a modest increase in error for dramatic efficiency gains—over $80\times$ faster than GPs—while maintaining voltage prediction accuracy at the order of 10^{-3}, sufficient for practical grid analysis. Further, the prediction error decreases as more training (context) samples are provided, with both CNN+DVA and Transformer+DVA converging to near-identical performance as illustrated in Figure 7 (Appendix D). These results highlight that while GPs remain the gold standard for accuracy, DVA-equipped PFNs offer a scalable alternative, enabling high-dimensional, uncertainty-aware power flow learning in real time for complex networked systems. Moreover, because voltages are in per-unit, MSE and MAE values around 10^{-3} are practically acceptable. In real systems, measurement devices typically have least counts of 10^{-3} p.u., so an error of 10^{-3} in a 1 kV system corresponds to just 1 V Molzahn et al. (2019). We also note that, consistent with the 10D case in Figure 2, PFNs equipped with vanilla attention failed to train sufficiently for this 64D problem and thus did not yield meaningful results. Training time for 64D models is approximately 14 hours for both Transformer and CNN-based PFNs on NVIDIA 4500ADA GPU.

5 Conclusions and Future Work

In this work, we propose Decoupled-Value Attention (DVA) to train Prior-Data Fitted Networks (PFNs), particularly for GP inference for high-dimensional inputs. Through experimental studies, we show that the proposed DVA halves the residual bias in PFN learning for 5D and 10D settings, and enables PFNs constructed with either CNNs or Transformers to achieve comparable accuracy once equipped with the attention mechanism. Leveraging these advantages, DVA enables PFNs to serve as highly efficient surrogates for high-dimensional power flow learning. On the IEEE 33-bus system with 64-dimensional load variations, DVA-equipped PFNs attained voltage prediction accuracy in the order of 10^{-5} while delivering more than an $80\times$ speedup over exact GP.

Future work will focus on scaling PFNs to even larger power networks and higher-dimensional uncertainty spaces, particularly in the context of power flow uncertainty quantification and planning problems. Another promising direction is the design of architectures that explicitly support input-space localization beyond attention mechanisms, allowing PFNs to capture neighborhood structure in physical systems more faithfully. We would also like to work on DVA limitation of having no output affinities, while maintaining localization. Together, these efforts can push PFNs toward practical deployment for real-time, uncertainty-aware decision making in modern power systems.

REFERENCES

Steven Adriaensen, Herilalaina Rakotoarison, Samuel Müller, and Frank Hutter. Efficient bayesian learning curve extrapolation using prior-data fitted networks. In *Advances in Neural Information Processing Systems 37*. NeurIPS, 2023.

J-L Akian, Luc Bonnet, Houman Owhadi, and Éric Savin. Learning "best" kernels from data in gaussian process regression. with application to aerodynamics. *Journal of Computational Physics*, 470:111595, 2022.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019. URL https://arxiv.org/abs/1907.10902.

Neil Barry et al. Risk-aware control and optimization for high-renewable power grids. *arXiv preprint arXiv:2204.00950*, 2022.

Long Minh Bui, Tho Tran Huu, Duy Dinh, Tan Minh Nguyen, and Trong Nghia Hoang. Revisiting kernel attention with correlated gaussian process representation. *arXiv preprint arXiv:2502.20525*, 2025.

Minghua Chen, Xiang Pan, Jiawei Zhao, and Min Zhou. Ml opf wiki: Machine learning for solving power flow equations. https://energy.hosting.acm.org/wiki/index.php/ML_OPF_wiki#Machine_Learning_for_Solving_Power_Flow_Equations, 2025. Accessed: 2025-08-24.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2022. URL https://arxiv.org/abs/2009.14794.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations (ICLR)*, 2021.

Carleton Coffrin, Russell Bent, Kaarthik Sundar, Yeesian Ng, and Miles Lubin. Powermodels.jl: An open-source framework for exploring power flow formulations. In *2018 Power Systems Computation Conference (PSCC)*, pp. 1–8, June 2018. doi: 10.23919/PSCC.2018.8442948.

Constantinos Daskalakis, Petros Dellaportas, and Aristeidis Panos. How good are low-rank approximations in gaussian process regression? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6463–6470, April 2022. doi: 10.1609/aaai.v36i6.20598.

David Duvenaud. The kernel cookbook. https://www.cs.toronto.edu/~duvenaud/cookbook/, n.d. Accessed: 2025-09-24.

Noah Hollmann, Samuel Müller, Eyke Hüllermeier, and Asja Fischer. Learning to learn with prior-data fitted networks. In *International Conference on Learning Representations (ICLR)*, 2022.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, jan 2025. doi: 10.1038/s41586-024-08328-6.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19565–19594. PMLR, jul 2023.

Jingyuan Liu and Pirathayini Srikantha. Kernel structure design for data-driven probabilistic load flow studies. *IEEE Transactions on Smart Grid*, 13(4):2679–2689, 2022. doi: 10.1109/TSG.2022.3159579.

Daniel K Molzahn, Ian A Hiskens, et al. A survey of relaxations and approximations of the power flow equations. *Foundations and Trends® in Electric Energy Systems*, 4(1-2):1–221, 2019.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations (ICLR)*, 2022.

Thomas Nagler. Statistical foundations of prior-data fitted networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25660–25676. PMLR, jul 2023.

Parikshit Pareek and Hung D Nguyen. A framework for analytical power flow solution using gaussian process learning. *IEEE Trans. on Sustainable Energy*, 13(1):452–463, 2021.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021.

David Rolnick, Priya L Donti, Lynn H Kaack, Kelly Kochanski, Alexandre Lacoste, Kris Sankaran, Andrew Slavin Ross, Nikola Milojevic-Dupont, Natasha Jaques, Anna Waldman-Brown, et al. Tackling climate change with machine learning. *ACM Computing Surveys (CSUR)*, 55(2):1–96, 2022.

H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, 1960. doi: 10.1093/comjnl/3.3.175.

Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 3531–3539, 2021.

Bendong Tan, Tong Su, Yu Weng, Ketian Ye, Parikshit Pareek, Petr Vorobev, Hung Nguyen, Junbo Zhao, and Deepjyoti Deka. Gaussian processes in power systems: Techniques, applications, and future works. *arXiv preprint arXiv:2505.15950*, 2025.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: a unified understanding of transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Author Wang and Others. Cross-kernel attention for efficient sequence modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Yuxin Wang, Botian Jiang, Yiran Guo, Quan Gan, David Wipf, Xuanjing Huang, and Xipeng Qiu. Prior-fitted networks scale to larger datasets when treated as weak learners. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 1090–1098. PMLR, may 2025.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, and et al. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

Zhitong Xu, Haitao Wang, Jeff M. Phillips, and Shandian Zhe. Standard gaussian process is all you need for high-dimensional bayesian optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.

Appendix

A Physics Equation Benchmarks

A.1 Rosenbrock Function

For our baseline experiments, we use the 5D–dimensional Rosenbrock function, defined as Rosenbrock (1960)

$$f(\mathbf{x}) = \sum_{i=1}^{d-1} \left[100\,(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right], \quad \mathbf{x} \in [-1, 1]^d,\ d = 5. \tag{9}$$

We normalize the input vectors $\mathbf{x}$ to the unit hypercube $[0, 1]^5$ before training, and outputs are standardized using Z-score normalization. For GP testing, we employ a Gaussian process surrogate with a RBF kernel with automatic relevance determination (ARD) length-scales Williams & Rasmussen (2006).

A.2 AC Power Flow Problem

The alternating current power flow (ACPF) problem is fundamental to power grid analysis, as it computes the steady-state voltages, currents, and power flows that satisfy Kirchhoff's laws under given nodal injections. Unlike ACOPF, which optimizes generator set-points, ACPF focuses on feasibility by solving the nonlinear power flow equations, which are given as:

$$P_i = \Re\left\{V_i \sum_{j \in N} Y_{ij}^* V_j^*\right\}, \quad Q_i \quad = \Im\left\{V_i \sum_{j \in N} Y_{ij}^* V_j^*\right\}, \quad \forall i \in N, \tag{10}$$

where P_i and Q_i are the real and reactive power injections at bus i, V_i is the complex bus voltage, and Y_{ij} are the elements of the bus admittance matrix.

In our setting, we explicitly consider uncertainty at each bus in both real and reactive power injections. For an IEEE 33-bus system Pareek & Nguyen (2021); Liu & Srikantha (2022), with the first bus designated as the slack bus (zero load), this leads to a 64-dimensional uncertainty input vector capturing nodal variability across all other buses. We follow the standard ACPF model used in `PowerModels` Coffrin et al. (2018), and we use `compute_ac_pf` function of `PowerModels.jl` to generate dataset.

A.2.1 Power Flow Learning with GPs

In the power flow learning setting, the goal is to approximate the mapping from net load vectors to system states such as bus voltages (magnitude and angle). This mapping, though implicitly defined by the nonlinear power flow equations (equation 10), is treated here as a supervised regression task where the net load serves as input and the voltage response as output.

We adopt a GP model to capture this relationship:

$$y(\boldsymbol{x}) = f_s(\boldsymbol{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2), \tag{11}$$

where $y(\boldsymbol{x})$ is the observed voltage at a node for load vector $\boldsymbol{x}$. With GP priors, $\boldsymbol{y}(\boldsymbol{x}) \sim \mathcal{GP}(0, K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_\varepsilon^2 I)$, and the kernel K encodes correlations between operating points. Owing to the smoothness of voltages as a function of load, the squared exponential kernel has been widely used in prior work Tan et al. (2025).

GP-based approximations have been shown to outperform analytically approximated linearized models in capturing power flow uncertainty Pareek & Nguyen (2021) and are favored over other learning methods such as DNN due to closed-form approximation nature of GP, and predictive variance availability etc. For more details on power flow modeling and GP surrogate of it, readers can refer to Tan et al. (2025).

B Implementation and Architectural Details

Synthetic Prior Data Generation: To assess different attention mechanisms in Prior-Data Fitted Networks (PFNs), we use synthetic datasets generated from GP priors, following Müller et al. (2022). Each regression dataset consists of input–output pairs (x, y), with inputs sampled uniformly and outputs drawn from a multivariate Gaussian with an RBF kernel having lengthscale, variance (output scale), and observation noise variance as hyperparameters. The inputs are normalized using Z-score normalization, while outputs are shifted to a range of 0.8–1.2 for all datasets.

For classification-based objectives, the continuous output space is discretized into buckets derived from quantiles of GP-sampled outputs Müller et al. (2022). Each bucket corresponds to a categorical class index, allowing regression-style PFN training to be cast into classification under a Riemannian distribution loss formulation. This strategy preserves ordering structure while making outputs compatible with categorical training setups. See Müller et al. (2022) for more information in this.

Table 4: Number of buckets for different input dimensions PDFs.

Dimensions	**1D**	**2D**	**5D**	**10D**	**64D**
Number of Buckets	100	100	500	500	500

Transformer Architecture: We used a Transformer architecture where input features and targets are first projected into a shared embedding space and then processed through a series of encoder blocks combining attention, feedforward layers, residual connections, and layer normalization. The model's hyperparameters, includes the model width (embedding size), number of attention heads, number of encoder blocks, and hidden dimension of the feedforward layers and all parameters are initialized using Xavier uniform initialization.

CNN-Attention Architecture: The CNN-attention model also encodes features and targets into a shared embedding space using linear layers. The embeddings are then processed through a stack of convolutional-attention blocks, where each block applies single dimension depthwise convolutions followed by attention, combined with residual connections and layer normalization. Finally, a small DNN head maps the processed embeddings to the model outputs. Key hyperparameters are model width, number of layers, and kernel size.

Hyperparameters Selection: To ensure fair evaluation across architectures and embedding dimensions, we employed `Optuna` Akiba et al. (2019) for automated hyperparameter tuning. Key parameters such as model width, hidden dimension size, number of attention blocks, number of heads, and dropout rate were jointly optimized, with `AdamW` and a linear warmup followed by step-wise decay. Each trial involved computing training and validation losses on PFN tasks, and `Optuna`'s pruning strategies enabled efficient exploration of the search space. The best-performing configurations were selected based on initial validation loss over 1000 trials, while training loss was also tracked to assess stability.

Table 5: Transformer Hyperparameter search ranges used in `Optuna`.

Model Width	**Hidden Dim**	**Attention Blocks**	**Heads**	**Dropout**
32–256	128–1024	1–4	2–8	0.0–0.5

Table 6: CNN Hyperparameter search ranges used in `Optuna`.

Model Width	**Layers**	**Kernel Size**
32–256	1–6	3, 5, 7

Parameter Calculation: To compute the total learnable parameters (Table 7), we loaded each model's state dictionary, where keys represent layers and values are tensors of weights or biases. For each tensor, we used `numel()` from `pytorch` and summed these values to obtain the total parameter count.

Table 7: Number of trainable parameters across input dimensions (same for VA and DVA).

Model	1D	2D	5D	10D
CNN	9,060	9,092	36,116	36,276
Tx	316,645	316,673	878,198	688,854

C Kernel-based Attention Experiment Details

For the experiments shown in Figure 1, we generated synthetic datasets using both smooth and non-smooth kernels, and trained 1D Transformer and CNN models (with the same backbone architectures as used in Table 2) for both kernel attention and DVA. The smooth datasets were sampled from an RBF kernel, which promotes locality and smoothness in the function. In contrast, the non-smooth datasets were generated using a linear–periodic kernel as discussed in Duvenaud (n.d.), which combines a linear trend with a periodic component, producing oscillatory patterns with irregular variations and reduced smoothness.

D Additional Results

Figure 4 presents the validation loss curves for five different models trained on the 10-dimensional input setting, plotted against the number of training samples N_{train}. Each curve represents the mean validation loss over six independent training runs, with shaded regions indicating the minimum and maximum loss values across these runs, illustrating the variability and robustness of the training process. As observed, the CNN + VA and Transformer + VA models show poorer training performance, consistent with the results discussed in the main paper. In contrast, the CNN + DVA and Transformer + DVA models exhibit significantly improved and more stable training behavior. These findings highlight the robustness of our implementation.

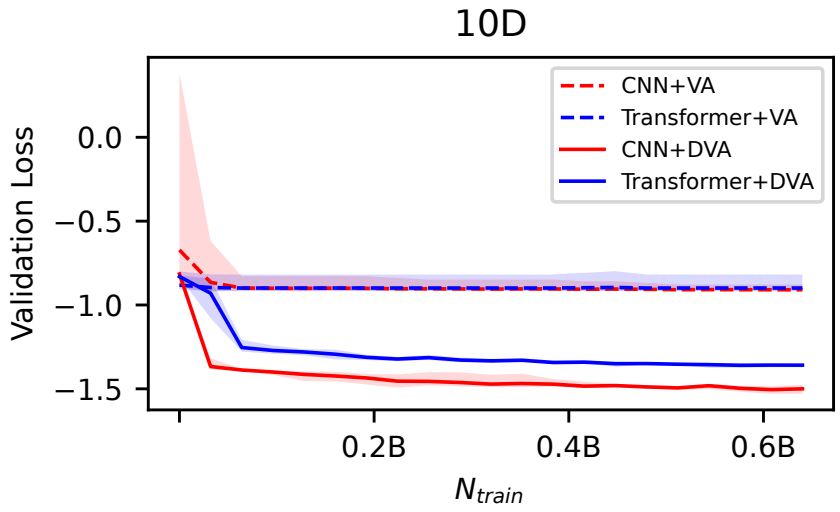

Figure 4: Robustification study for training 10D PFNs. Curves show the mean validation loss over 6 runs; shaded regions represent the minimum and maximum loss values across runs.

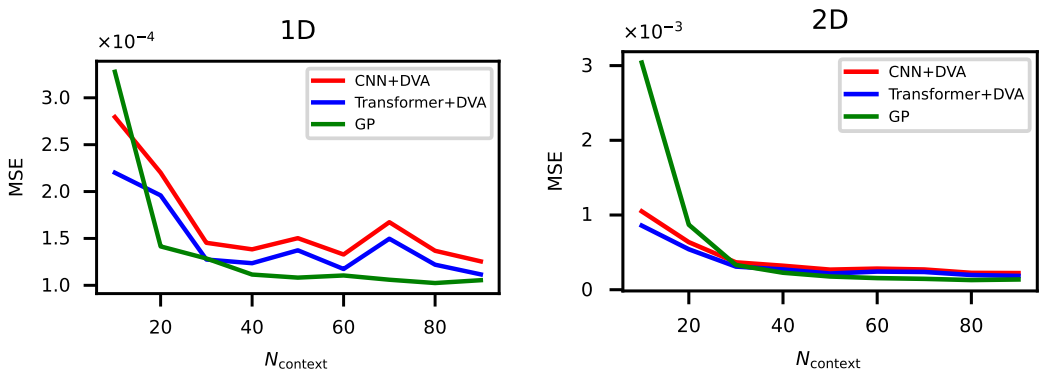

Figure 5: **Comparison with GP for 1D & 2D PFNs:** MSE as a function of context size. All models are tested using $n_{\text{test}} = 500$, for $N_{context}$. The results show that error consistently decreases with larger context sizes, and that CNN- and Transformer-based PFNs with DVA approach the performance of exact GP inference even in higher dimensions. Exact GP baselines were fit using `scikit-learn` with $N_{context}$ training samples.

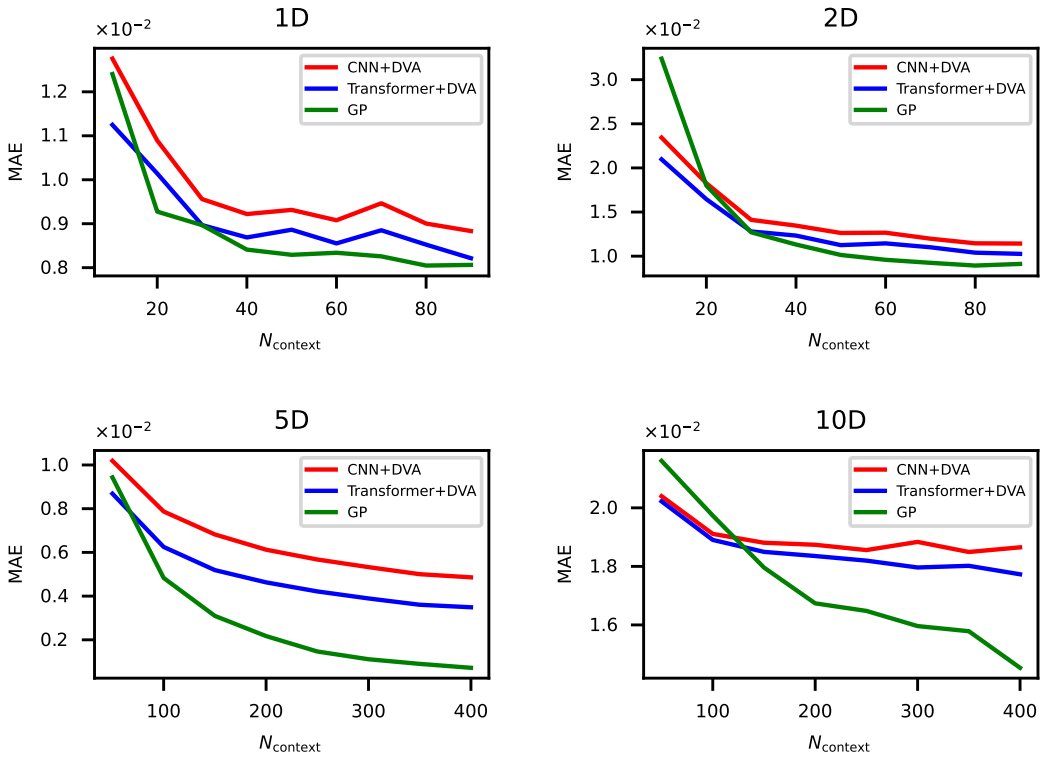

Figure 6: **MAE for PFNs across context sizes:** Mean absolute error as a function of N_{context}. Models were tested with $n_{\text{test}} = 500$ points per dataset. 1D and 2D PFNs were trained with 100, while 5D and 10D PFNs used 500 points per dataset. Error decreases with larger context sizes, and CNN- and Transformer-based PFNs with decoupled-value attention (DVA) approach the performance of exact GP, even in higher dimensions. Exact GP baselines were fit using `scikit-learn`.

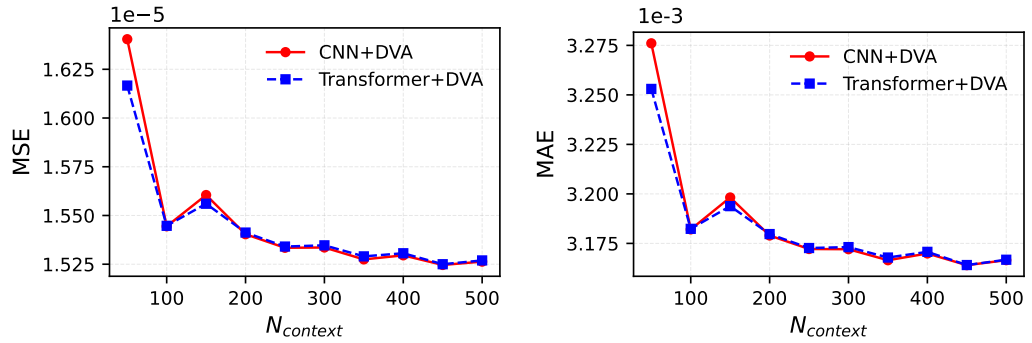

Figure 7: **Learning performance on the 64D power-flow task:** The plots show variation of MSE (**left**) and MAE (**right**) with the number of training context samples (N_{context}). Both CNN+DVA and Transformer+DVA exhibit decreasing errors with additional context and converge to near-identical accuracy. Testing is performed on 4500 out-of-sample testing data of voltages.

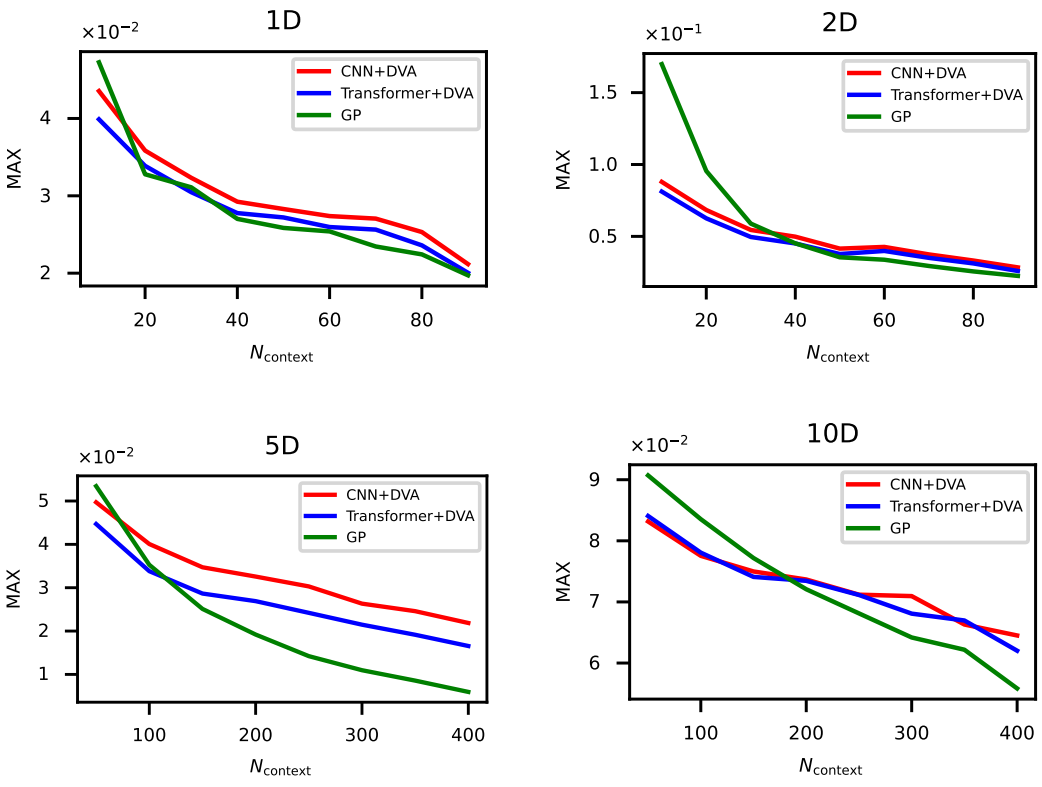

Figure 8: **Maximum error for PFNs across context sizes:** Maximum error as a function of N_{context}. Models were tested with $n_{\text{test}} = 500$ points per dataset. 1D and 2D PFNs were trained with $n = 100$, while 5D and 10D PFNs used $n = 500$ points per dataset. Error decreases with larger context sizes, and CNN- and Transformer-based PFNs with decoupled-value attention (DVA) approach the performance of exact GP, even in higher dimensions. Exact GP baselines were fit using `scikit-learn`.

Table 8: Comparison of MSE values for different models with increasing training points for Rosenbrock Function approximation.

Training Points	GP	CNN+DVA	Transformer+DVA
10	1.02e-2	8.65e-3	9.12e-3
50	5.76e-3	5.41e-3	4.01e-3
100	3.92e-4	4.13e-3	2.41e-3
200	7.70e-5	3.06e-3	1.84e-3
500	1.00e-7	1.61e-3	6.83e-4