Scaling Prior-Data Fitted Networks for Physical System Learning

Parikshit Pareek, Ph.D.

Assistant Professor
Department of Electrical Engineering
Indian Institute of Technology Roorkee (IIT Roorkee)

Time Equation for ML Models

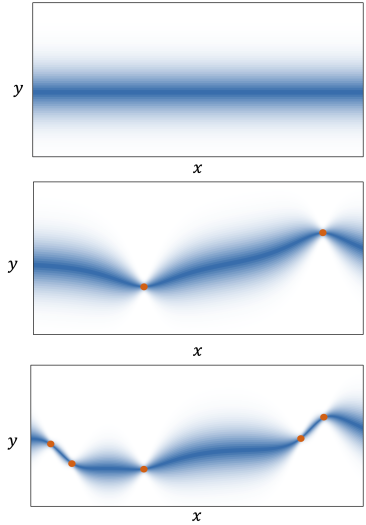
$$T_{data} + T_{train} + T_{predict}$$

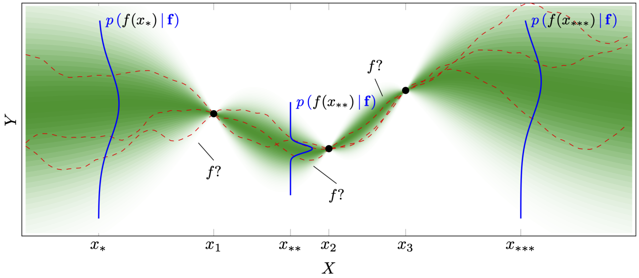
Training Data

$$\mathcal{M}(\theta_i | \mathcal{D}_i) \; : \; \texttt{Need Retraining for Each } \mathcal{D}.$$

Model Weights

Gaussian Process Working Idea: In Brief





The distribution of $f(x^\star)$, $f(x^{\star\star})$ and $f(x^{\star\star\star})$ for the three inputs $x^\star$, $x^{\star\star}$ and $x^{\star\star\star}$, conditional on the observed values $f(x_1)$, $f(x_2)$ and $f(x_3)$.

Andreas Lindholm *et.al.* "An introduction to Gaussian process regression" Department of Information Technology, Uppsala University.

Prior-Data Fitted Networks (PFNs)

Goal: Perform Bayesian Inference in a single forward pass.

Why? Bayesian inference is slow, and repeated retraining makes the computational cost prohibitive

How? Learn to approximate the *Posterior Predictive Distribution (PPD)* $p(y|x, \mathcal{D})$

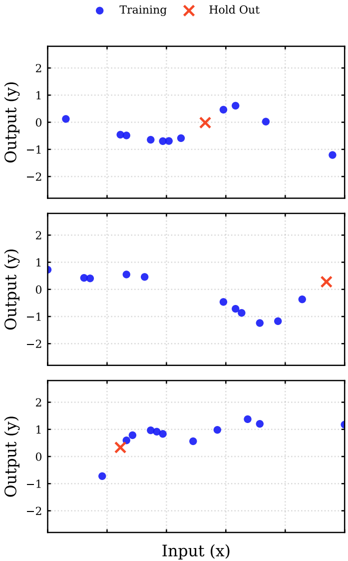
Amortized Bayesian Inference

Given a new dataset $\mathcal{D}_{\text{context}}$ and query point x_{test}, it outputs

$$q_{\theta^\star}(y_{\text{test}} \mid x_{\text{test}}, \mathcal{D}_{\text{context}}) \approx p(y_{\text{test}} \mid x_{\text{test}}, \mathcal{D}_{\text{context}})$$

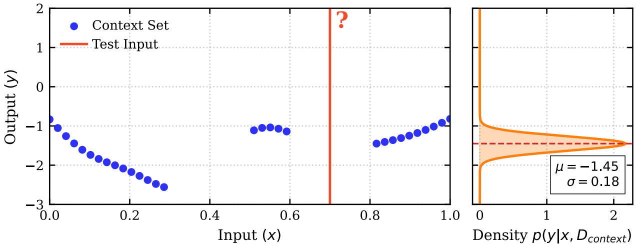
in a single forward pass, where $\theta^\star$ are the optimal PFN parameters.

PFN Idea



Train PFN using Negative Log-Likelihood Loss over hold-out samples

$$\ell_\theta = \sum_{k=1}^{K} \left[-\log q_\theta(\mathbf{y}^k \mid \mathbf{x}^k, \mathcal{D}^k) \right]$$



PFN: Training & Inference

Sample prior datasets $D^{(i)} \sim p(\mathcal{D})$

$D^{(1)} = D^{(1)}_{train} \cup \{(x^{(1)}_{test}, y^{(1)}_{test})\}$

$\vdots$

$D^{(K)} = D^{(K)}_{train} \cup \{(x^{(K)}_{test}, y^{(K)}_{test})\}$

Train the PFN by minimizing

$-\sum_{i=1}^{K} \log q_\theta(y^{(i)}_{test} | x^{(i)}_{test}, D^{(i)}_{train})$

PFN with parameters θ^*

Actual context dataset & test input

$(D_{context}, x_{test})$

$q_{\theta^*}(y_{test} | x_{test}, D_{context}) \approx p(y_{test} | x_{test}, D_{context})$

Samuel Müller, Noah Hollmann, *et.al* "Transformers can do bayesian inference". ICLR 2022.

One + Two Questions

Scalability

Scalability: Can PFNs perform GP-style inference for 50D regression?

No - standard PFNs don't scale beyond $\sim$10D

Attention Encoding: Localization

Why is everyone encoding $x + y$ in PFNs? How does this affect localization?

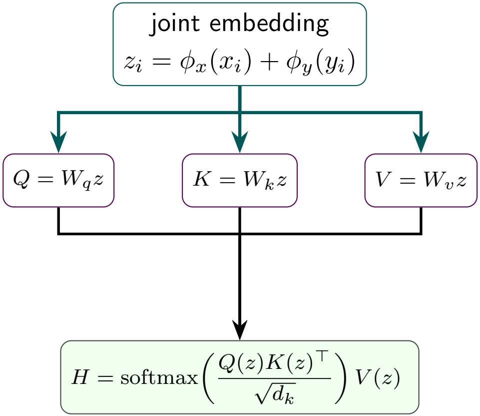
Localization: Nearby points provide more info about the value at an unknown point than distant ones.

Backbone vs Attention

Are transformers all we need, or can PFNs also be built using CNNs etc.?

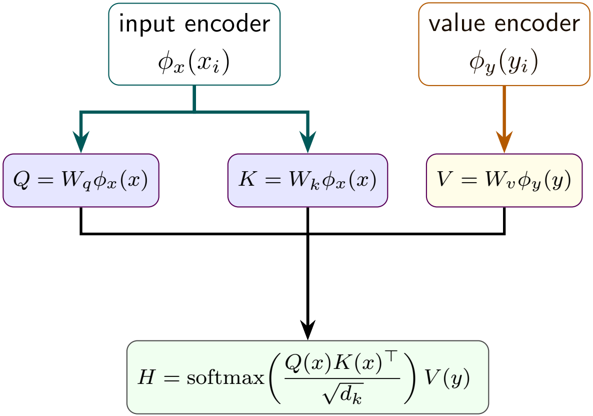
Main Idea: Decouple Input and Output

Vanilla Attention



Mixing Input & Output

Decoupled-Value Attention (DVA)



Keeping Input & Output Separate

Theorem: Enforcing Localization

Theoretical Result

For DVA with linear embeddings, the attention weight $\alpha_i(x_\star)$ (for a test input $x_\star$) assigned to a context point x_i is proportional to a **Mahalanobis RBF Kernel**:

$$\alpha_i(x_\star) \propto \exp\left(-\frac{1}{2\tau}\|(x_\star - x_i)\|_A^2\right)$$

Implication:

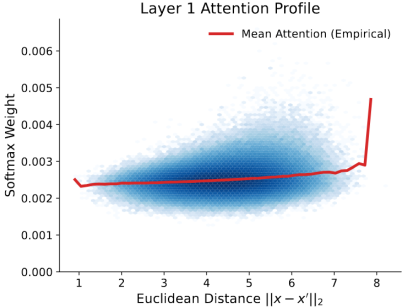
- As the distance $\|x_\star - x_i\|$ increases, the attention weight decays **exponentially**.
- This mathematically forces the Model to behave like a GP Kernel: localization.
- This holds true for any backbone architecture.

$\|u\|_A^2 = u^T A u$

Empirical Evidence: 10D Input Space

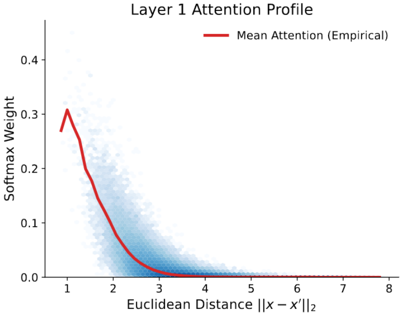
Does the theory hold in practice? We look at the attention weights in a 10D task.

Vanilla Attention



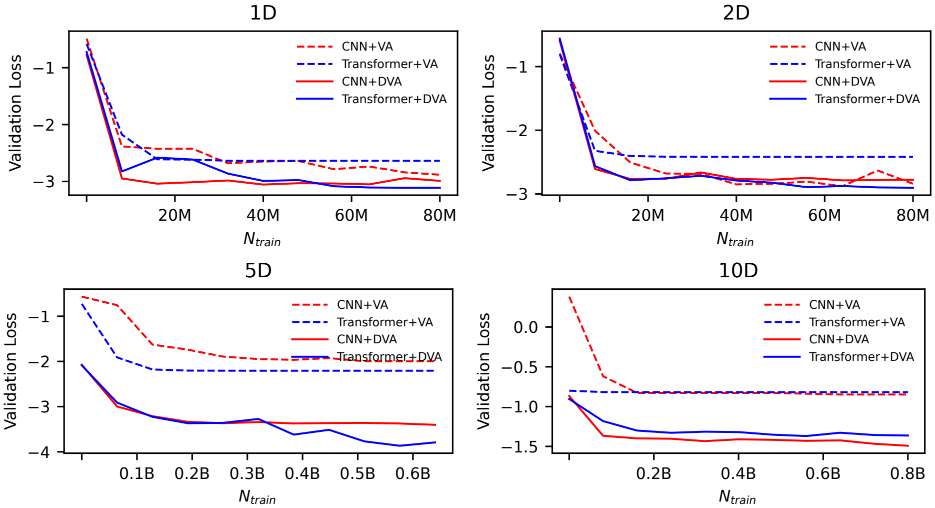
No Localization
Weights are flat/uniform.
The model fails to localize inputs.

DVA (Ours)

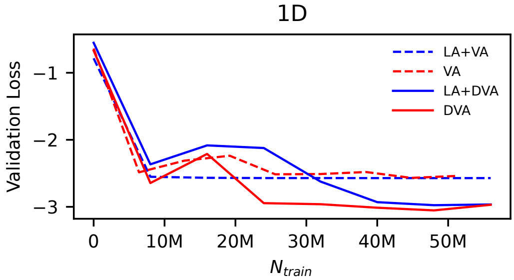


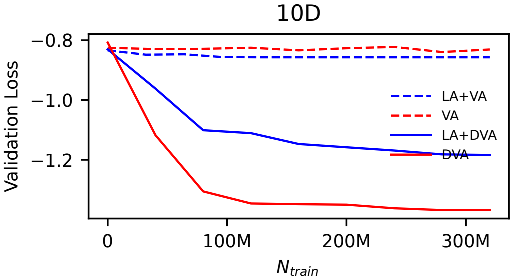
Exponential Decay
Weights drop as distance increases.
(Matches Theorem 1)

DVA Outperforms VA: Scale and Error



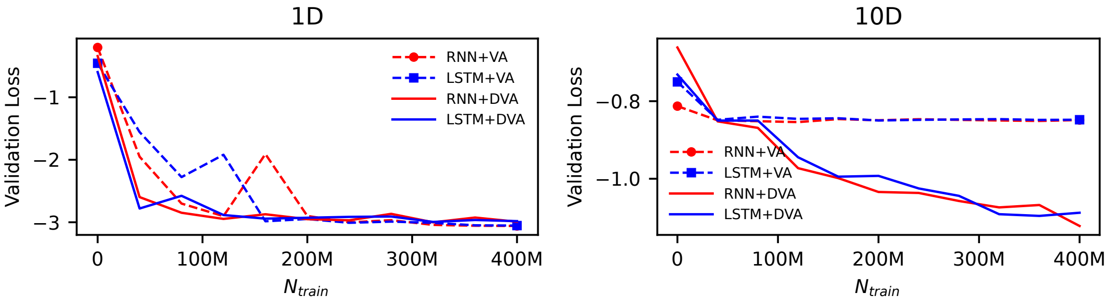
Decoupling Helps with Linear Attention Too





`Softmax DVA` $\gg$ `Linear DVA` $\gg$ `Softmax VA` $\gg$ `Linear VA`

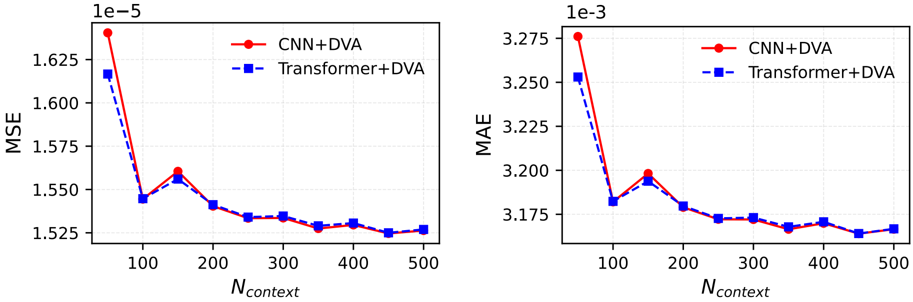
Attention Agnostic: RNN/LSTM Results



With DVA, PFN training performance is nearly identical across backbones
: CNN, RNN, LSTM, Transformer.

64D Power-Flow Learning

- Model: IEEE 33-bus AC power flow (64D inputs: 32 real + 32 reactive loads) $\rightarrow$ 32 bus voltages.
- Exact GP (per-bus) yields lowest MSE/MAE but is *slow* for real-time (needs 32 GPs).
- CNN+DVA and Transformer+DVA PFNs trade small accuracy loss (MAE $\sim 10^{-3}$) for $\sim 80\times$ faster inference.



The PPD of Power Flow

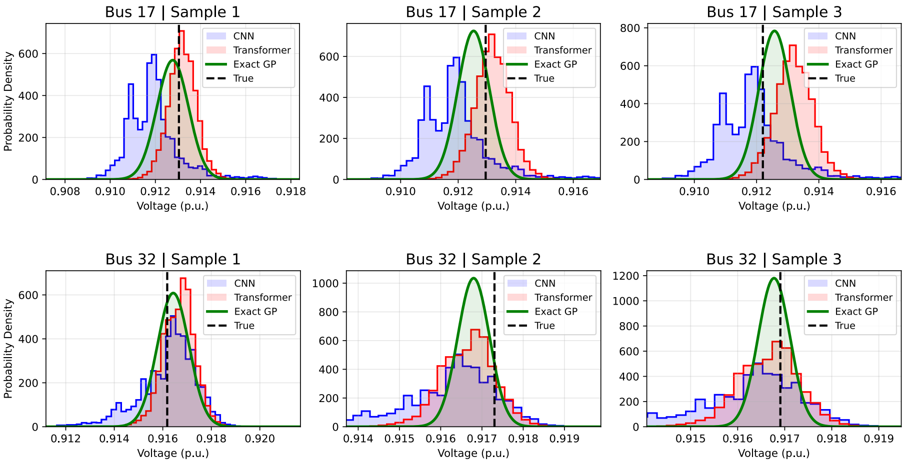
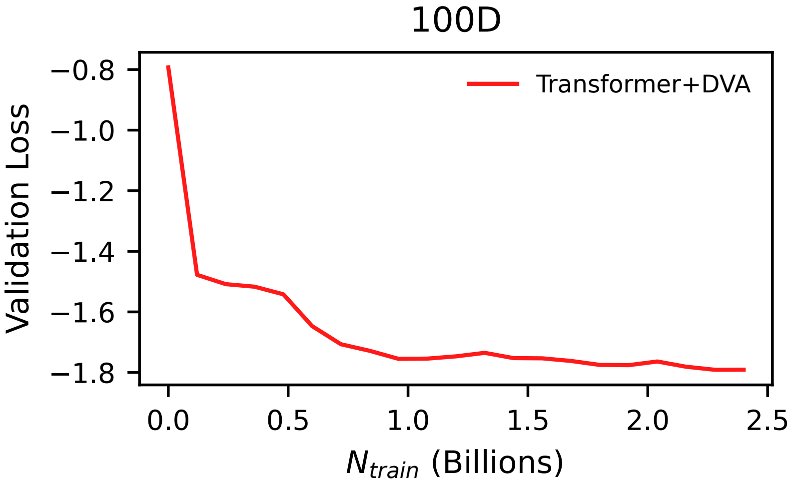


Figure: Comparing the PPD for three distinct samples for (Top) Bus 18. (Bottom) Bus 33

100D is Possible Too



Summary & Takeaways

- **Decoupled-Value Attention:** A specialized attention rule that preserves input locality and mirrors GP inference.
- **Bias Reduction:** DVA cuts PFN bias by $> 50\%$ in high-D tasks (5D, 10D) compared to vanilla attention.
- **Attention $\gg$ Backbone:** CNN/RNN/LSTM PFNs with DVA perform on par with Transformers, despite far fewer parameters.
- **Scaling to 64D:** DVA-enabled PFNs successfully learn 64D physical equations at $\sim 80\times$ GP speed.

Big Question

What if we could make a PFN which works for all ND GP Regressions?
A FOUNDATION MODEL FOR REGRESSION

Best Part of The Work!

My Coauthors are Third Year UG Students @IITR!

Kaustubh Sharma
B.Tech Electrical Engg.
kaustubh202.github.io

Simardeep Singh
B.Tech Metallurgical Engg.
linkedin.com/in/simar7220

Lab Website

Full Pre-Print

Funding Support

Faculty Initiation Grant

Anusandhan National Research Foundation

PM Early Career Research Grant-2025