OUT-OF-SUPPORT GENERALISATION VIA WEIGHT-SPACE SEQUENCE MODELLING

Roussel Desmond Nzoyem
University of Bristol
Bristol, UK
rd.nzoyemngueguin@bristol.ac.uk

ABSTRACT

As breakthroughs in deep learning transform key industries, models are increasingly required to extrapolate on datapoints found outside the range of the training set, a challenge we coin as out-of-support (OoS) generalisation. However, neural networks frequently exhibit catastrophic failure on OoS samples, yielding unrealistic but overconfident predictions. We address this challenge by reformulating the OoS generalisation problem as a sequence modelling task in the weight space, wherein the training set is partitioned into concentric shells corresponding to discrete sequential steps. Our WeightCaster framework yields plausible, interpretable, and uncertainty-aware predictions without necessitating explicit inductive biases, all the while maintaining high computational efficiency. Emprical validation on a synthetic cosine dataset and real-world air quality sensor readings demonstrates performance competitive or superior to the state of the art. By enhancing reliability beyond in-distribution scenarios, these results hold significant implications for the wider adoption of artificial intelligence in safety-critical applications.

1 INTRODUCTION

Over the past decade, deep learning has revolutionised pivotal industries, ranging from natural language processing [Vaswani et al., 2017; Guo et al., 2025] and autonomous driving [Badue et al., 2021; Dhaif and El Abbadi, 2024] to drug discovery [Jumper et al., 2021; Blanco-Gonzalez et al., 2023]. Despite vast data quantities required for training, unfamiliar testing scenarios undermining model reliability persist. For instance, a language model trained exclusively on English text may fail catastrophically on French queries, exemplifying the out-of-distribution (OoD) challenge. Within these problems, very few have captured the interest of the scientific community as much as cases where testing and training supports are disjoint, which we characterise as *out-of-support* (OoS).

Traditional OoS solutions rely on incorporating inductive biases, such as enforcing known dynamics [Nzoyem et al., 2025; Rackauckas et al., 2020] or prioritising discriminative features [Keshtmand et al., 2026]. However, these methods falter when valid inductive biases are unavailable. Strategies like Distributionally Robust Optimisation [Kuhn et al., 2025] and meta-learning [Caruana, 1997; Hospedales et al., 2021] similarly necessitate prior knowledge of potential test distributions. While non-parametric approaches like Gaussian Processes [Williams and Rasmussen, 1995] offer inherent uncertainty estimates, they scale poorly to large datasets.

Leveraging recent advances in sequence modelling and weight-space learning [Schürholt et al., 2024; Nzoyem et al., 2026], we propose **WeightCaster** : a framework recasting OoS generalisation tasks as forecasting problems through the partitioning of the training set into concentric *shells* that we call "rings". By mapping each ring to a discrete time step, we learn sequential dynamics extrapolatable to the test set. Our contributions are threefold: (1) a computationally efficient, parametric, interpretable, and inductive bias-free framework for OoS generalisation; (2) a linearisation strategy enabling the framework to provide uncertainty estimates both in-distribution (InD) and OoS; and (3) empirical validation on synthetic sinusoidal and real-world air quality experiments revealing superior or competitive performance at low parameter count.

1.1 Problem Setting

In this typical machine learning problem, we are interested in learning the parametrised mapping $f_\theta : \mathbb{R}^{D_x} \mapsto \mathbb{R}^{D_y}$ such that $y = f_\theta(x), \forall (x, y) \sim p$, where θ is the array of model parameters, and p denotes the joint probability distribution of the data. We denote by p_{tr} the training data distribution from which a *fixed* number of training inputs $X^{\text{tr}} \in \mathbb{R}^{N_{\text{tr}} \times D_x}$ are sampled along with their corresponding outputs $Y^{\text{tr}} \in \mathcal{Y}$.[1]

OoS requires models to maintain predictive power on input samples $X^{\text{te}} \in \mathbb{R}^{N_{\text{te}} \times D_x}$ from p_{te} defined in regions of the input space where the training density is zero. Concretely, this means $\text{Supp}(X^{\text{tr}}) \cap \text{Supp}(X^{\text{te}}) = \emptyset$, where Supp denotes the support of a discrete set of points, i.e., the range of $\mathbb{R}^{D_x}$ covered by these points.

1.2 Related Work

OoS problems have traditionally been studied under the umbrella of OoD generalisation. Distributionally Robust Optimisation (DRO) [Kuhn et al., 2025] remains one of the strongest approaches, as it trains models to perform well under worst-case scenarios. DRO, however, requires the specification of a complex ambiguity set of possible testing distributions p_{te}. Engression [Shen and Meinshausen, 2025] is designed to bridge the gap between traditional regression and full distributional modelling, specifically targeting the support-shift limitations of standard DRO. Beyond simple point estimates, Engression captures the stochastic nature of the data generation process itself.

Recent literature has shifted toward Invariant Learning [Arjovsky, 2020] and causal discovery to identify features that remain relevant outside the training support [Keshtmand et al., 2026]. Concurrently, non-parametric methods, most notably Gaussian Processes (GPs) [Williams and Rasmussen, 1995], have been revitalised for OoS tasks due to their principled approach to uncertainty. Unlike most parametric models that collapse in unseen regions, non-parametric approaches revert to a prior belief when the test data escapes the support of the training set. This comes at a cost, and our approach addresses the main limitation of GPs by requiring significantly less computational resources to fit large training sets X^{tr}.

Another family of OoS generalisation techniques is meta-learning [Caruana, 1997; Hospedales et al., 2021; Nzoyem, 2025; Finn et al., 2017], which aims to fine-tune (a subset of) the model's parameters θ at test time on a new task. Like DRO, however, a successful meta-learner requires some inductive bias of what the target testing distribution could be. In contrast, our approach attempts to extrapolate beyond the training domain by providing the most likely OoS predictions under a sequence model assumption, thereby eliminating the need for explicit inductive biases and test-time fine-tuning.

Our approach is also heavily connected to the nascent area of weight-space learning (WSL) [Schürholt et al., 2024]. Remarking on the growing capacity of public model repositories such as CivitAI and HuggingFace [Jain, 2022], WSL has recently captured the attention of the deep learning community, showcasing huge breakthroughs in implicit neural representations [Dupont et al., 2022], generalisation error prediction [Unterthiner et al., 2020], and sequence modelling [Nzoyem et al., 2026].

2 Method

Fig. 1 summarises our proposed **WeightCaster** using a 1D sine wave as our example. We begin by selecting an anchor point from the training dataset, and we decompose the training domain into successive *hyperspherical shells* (equivalent to intervals in 1D or annuli in 2D). While their geometric structure might be different depending on D_x and the chosen distance metric, we refer to these shells as "rings" for simplicity. Each ring corresponds to a step in a sequence model, and a weight-space sequence model learns to predict suitable weights for each step.

In the following paragraphs of this section, we detail the ingredients of the WeightCaster framework. We begin by outlining the input domain decomposition and weight-space sequence modelling strategies. Deterministic training and inference algorithms are discussed, followed by a stochastic framework for uncertainty estimation.

[1] $\mathcal{Y}$ could be a subset of $\mathbb{R}^{D_y}$ for regression, or a discrete set such as $\{0, 1\}$ for classification.

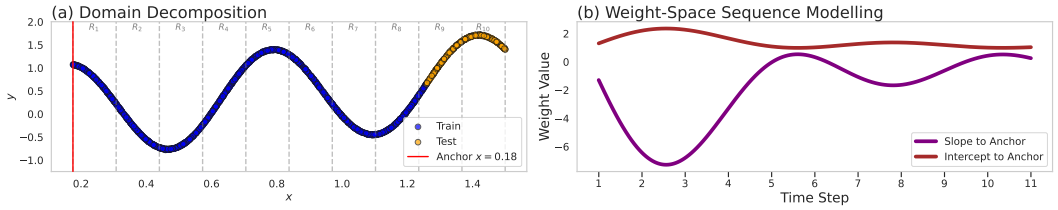

Figure 1: Illustration of the two main steps of the WeightCaster framework for sinusoidal extrapolation. **(a)** First, an anchor point is chosen and the input domain $\mathbb{R}^1$ is decomposed into $T = 10$ "rings", here clearly delineated as intervals. Within each ring, we consider a simple linear model $\hat{y} = \theta^1 \cdot x + \theta^2$, where $\theta = [\theta^1, \theta^2]^T$ contains the slope and intercept to anchor, respectively. **(b)** Optimal weights $\{\theta_t\}_{t=1}^{T_{\text{tr}}}$ for the data in each ring are subsequently computed by fitting a weight-space sequence model, one ring corresponding to one time step. Suitable weights for OoS datapoints are obtained by rolling out the sequence model for time steps $t \geq T_{\text{tr}}$ (in this example, $T_{\text{tr}} = 9$).

2.1 Domain Decomposition

Domain decomposition is the first stage of the WeightCaster framework. Since the input domain is a metric space, we can define a distance metric $d(\cdot, \cdot)$ on $\mathbb{R}^{D_x}$. Next, an anchor point $\underline{x}$ is chosen. For each point in either the training or testing dataset, we can compute its distance to $\underline{x}$.

Depending on the desired granularity, we construct $T > 1$ rings $\{R_t\}_{t=1}^{T}$ of *equal* radii δ. Using precomputed distances to the anchor, we assign a ring identifier to each data point. The data within the ring R_t is denoted X_t^{tr} for train samples, and X_t^{te} for testing samples. Necessarily, T_{tr} that identifies the outermost ring containing a training datapoint is such that $T_{\text{tr}} \leq T$.

Conventional machine learning seeks to learn a single model $\hat{y} = f_\theta(x), \forall x \in X^{\text{tr}} \triangleq \bigcup_{t=1}^{T_{\text{tr}}} X_t^{\text{tr}}$. In contrast, WeightCaster seeks to fit one model θ_t for each ring, such that $\hat{y} = f_{\theta_t}(x), \forall x \in X_t^{\text{tr}}$. We achieve this via weight-space sequence modelling.

2.2 Weight-Space Sequence Modelling

The domain decomposition above naturally provides a sequential ordering for the weights θ_t, which we use to formulate an initial value problem (IVP). Specifically, our optimisation procedure is the weight-space learning problem defined as follows[2]

$$\phi^*, \theta_1^* = \arg\min_{\phi, \theta_1} \sum_{t=1}^{T_{\text{tr}}} \mathop{\mathbb{E}}_{(x,y)\sim(X_t^{\text{tr}}, Y_t^{\text{tr}})} \left[\ell(f_{\theta_t}(x), y)\right], \quad \text{subject to} \quad \{\theta_t\}_{t=2}^{T_{\text{tr}}} = G_\phi(\theta_1), \tag{1}$$

where:

- $\ell(\cdot, \cdot)$ is a discrepancy function, such as mean squared error (MSE) or cross-entropy;
- $G_\phi(\cdot)$ is a higher level neural functional, a *state-to-sequence*[3] model parametrised by ϕ.

Our goal is to not only find optimal parameters ϕ^*, but also the initial weights θ_1^* for the IVP via gradient descent. Note how Eq. (1) provides no supervision signal beyond T_{tr}. This is at the core of our approach, as the sequence model G_ϕ will predict OoS weights $\{\theta_t\}_{t=T_{\text{tr}}+1}^{T}$ using the same dynamics that were learned within the training domain.

By learning the dynamics of the weights θ_t through exposure to disjoint but consecutive rings, WeightCaster effectively takes a *global* view of the data and learns a generalisable predictive distribution, rather than the *local* view taken by conventional deep learning. The overall algorithms for training and testing are presented in Algorithm 1.

[2] Note the use of the sum and not the average; this is because some rings might be empty.

[3] Conventional *sequence-to-sequence* models such as Transformers, SSMs, LSTMs, WSL-RNN are equally suitable for this task [Nzoyem, 2025].

Algorithm 1 WeightCaster Training and Inference

Training

Input: Data $X^{\text{tr}}, Y^{\text{tr}}$, anchor $\underline{x}$, distance $d(\cdot,\cdot)$, ring width δ, random ϕ and θ_1, batch size B
Output: Trained ϕ, θ_1

Partition X^{tr} into subsets $\{X_t^{\text{tr}}\}_{t=1}^{T_{\text{tr}}}$ using $d(\underline{x},\cdot)$ and δ

$\{\theta\}_{t=2}^{T} \leftarrow G_\phi(\theta_1)$

while not converged **do**
  $\mathcal{L}_{\text{tot}} \leftarrow 0$
  for $t = 1$ **to** T_{tr} **do**
    $(X, Y) \leftarrow \text{Subsample}(X_t^{\text{tr}}, Y_t^{\text{tr}}, B)$
    $\mathcal{L}_{\text{tot}} \leftarrow \mathcal{L}_{\text{tot}} + \frac{1}{B} \sum_{x,y \in X,Y} \ell(f_{\theta_t}(x), y)$
  end for
  Update ϕ, θ_1 via $\nabla \mathcal{L}_{\text{total}}$
end while

Inference

Input: Test point x, anchor $\underline{x}$, distance $d(\cdot,\cdot)$, ring width δ, trained ϕ and θ_1
Output: Prediction $\hat{y}$

$d_{\text{test}} = d(\underline{x}, x)$

$t_{\text{test}} = \lfloor d_{\text{test}}/\delta \rfloor$

$\{\theta\}_{t=2}^{t_{\text{test}}} \leftarrow G_\phi(\theta_1)$

$\hat{y} = f_{\theta_{t_{\text{test}}}}(x)$

2.3 Stochastic Framework for Regression

To handle uncertainty, we extend the weight-space sequence model to a stochastic framework. Instead of a point estimate θ_t, the G_ϕ outputs the parameters of a distribution over weights. This approach allows us to propagate uncertainty from the weight space to the prediction space.

Reparameterisation trick. We assume the weights at time step t follow a Gaussian distribution $\theta_t \sim q(\theta_t) = \mathcal{N}(\mu_t, \text{diag}(\sigma_t^2))$. To enable backpropagation through the sampling process, we employ the reparameterisation trick [Kingma et al., 2013]

$$\theta_t = \mu_t + \sigma_t \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{2}$$

where μ_t and σ_t are the mean and standard deviation predicted by G_ϕ, and $\odot$ denotes the element-wise product. 0 and I are respectively the zero vector and the intentiy matrix of suitable dimensions.

Marginalisation via linearisation. A primary challenge in OoS generalization is obtaining the predictive distribution $p(y|x)$, which requires marginalising over the weights

$$p(y|x) = \int p(y|x,\theta) q(\theta) \mathrm{d}\theta. \tag{3}$$

Since this integral is analytically intractable for deep neural networks, we perform a first-order Taylor expansion of the model $f_\theta(x)$ around the mean weights μ_t

$$f_\theta(x) \approx f_{\mu_t}(x) + \mathbf{J}(\theta - \mu_t), \tag{4}$$

where $\mathbf{J} = \frac{\partial f_\theta(x)}{\partial \theta}\Big|_{\theta=\boldsymbol{\mu}_t}$ is the Jacobian of the model outputs with respect to the weights. Under this linear approximation, we write the predictive distribution as

$$\hat{y} \sim \mathcal{N}(\mu_y, \Sigma_y) \quad \text{with} \quad \begin{cases} \mu_y = f_{\mu_t}(x), \\ \Sigma_y = \mathbf{J}\text{diag}(\sigma_t^2)\mathbf{J}^\top + \sigma_{\text{noise}}^2 I, \end{cases} \tag{5}$$

where σ_{noise} is a hyperparameter included for numerical stability and to account for underlying noise in the ground truth output measurements. This allows us to obtain not only mean predictions but also a principled covariance Σ_y that reflects model uncertainty.

Loss function regularisation. To prevent the model from producing overconfident predictions in OoS regions, we regularise the loss function in Eq. (1). We introduce a KL divergence term between the predicted distribution and a standard Gaussian prior $p(\hat{y}) = \mathcal{N}(0, I)$. The loss function becomes

$$\mathcal{L}(\phi, \mu_1, \sigma_1) = \sum_{t=1}^{T_{\text{tr}}} \mathop{\mathbb{E}}_{(x,y)\sim(X_t^{\text{tr}}, Y_t^{\text{tr}})} \left[\ell(f_{\mu_t}(x), y)\right] + \beta \cdot D_{\text{KL}}\left(\mathcal{N}(\mu_y, \Sigma_y) | \mathcal{N}(0, I)\right), \tag{6}$$

where μ_1, σ_1 are respectively the learned initial mean and standard deviation weights, and β is a scaling hyperparameter. This loss promotes a graceful reversal toward the prior rather than collapsing as the model moves further from the training support.

3 Main Results

3.1 Experimental Setup

We evaluate WeightCaster on two regression benchmarks designed for OoS generalisation: a synthetic periodic function and a real-world sensor correlation task.

Cosine Dataset. A classic 1D regression problem where $y = \cos(10x) + 0.5x + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 25e-6)$. The data is partitioned into disjoint train and test sets. This task requires the model to extrapolate the trend and periodicity into the unseen intervals. We set $\beta = 1e-2$. The anchor $\underline{x}$ is chosen as the *average* across X^{tr}. We specify $T = 600$ rings to simulate the weight-space sequence, of which $T_{\text{tr}} = 300$ overlap the training data.

AirQuality Dataset. Derived from the UCI Air Quality dataset [Vito, 2008], we model the relationship between two chemical sensors: PT08.S5 (O3) as the input x and PT08.S3 (NOx) as the target y. Once normalised, we split the data based on a threshold of the O_3 sensor readings ($x > 1$), creating a distinct support shift between training and testing distributions. We use $\beta = 5e-2$. The anchor $\underline{x}$ is chosen as the *minimum* across X^{tr}. For this dataset, we set $T = 80$ and $T_{\text{tr}} = 40$.

For both regression tasks, we use a linear regression model $f_\theta(x) = x \cdot \theta^1 + \theta^2$ with 2 scalar learnable parameters. For the sequence model G_ϕ, we considered an autoregressive linear recurrence in weights space $\theta_{t+1} = \phi\theta_t$, with $\phi \in \mathbb{R}^{2\times 2}$ as a learnable matrix. The discrepancy metric $\ell(\cdot, \cdot)$ is the MSE. Training WeightCaster is performed using the Adabelief optimiser [Zhuang et al., 2020] within the JAX ecosystem [Bradbury et al., 2018]. We consider three baselines: a standard MLP [McCulloch and Pitts, 1943], a Gaussian Process [Williams and Rasmussen, 1995], and an Engression model [Shen and Meinshausen, 2025].

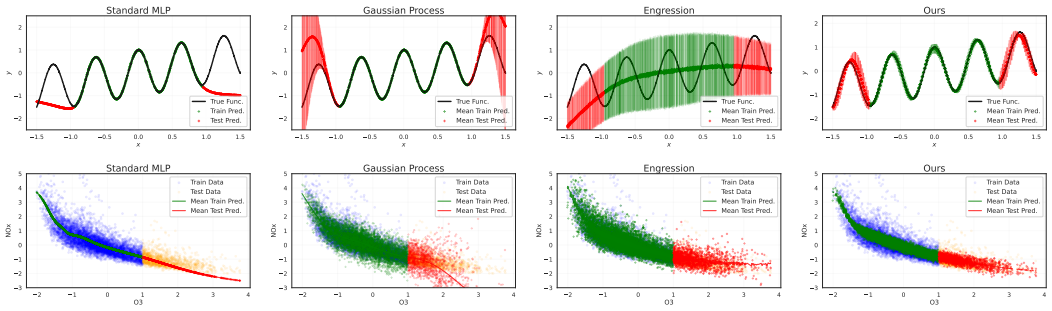

Figure 2: Performance of WeightCaster compared to baselines. (Top Row) Extrapolation on the Cosine wave experiment. (Bottom Row) OoS generalisation on the AirQuality sensor dataset. Shaded areas represent the pointwise 2σ uncertainty estimates.

3.2 Discussion

The results in Fig. 2 and Table 1 demonstrate that WeightCaster effectively sidesteps the catastrophic extrapolation failures common in standard neural networks. By reformulating the problem as weight-space sequence modelling, we allow the model to learn the evolution of the function's parameters.

Table 1: Train (in-distribution) and test (out-of-support) MSEs across datasets (↓).

Method	Cosine (InD)	Cosine (OoS)	AirQuality (InD)	AirQuality (OoS)
Standard MLP	0.00021	2.3672	0.3175	0.2284
Gaussian Process	0.00002	1.3973	0.3203	0.7053
Engression	0.50802	1.3240	0.3240	0.1603
WeightCaster (Ours)	0.00190	0.3502	0.3484	0.1381

The success in the **Cosine** experiment suggests that our sequence model captures the underlying periodicity of the weight trajectory, allowing it to "forecast" the weights of the outermost rings accurately. Interestingly, Engression fails to capture the conditional distribution $p(y|x)$, neither in-distribution nor out-of-distribution.

In the **AirQuality** task, WeightCaster performs on-par with state-of-art techniques like Engression, especially on the testing OoS data. Unlike non-parametric methods like Gaussian Processes that scale poorly with the training data size, WeightCaster maintains the computational efficiency of parametric models while providing competitive uncertainty estimates via linearisation.

Crucially, WeightCaster exhibits such powerful performance at extremely low parameter count $D_{\text{WeightCaster}} = D_\theta + D_\phi = D_\theta + D_\theta^2 = 2 + 4 = 6$.[4] In effect, WeightCaster seeks to fit each θ_t on a fraction of the original dataset, which requires less parameters, thus limiting the size of the regression model θ. This translates to strong computational and memory savings, underscoring a significant benefit absent in other parametric models such as the standard MLP or Engression.

Additionally, the square matrix ϕ in the linear recurrence G_ϕ captures the weight dynamics. An eigendecomposition of this matrix would reveal important characteristics for generalisation to a (theoretical) infinitely wide out-of-support domain. This makes our framework highly interpretable, which is critical when deploying AI models in the real-world.

Conversely, the error bars in Fig. 2 suggest that WeightCaster tends to be just as confident in the training domain as it is out-of-support. While this can be addressed by careful tuning of the scaling hyperparameter β, it displays a critical limitation in that WeightCaster introduces several such hyperparameters (e.g., $d, \delta, \underline{x}, \beta$ as seen in Algorithm 1), all of which require tuning to achieve satisfying results.

4 Conclusion

In this paper, we introduced WeightCaster , a framework that transforms out-of-support generalisation into a weight-space sequence forecasting task. By decomposing the input space into concentric rings and modelling the trajectory of optimal weights, we enable reliable extrapolation with no inductive biases. Our stochastic formulation further provides uncertainty estimates through model linearisation and output-space KL regularization. Experimental results on periodic synthetic data and real-world air quality sensors confirm that WeightCaster outperforms standard regression baselines and competitive generalisation methods. Future work will explore the theoretical underpinnings of this approach in the infinite-length $T \to \infty$ regime, along with the scaling to high-dimensional manifold data. Chosing an appropriate anchor location $\underline{x}$ remains a challenge we wish to solve. We will also explore proven strategies for more confident in-distribution uncertainties while simultaneously requiring conservative out-of-support estimates.

Broader Impact

By enabling generalisation to unseen scenarios, this work could contribute to the mitigation of catastrophic failures in critical sectors such as environmental monitoring, healthcare, and infrastructure management. Our approach encourages a more transparent understanding of a model's operational limits, fostering the responsible deployment of predictive AI. To accelerate usability and foster reproducibility, we provide our code at `https://github.com/ddrous/weightcaster`.

[4] In our recurrent implementation, we use an ANODE-style augmentation [Dupont et al., 2019] for increased representational power, resulting in slightly larger parameters $D_\phi = (D_\theta + a)^2$, with a the augmentation size.

ACKNOWLEDGMENTS

This work was supported by UK Research and Innovation grant EP/S022937/1: Interactive Artificial Intelligence.

REFERENCES

Martin Arjovsky. *Out of distribution generalization in machine learning*. PhD thesis, New York University, 2020.

Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert systems with applications*, 165:113816, 2021.

Alexandre Blanco-Gonzalez, Alfonso Cabezon, Alejandro Seco-Gonzalez, Daniel Conde-Torres, Paula Antelo-Riveiro, Angel Pineiro, and Rebeca Garcia-Fandino. The role of ai in drug discovery: challenges, opportunities, and strategies. *Pharmaceuticals*, 16(6):891, 2023.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

Zahraa Salah Dhaif and Nidhal K El Abbadi. A review of machine learning techniques utilised in self-driving cars. *Iraqi Journal For Computer Science and Mathematics*, 5(1):1, 2024.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *Advances in neural information processing systems*, 32, 2019.

Emilien Dupont, Hyunjik Kim, SM Eslami, Danilo Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. *arXiv preprint arXiv:2201.12204*, 2022.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

Shashank Mohan Jain. Hugging face. In *Introduction to transformers for NLP: With the hugging face library and models to solve problems*, pages 51–67. Springer, 2022.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Nawid Keshtmand, Raul Santos-Rodriguez, and Jonathan Lawry. Counterfactual generation for out-of-distribution data. In *Northern Lights Deep Learning Conference 2026*, 2026.

Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

Daniel Kuhn, Soroosh Shafiee, and Wolfram Wiesemann. Distributionally robust optimization. *Acta Numerica*, 34:579–804, 2025.

Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

Roussel Desmond Nzoyem. Learning to learn sequential dynamics: Context-aware out-of-distribution adaptation for time series and physical systems, 2025.

Roussel Desmond Nzoyem, David A.W. Barton, and Tom Deakin. Neural context flows for meta-learning of dynamical systems. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8vzMLo8LDN.

Roussel Desmond Nzoyem, Nawid Keshtmand, Enrique Crespo Fernandez, Idriss Tsayem, Raul Santos-Rodriguez, David AW Barton, and Tom Deakin. Weight-space linear recurrent neural networks. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=zHdKaF3ZM7.

Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

Konstantin Schürholt, Michael W Mahoney, and Damian Borth. Towards scalable and versatile weight space learning. *arXiv preprint arXiv:2406.09997*, 2024.

Xinwei Shen and Nicolai Meinshausen. Engression: extrapolation through the lens of distributional regression. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 87(3):653–677, 2025.

Thomas Unterthiner, Daniel Keysers, Sylvain Gelly, Olivier Bousquet, and Ilya Tolstikhin. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Saverio Vito. Air Quality. UCI Machine Learning Repository, 2008. DOI: https://doi.org/10.24432/C59K5F.

Christopher Williams and Carl Rasmussen. Gaussian processes for regression. *Advances in neural information processing systems*, 8, 1995.

Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in neural information processing systems*, 33:18795–18806, 2020.