FUNCTIONAL DISTRIBUTION NETWORKS (FDN)

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern probabilistic regressors often remain overconfident under distribution shift. We present *Functional Distribution Networks* (FDN), an input-conditioned distribution over network weights that induces predictive mixtures whose dispersion adapts to the input. FDN is trained with a β-ELBO and Monte Carlo sampling. We further propose an evaluation protocol that cleanly separates interpolation from extrapolation and stresses OOD sanity checks—e.g., that predictive likelihood degrades under shift while in-distribution accuracy and calibration are maintained. On standard regression tasks, we benchmark against strong Bayesian, ensemble, dropout, and hypernetwork baselines under matched parameter and update budgets, and assess accuracy, calibration, and shift-awareness with standard diagnostics. Together, the framework and protocol aim to make OOD-aware, well-calibrated neural regression practical and modular.

1 INTRODUCTION

Modern neural predictors are routinely deployed under dataset shift, where test inputs depart from the training distribution. In these regimes, point predictions from deterministic networks and naïve uncertainty surrogates from traditional stochastic heuristics often become *overconfident*—assigning high probability to wrong outcomes, particularly off-support/extrapolation—undermining reliable decision making. Bayesian Neural Networks (BNNs), MLP with dropout, Deep Ensembles, and Hypernetworks are strong practical baselines, yet they can still under-react outside the training support or require substantial ensembling/sampling to behave robustly (Quiñonero-Candela et al., 2009; Ovadia et al., 2019; Guo et al., 2017; MacKay, 1992; Neal, 1996; Graves, 2011; Blundell et al., 2015; Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017; Ha et al., 2017).

This motivates architectures that are *uncertainty-aware* and *calibrated*—sharp in-distribution (ID) and widening appropriately OOD (higher CRPS, wider intervals). We pursue this with *Functional Distribution Networks (FDN)*, which place input-conditional distributions over weights to modulate uncertainty locally in x. Concretely, FDN amortizes an input-conditional posterior $q_\phi(\theta \mid x)$ via small Hypernetworks and trains it with a Monte Carlo likelihood and a β-ELBO. Under matched budgets, FDN is ID-competitive and well-calibrated by *scale*: on low-frequency/piecewise-smooth shifts (step, quadratic tasks) it achieves near-unity MSE–Var slopes with high Spearman and favorable AURC/CRPS; on oscillatory shifts (sine task) it retains rank but under-scales variance—an explicit target for improvement.

Overview. Rather than treating weights as fixed (or globally random), FDN places an *input-conditioned* distribution over weights:

$$\theta \mid x \sim p(\theta \mid x), \qquad y \mid x, \theta \sim p(y \mid x, \theta). \tag{1}$$

For tractability, we set the conditional prior to be input-agnostic, $p(\theta \mid x) = p_0(\theta) = \mathcal{N}(0, \sigma_0^2 I)$. This keeps the KL term in the ELBO simple and provides a stable regularizer; all input dependence is carried by the amortized posterior $q_\phi(\theta \mid x)$. Many common models can be cast in this form via different choices of $q_\phi(\theta \mid x)$; we detail these instantiations in Appendix A.

Intuitively, sampling θ from an x-dependent posterior lets the function adapt locally; when x lies far from the training support, the conditional weight distribution can *broaden*, yielding wider, appropriately uncertain predictive densities.

Training objective. We train with a Monte Carlo β–ELBO over $q_\phi(\theta \mid x)$, report an IWAE variant, and derive the LP–FDN *layer-wise* KL (averaged over sampled activations); details in Appendix B.

Evaluation protocol. We split test points into *interpolation* (ID) and *extrapolation* (OOD). We summarize shift by deltas $\Delta(\cdot) = \mathbb{E}_{\mathrm{OOD}}[\cdot] - \mathbb{E}_{\mathrm{ID}}[\cdot]$. A well-behaved model should widen uncertainty off-distribution: typically $\Delta\mathrm{Var} > 0$, with higher error and worse distributional score ($\Delta\mathrm{MSE} > 0$, $\Delta\mathrm{CRPS} > 0$). Calibration is assessed via the MSE–variance relation (ideal slope ≈ 1, intercept ≈ 0) and positive rank agreement (Spearman $\rho > 0$).

Contributions.

- **Model:** We define FDN as an input-conditioned distribution over network weights, realized by a lightweight hypernetwork and optimized by the β-*ELBO*. We consider two variants: *Input-Conditioned Functional Distribution Network (IC-FDN)*, which conditions only on the input, and *Layer-Progressive Functional Distribution Network (LP-FDN)*, which conditions each layer on the previous activation and samples sequentially through depth.
- **Protocol:** We propose a simple, reproducible extrapolation protocol that makes the target behavior explicit ($\Delta\mathrm{Var} > 0$) and complements it with calibration diagnostics (MSE–variance slope and rank correlation; Risk-Coverage curves).
- **Fair comparisons:** To avoid confounding factors, we match parameter counts and *update budgets* across methods, ensuring apples-to-apples comparisons with baseline.
- **Empirical findings.** Under matched budgets, FDN delivers competitive ID accuracy and the strongest *scale calibration* on low-frequency or piecewise-smooth shifts (quadratic, step tasks)—near-unity MSE–Var slopes with small intercepts and high rank. On highly oscillatory shift (sine), FDN preserves rank but under-scales variance, increasing AURC.

Scope, significance, and modularity. FDN is simple to implement, compatible with standard training pipelines, and agnostic to the backbone. By conditioning *weight* uncertainty on the input (and, optionally, compact context), it yields uncertainty that is sharp in-distribution yet broadens gracefully off-support. Crucially, FDN is a *layer-level* component: it can replace conventional MLP layers, variational (Bayesian) layers, or Hypernetwork layers with minimal code changes. This allows practitioners to retrofit *predictive uncertainty* into existing architectures without a full redesign: keeping the backbone and training loop, swapping selected blocks for FDN layers, and training under the same objective (plus a KL term).

Position relative to Neural Processes. Unlike Neural Processes (NP), which amortize a predictive distribution directly from a context set, FDN does not access the output head in that way. Instead, it modulates *weight* uncertainty via an input-conditional variational family $q_\phi(\theta \mid x)$ while leaving the predictive head unchanged. The goal is not to replace NP-style predictors, but to plug into and compete with standard layer stacks.

Code availability. All code, configs, and scripts to reproduce the experiments will be released upon acceptance.

2 Related Work

Uncertainty in neural regression. BNN place distributions over weights and infer posteriors via variational approximations or MCMC (MacKay, 1992; Neal, 1996; Graves, 2011; Blundell et al., 2015). Deep Ensembles average predictions from independently trained networks and are a strong practical baseline (Lakshminarayanan et al., 2017; Maddox et al., 2019). MLP (with dropout) interprets dropout at test time as approximate Bayesian inference (Gal & Ghahramani, 2016). Heteroscedastic regression learns input-dependent output variance but retains deterministic weights (Nix & Weigend, 1994; Kendall & Gal, 2017).

Hypernetworks and conditional weight generation. Hypernetworks generate the weights of a primary network using an auxiliary network (Ha et al., 2017); related work explores dynamic, input-conditioned filters and conditional computation (De Brabandere et al., 2016; Brock et al., 2018).

Bayesian/uncertainty-aware Hypernetworks place distributions over generated weights and train them variationally (Krueger et al., 2017). Our FDN differs by conditioning *weight distributions* on inputs to modulate uncertainty itself, not only deterministic weights.

Meta-learning and context-conditioned predictors. Neural Processes (NP) learn a distribution over functions conditioned on a *context set* $\mathcal{C}$, typically via a global latent that captures function-level uncertainty and yields $p(y \mid x, \mathcal{C})$ (Garnelo et al., 2018; Kim et al., 2019). Gradient-based meta-learning (e.g., MAML) instead optimizes an initialization that is adapted per task via inner-loop gradients (Finn et al., 2017). FDN is complementary: it amortizes *weight uncertainty* directly on the input (and optionally a lightweight context) through $q_\phi(\theta \mid x,c)$, requires no inner-loop adaptation or explicit context set, and injects uncertainty at the *layer level* rather than through a single global function latent. Practically, this makes FDN a drop-in module for standard backbones, enabling input-aware epistemic uncertainty without redesigning the predictive head or training pipeline.

Calibration and OOD behavior. Proper scoring rules such as the continuous ranked probability score (CRPS) are strictly proper and reward calibrated predictive distributions (Gneiting & Raftery, 2007). Empirical studies highlight overconfidence under dataset shift and introduce OOD benchmarks (Ovadia et al., 2019). Our evaluation therefore separates interpolation from extrapolation and uses the monotonic relationship between per-sample squared error (MSE) and predicted variance as a simple diagnostic calibration check.

Positioning. Compared to BNNs, FDN sidesteps global posteriors by amortizing *local* weight distributions $q_\phi(\theta \mid x)$. Compared to Deep Ensembles, FDN uses shared parameters and stochastic generation instead of replicating full models. Compared to Hypernetworks, FDN explicitly models *uncertainty over* generated weights and regularizes it with a KL prior, enabling principled OOD expansion.

3 METHOD

3.1 PRELIMINARIES

Let $f_\theta : \mathbb{R}^{d_x} \to \mathbb{R}^{D_y}$ denote a network that outputs the parameters of a d_y-dimensional Gaussian predictive head. Here d_x and d_y are the input and output dimensions, and $D_y = \frac{d_y(d_y+3)}{2}$ is the number of free parameters for a *full-covariance* d_y-Gaussian (mean plus Cholesky-coded covariance). We write the observation model as

$$p(y \mid x, \theta) = \mathcal{N}\big(y;\, \mu_\theta(x),\, \Sigma_\theta(x)\big), \qquad \text{where} \quad f_\theta(x) := \big(\mu_\theta(x),\, \Sigma_\theta(x)\big).$$

Homoscedastic means the noise covariance is constant across inputs ($\Sigma_\theta(x) \equiv \Sigma$); *heteroscedastic* means it varies with x, θ ($\Sigma_\theta(x)$). This is orthogonal to the *correlation structure*:

- **Isotropic (uncorrelated, equal variance):** $\Sigma = \sigma^2 I$ (homoscedastic) or $\Sigma_\theta(x) = \sigma_\theta^2(x) I$ (heteroscedastic).
- **Diagonal (uncorrelated, per-dimension variance):** $\Sigma = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_{d_y}^2)$, or $\Sigma_\theta(x) = \mathrm{diag}\big(\exp s_\theta(x)\big)$ with $s_\theta(x) = \log \sigma_\theta^2(x)$.
- **Full (correlated):** any symmetric positive-definite covariance admits a Cholesky factorization, homoscedastic or heteroscedastic:

$$\Sigma = LL^\top \quad \text{or} \quad \Sigma_\theta(x) = L_\theta(x) L_\theta(x)^\top, \;\; \mathrm{diag}\, L_{(\cdot)} > 0.$$

The per-example β-ELBO then becomes

$$\mathcal{L}^{(i)}_{\text{Gauss}} = \frac{1}{2K} \sum_{k=1}^{K} \Big[\big\| y_i - \mu_{\theta_k}(x_i) \big\|^2_{\left(\Sigma_{\theta_k}(x_i)\right)^{-1}} + \log\det\big(2\pi\, \Sigma_{\theta_k}(x_i)\big) \Big] + \beta\, D_{\mathrm{KL}}(q_\phi(\theta \mid x_i) \| p_0(\theta)),$$

where $\|v\|_A^2 := v^\top A v$. In the *homoscedastic* case the covariance is constant across inputs, so a full Σ encodes a single, global correlation structure among output dimensions; if Σ is diagonal, the data term reduces to a (constant-)weighted least-squares plus a constant log-determinant. In

contrast, *heteroscedastic* models use an input-dependent $\Sigma_\theta(x)$, so the weights (and, for full $\Sigma_\theta(x)$, correlations) vary with x, θ.

For $d_y = 1$ the covariance reduces to a scalar $\sigma_\theta^2(x)$. *Homoscedastic in* x means $\sigma_\theta^2(x) \equiv \sigma_\theta^2$ (constant across inputs for a fixed θ), whereas *heteroscedastic in* x means $\sigma_\theta^2(x)$ varies with x. Orthogonally, because we draw stochastic weights $\theta^{(k)}$, one can distinguish dependence on the sampled weights: *homoscedastic in* θ means $\sigma_\theta^2(x)$ is effectively deterministic (identical across $\theta^{(k)}$ for a given x), while *heteroscedastic in* θ means $\sigma_{\theta^{(k)}}^2(x)$ changes with the sampled weights (as in FDN/BNN where the variance head depends on $\theta^{(k)}$).

In the **isotropic heteroscedastic** case ($\Sigma_{\theta^{(k)}}(x) = \sigma_{\theta^{(k)}}^2(x) I$) the Gaussian NLL/NLPD contribution is

$$\frac{1}{2K} \sum_{k=1}^{K} \left[\left(\frac{y_i - \mu_{\theta^{(k)}}(x_i)}{\sigma_{\theta^{(k)}}(x_i)} \right)^2 + \log\big(2\pi\, \sigma_{\theta^{(k)}}^2(x_i)\big) \right].$$

Thus, the data term is a *per-input, per-sample weighted* MSE plus a variance penalty. If one assumes **isotropic homoscedastic** noise (σ^2 constant), the data term is proportional to MSE up to an additive constant; in practice the fit–regularization trade-off can be tuned either by setting σ^2 or, equivalently, by adjusting β to re-balance the data term against the KL. Since our focus is on uncertainty-aware metrics rather than cross-output correlations, we restrict attention to $d_y = 1$ and adopt the isotropic homoscedastic case (a single, constant variance σ^2 which we will absorb into β). A more general treatment—including heteroscedastic and full-covariance models, as well as structured priors—is left to future work.

Why homoscedastic noise? We fix the observation variance to a single scalar σ^2 (set to 1 in our runs) for three reasons: (i) *isolation of epistemic effects*: our focus is on input-conditional weight uncertainty $q_\phi(\theta \mid x)$; a fixed σ^2 avoids confounding with an input-dependent noise head; (ii) *fairness and simplicity*: all baselines use the same likelihood, so differences reflect how each method models weight uncertainty rather than aleatoric parameterization; (iii) *capacity control equivalence*: with fixed σ^2, the data term is a rescaled MSE and changing σ^2 simply rescales the KL weight (effective β).

We refer the reader to Appendix C for the heteroscedastic variant and the predictive-variance decomposition.

3.2 FDN: Input-Conditioned Weight Distributions

We drop explicit context and condition only on signals from the network itself. For each layer ℓ, FDN places a diagonal-Gaussian over its weights whose parameters are produced by a small Hypernetwork $A_\ell(\cdot)$. We choose the conditioning signal

$$s_\ell^{(k)} \in \begin{cases} x, & \text{IC-FDN} \\ a_{\ell-1}^{(k)}, & \text{LP-FDN (with } a_0^{(k)} = x), \end{cases}$$

and set

$$(\mu_{W,\ell},\, \rho_{W,\ell},\, \mu_{b,\ell},\, \rho_{b,\ell}) \;=\; A_\ell(s_\ell^{(k)}), \qquad \sigma_{W,\ell} \;=\; \varepsilon + \mathrm{softplus}(\rho_{W,\ell}), \quad \sigma_{b,\ell} \;=\; \varepsilon + \mathrm{softplus}(\rho_{b,\ell}),$$

with a small floor $\varepsilon = 10^{-3}$ for numerical stability. Sampling then proceeds as

$$W_\ell^{(k)} = \mu_{W,\ell} + \sigma_{W,\ell} \odot z_{W,\ell}^{(k)}, \qquad b_\ell^{(k)} = \mu_{b,\ell} + \sigma_{b,\ell} \odot z_{b,\ell}^{(k)}, \qquad z_{\{\cdot\},\ell}^{(k)} \sim \mathcal{N}(0, I),$$

and, for LP-FDN, $s_\ell = a_{\ell-1}^{(k)}$ with $a_\ell^{(k)} = f_\ell(a_{\ell-1}^{(k)}; W_\ell^{(k)}, b_\ell^{(k)})$ (sequential across layers).

Variational family (compact). FDN uses a layer-wise diagonal-Gaussian over weights, conditioned on $s_\ell \in \{x,\ a_{\ell-1}^{(k)}\}$:

$$q_\phi(\theta \mid x) = \prod_{\ell=1}^{L} \mathcal{N}\big(\mathrm{vec}(W_\ell); \mu_{W,\ell}(s_\ell),\, \mathrm{diag}\, \sigma_{W,\ell}^2(s_\ell)\big)\, \mathcal{N}\big(b_\ell; \mu_{b,\ell}(s_\ell),\, \mathrm{diag}\, \sigma_{b,\ell}^2(s_\ell)\big),$$

Algorithm 1 FDN (Unified for IC-/LP-FDN): Training and Prediction

Inputs: dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$; base net f_θ with layers $1{:}L$; per-layer samplers $q_\phi^l(\theta_l \mid c_l)$ (diag. Gaussians); prior $p_0(\theta) = \prod_l p_0^l(\theta_l)$; MC K; KL schedule $\{\beta_t\}$; variant $v \in \{\text{IC}, \text{LP}\}$.
for step $t = 1, 2, \ldots$ **do**
  Sample minibatch $\mathcal{B}$; set $\Sigma_{\text{NLL}} \leftarrow 0$, $\Sigma_{\text{KL}} \leftarrow 0$
  for each $(x, y) \in \mathcal{B}$ **do**
    for $k = 1, \ldots, K$ **do**
      $h_0^{(k)} \leftarrow x$
      for $l = 1, \ldots, L$ **do**
        $c_l \leftarrow \begin{cases} x & \text{if } v = \text{IC} \\ h_{l-1}^{(k)} & \text{if } v = \text{LP} \end{cases}$
        $\varepsilon_l^{(k)} \sim \mathcal{N}(0, I), \quad \theta_l^{(k)} \leftarrow \mu_\phi^l(c_l) + \sigma_\phi^l(c_l) \odot \varepsilon_l^{(k)}$
        $h_l^{(k)} \leftarrow \text{layer}_l\big(h_{l-1}^{(k)}; \theta_l^{(k)}\big)$
        $\Sigma_{\text{KL}} \mathrel{+}= D_{\text{KL}}\big(q_\phi^l(\theta_l \mid c_l) \,\|\, p_0^l(\theta_l)\big)$
      end for
      $(\mu^{(k)}, \Sigma^{(k)}) \leftarrow \text{head}\big(h_L^{(k)}\big)$ {$\Sigma^{(k)}$ may be fixed (homoscedastic)}
      $\ell_k \leftarrow \log \mathcal{N}\big(y; \mu^{(k)}, \Sigma^{(k)}\big)$
    end for
    $\Sigma_{\text{NLL}} \mathrel{+}= -\frac{1}{K} \sum_{k=1}^{K} \ell_k$ {ELBO (mean-of-logs)}
  end for
  $\mathcal{L} \leftarrow \frac{1}{N}\Big(\Sigma_{\text{NLL}} + \beta_t \, \Sigma_{\text{KL}}\Big)$; update ϕ by backprop on $\mathcal{L}$
end for
Predict at $x_\star$**:** repeat the per-layer sampling with $c_l = \{x_\star \text{ or } h_{l-1}^{(k)}\}$ per v to obtain $(\mu_\star^{(k)}, \Sigma_\star^{(k)})$, and return $\hat{p}(y \mid x_\star) \approx \frac{1}{K} \sum_k \mathcal{N}\big(y; \mu_\star^{(k)}, \Sigma_\star^{(k)}\big)$.

with Hypernetwork outputs $(\mu_{W,\ell}, \rho_{W,\ell}, \mu_{b,\ell}, \rho_{b,\ell}) = A_\ell(s_\ell)$ and

$$\sigma_{W,\ell} = 10^{-3} + \text{softplus}(\rho_{W,\ell}), \qquad \sigma_{b,\ell} = 10^{-3} + \text{softplus}(\rho_{b,\ell}).$$

In compact form (concatenating all layers),

$$q_\phi(\theta \mid x) = \mathcal{N}\big(\theta;\, \mu_\phi(x),\, \text{diag}\, \sigma_\phi^2(x)\big), \qquad \theta^{(k)} = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon^{(k)}, \;\; \varepsilon^{(k)} \sim \mathcal{N}(0, I),$$

and the predictive density is the Monte Carlo mixture

$$p(y \mid x) \;\approx\; \frac{1}{K} \sum_{k=1}^{K} p\big(y \mid x, \theta^{(k)}\big), \qquad \theta^{(k)} \sim q_\phi(\theta \mid x).$$

Prior and regularization. We regularize $q_\phi(\theta \,|\, x)$ toward a simple reference $p_0(\theta) = \prod_\ell \mathcal{N}(0, \sigma_0^2 I)$ via a β-weighted KL term (we use $\sigma_0 = 1$ in all experiments). For diagonal Gaussians,

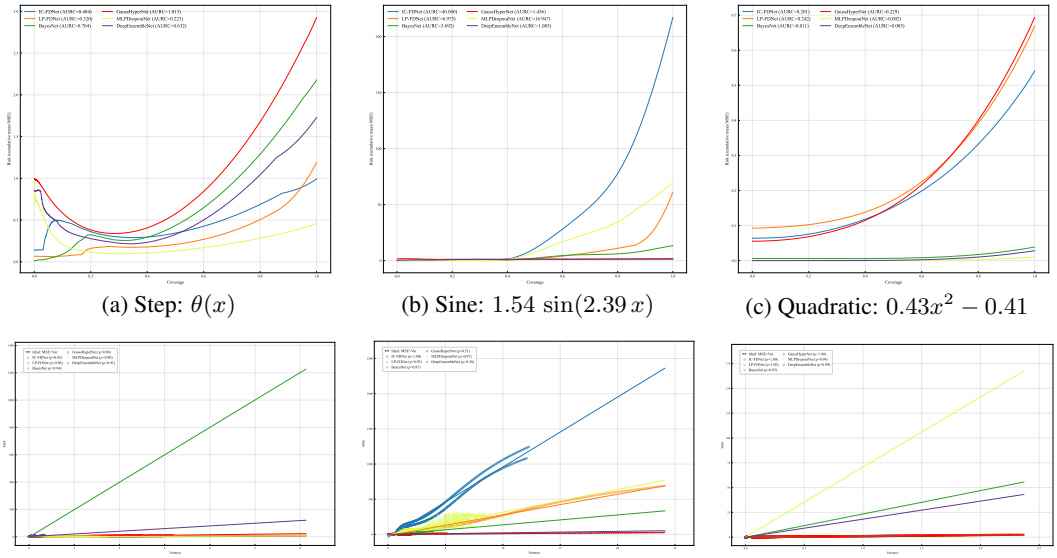
$$D_{\text{KL}}\big(\mathcal{N}(\mu, \text{diag}\, \sigma^2) \,\big\|\, \mathcal{N}(0, \sigma_0^2 I)\big) = \tfrac{1}{2} \sum_j \left(\frac{\sigma_j^2 + \mu_j^2}{\sigma_0^2} \;-\; 1 \;-\; \log \frac{\sigma_j^2}{\sigma_0^2} \right),$$

and we sum this over all layers (for both W_ℓ and b_ℓ). The variance floor is implemented by the ε in $\sigma = \varepsilon + \text{softplus}(\rho)$ rather than a hard bound.

Algorithm 1 summarizes training (β-ELBO with re-parameterized gradients) and inference (Monte-Carlo mixtures over weight draws) for both IC-FDN and LP-FDN.

4 Experiments

Tasks and splits. We evaluate FDN on synthetic 1D meta–regression tasks and standard small/medium regression benchmarks. For the 1D analysis, let the ambient domain be $\mathcal{R} = (-L, L)$



(a) Step: $\theta(x)$ (b) Sine: $1.54\ \sin(2.39\,x)$ (c) Quadratic: $0.43x^2 - 0.41$

(d) Step: $\theta(x)$ (e) Sine: $1.54\ \sin(2.39\,x)$ (f) Quadratic: $0.43x^2 - 0.41$

Figure 1: Suite of 1D regression evaluations across three functions: (a–c) Risk–Coverage (AURC) curves (lower is better) for the step $\theta(x)$, sine $1.54\ \sin(2.39\,x)$, and quadratic $0.43x^2 - 0.41$; (d–f) corresponding MSE vs. predicted variance scatter plots (dashed line shows the ideal $\mathrm{MSE} = \mathrm{Var}$). Colors denote models; legends within each panel report per-model statistics (e.g., AURC or Spearman's ρ).

and define the *interpolation* region $\mathcal{R}_{\text{interp}} = [-l, l] \subset \mathcal{R}$. Training inputs satisfy $x_i \in \mathcal{R}_{\text{interp}}$ with $y_i = f(x_i)$. At test time we report metrics separately on the ID split $\mathcal{R}_{\text{interp}}$ and the OOD split $\mathcal{R}_{\text{extrap}} = \mathcal{R} \setminus \mathcal{R}_{\text{interp}}$, and we also include the aggregation of both. This protocol cleanly isolates ID interpolation from OOD extrapolation.

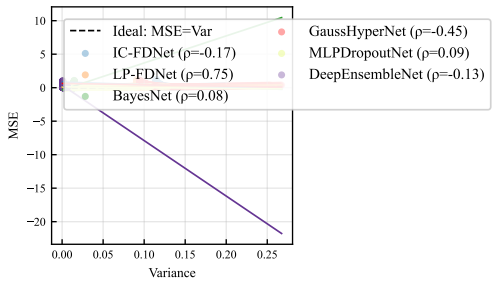
4.1 Complexity, Capacity, and Fairness

Baselines. We evaluate four stochastic baselines: MLP with dropout (**MLPDropoutNet**), Deep Ensemble of MLP (**DeepEnsemblesNet**), Variational BNN (**BayesNet**), and Gaussian Hypernetwork (**GaussianHyperNet**). Because our study centers on calibrated *predictive distributions* (CRPS, MSE–Var slope/intercept, AURC), we omit the **Deterministic MLP** and the (input-conditioned) **Hypernetwork** from the main uncertainty analysis; their ID/OOD MSE is comparable to Ensembles/Dropout. Training details (optimizer, batch size, learning rate, prior scale σ_0, variance floor $\sigma_{\min}$) appear in Table 1 (Appendix D).

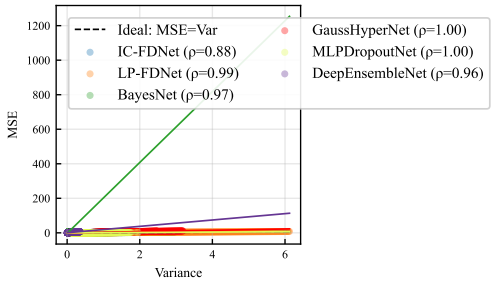
Link-budget. We consider networks with a single hidden layer and fix the parameter budget to $P \approx 1000$ (±5%) for all models, counting *all* trainable parameters, including any Hypernetwork components; counts appear in Table 2 (Appendix D). To equalize the *update* budget, ensembles with M members use *epoch-split* training (epochs divided by M). For non-ensemble networks we use one Monte Carlo draw per update ($K = 1$), keeping per-step cost comparable and the total number of parameter updates matched across models.

Note: To hit the P target, MLPDropoutNet uses widened hidden layers, increasing capacity/expressive power and potentially improving ID MSE independent of uncertainty quality; hence our emphasis on calibration-centric metrics.

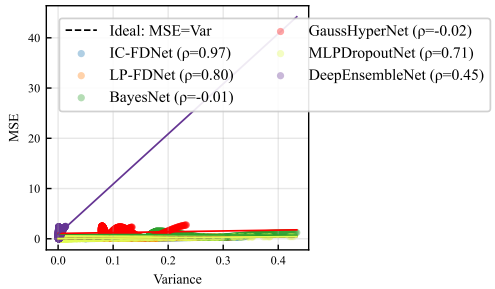
Metrics, diagnostics, and reporting. We evaluate five axes with compact visuals: **Accuracy** (MSE↓); **Uncertainty quality** (predicted $\mathrm{Var}[\hat{y} \mid x]$—small ID, increasing in OOD); **Calibration** via the MSE–variance relation—(i) *rank* agreement, Spearman $\rho(\mathrm{Var}, \mathrm{MSE})$, and (ii) *scale* fit $\mathrm{MSE} \approx a + b\,\mathrm{Var}$ with ideal $a \approx 0$ (error offset at near-zero variance) and $b \approx 1$ (maps uncertainty to error; $b > 1$ = variances too small/overconfident; $b < 1$ = too large/underconfident); **Robustness**



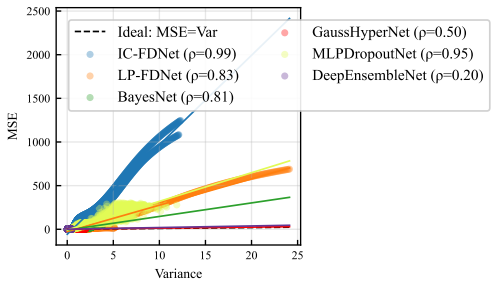
(a) Step (ID): $\theta(x)$



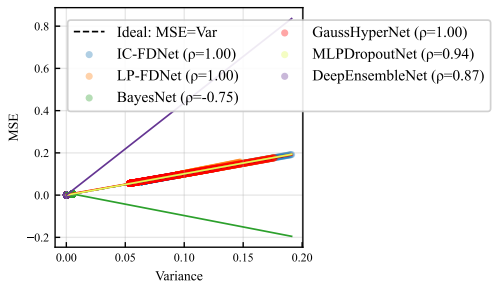
(b) Step (OOD): $\theta(x)$



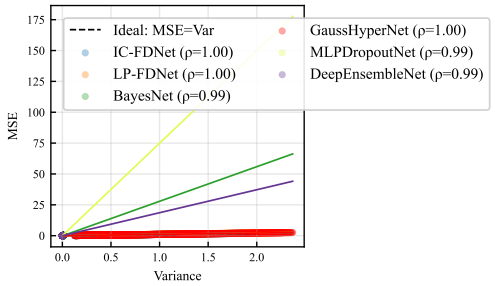
(c) Sine (ID): $1.54\ \sin(2.39\,x)$



(d) Sine (OOD): $1.54\ \sin(2.39\,x)$



(e) Quadratic (ID): $0.43x^2 - 0.41$



(f) Quadratic (OOD): $0.43x^2 - 0.41$

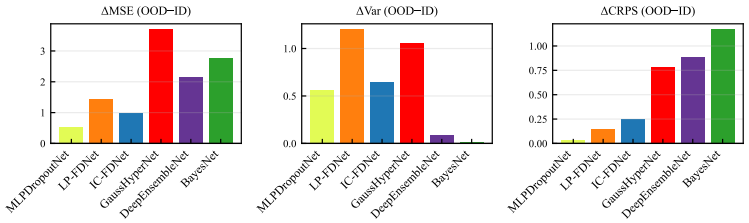
Figure 2: MSE vs. predicted variance on three 1D regression tasks in a 3×2 grid: rows correspond to tasks $\theta(x)$, $1.54\ \sin(2.39\,x)$, and $0.43x^2 - 0.41$; left column shows test interpolation (ID), right column shows test extrapolation (OOD). Colors denote models; legend entries include Spearman's ρ between variance and MSE. The dashed line marks the ideal relation $\mathrm{MSE} = \mathrm{Var}$.

(CRPS↓, strictly proper, full-predictive); and **Selective risk** (AURC↓ from variance-ranked abstention, capturing decision usefulness). We plot MSE–Var scatter with an $\mathrm{MSE} = \mathrm{Var}$ guide and risk–coverage curves across tasks (Fig. 1); ID/OOD MSE–Var scatter (Fig. 2); and bar charts of ID→OOD deltas (e.g., $\Delta\mathrm{MSE} = \mathrm{MSE}_{\mathrm{OOD}} - \mathrm{MSE}_{\mathrm{ID}}$) (Fig. 3).

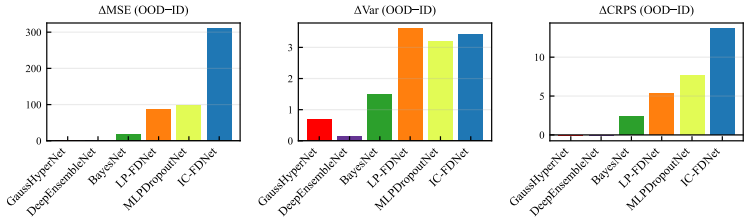
4.2 Results

Discussion. Across the three tasks (Table 3, 4, 5 in Appendix D), FDNet's[1] core strength is *calibration by scale*: on **step** and **quadratic**, both IC–FDNet and LP–FDNet achieve MSE–Var slopes close to the ideal ($b \approx 1$; e.g., $b \in [1.04, 1.25]$) with small intercepts, while maintaining strong rank agreement (Spearman $\rho \geq 0.85$, often $= 1.0$ on quadratic). In these regimes, FDNet's predictive variance increases in lock–step with difficulty (ΔVar large) and its selective abstention is competitive (AURC low–moderate). Importantly, classical baselines that fit ID sharply (e.g., Dropout, Ensem-

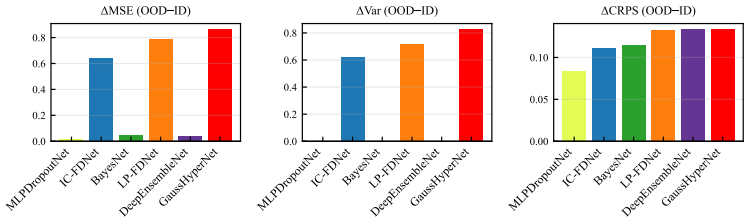
[1] *Terminology. FDN* denotes the framework. We use *FDNet* for the specific architectures considered in the experiments.



(a) Step: $\theta(x)$



(b) Sine: $1.54\ \sin(2.39\,x)$



(c) Quadratic: $0.43x^2 - 0.41$

Figure 3: ΔMSE, ΔVar, and $\Delta CRPS$ for three 1D regression tasks: (a) step $\theta(x)$, (b) sine $1.54\ \sin(2.39\,x)$, and (c) quadratic $0.43x^2 - 0.41$.

bles) show *mis-scaled* uncertainties on these tasks (very large b on quadratic and step), indicating that even when they rank hard points reasonably well, their variance does not track the magnitude of the error.

On the **sine** task—our most oscillatory OOD setting—the picture flips. All methods suffer, but FDNet's *magnitude calibration* deteriorates (very large b with negative a), despite excellent ranking ($\rho \geq 0.93$). FDNet (and MLPDropoutNet) do raise variance substantially OOD ($\Delta\text{Var} \approx 3$), yet the error explodes faster than the variance grows (huge ΔMSE), yielding poor AURC. In contrast, DeepEnsembleNet and Gaussian HyperNet see far smaller error increases (much lower ΔMSE), which mechanically keeps their AURC low even though their ranking can be weak (e.g., $\rho = 0.28$ for Ensembles). The takeaway is that *AURC blends both calibration and absolute error growth*: excellent rank calibration cannot compensate for large OOD error if the scale is under-estimated.

Calibration and ranking across splits (ID vs OOD). On ID, all methods cluster near the origin (low error and variance), so Spearman is low-information and calibration is judged mostly by small intercepts. OOD reveals the separation: on **step** and **quadratic**, IC/LP–FDN maintain strong *ranking* and sit close to the $\text{MSE} = \text{Var}$ diagonal (near-unity slopes, small intercepts), i.e., variance scales with error; several baselines lie visibly above the guide (steep fits), signaling overconfidence in magnitude. On **sine**, FDNet still ranks difficult points well (high Spearman) but under-scales uncertainty (steep fitted lines), so errors grow faster than variance. These patterns are evident in the MSE–Var scatters (Fig. 2) and consistent with the ID→OOD deltas (Fig. 3): FDN shows large ΔVar on step/quadratic (tracking difficulty), whereas on sine the surge in ΔMSE dwarfs ΔVar. In short, on smooth shifts (step, quadratic) *multiple* methods rank well but FDN is best on *scale*; on oscillatory OOD (sine task), FDN uniquely preserves high rank while most baselines ranking degrades, though FDN's variance still under-scales the error.

Where FDNet is strong. (i) *Scale calibration on smooth shifts*: near-ideal slopes on step/quadratic show that FDNet's input-conditioned weight uncertainty translates into variances that *numerically* match error growth. (ii) *Ranking*: consistently high ρ means FDNet reliably flags risky points. (iii) *Sensitivity to shift*: large ΔVar indicates FDN reacts to OOD, avoiding the flat-variance pathology seen in some baselines.

Where FDNet is weak. (i) *Highly oscillatory OOD*: on sine, variance grows but not *enough* relative to error (large b, poor AURC), pointing to an *under-scaled* uncertainty head under rapid frequency mismatch. (ii) *CRPS under severe shift*: large ΔCRPS mirrors the magnitude miscalibration.

Advantages and implications. FDNet's principal advantage is calibrated scaling when the OOD distortion is *smooth/monotone* (step/quadratic), where many standard methods remain overconfident (large b). Its high ρ across tasks suggests FDNet is a strong *triage signal* even when the absolute scale lags (sine). Practically, this favors FDNet in settings where errors grow gradually outside the training manifold (safety margins, defer policies), while highlighting a clear avenue for improvement on oscillatory shifts: stronger variance scaling (e.g., temperature on σ_ϕ, variance-floor tuning), richer priors, or layer-wise β schedules. Overall, under matched capacity and update budgets, FDN delivers robust rank calibration and, on two of three tasks, near-ideal error–uncertainty scaling—precisely the behaviors a stochastic regressor should exhibit under controlled distribution shift.

High-level observations and takeaways. Under matched budgets, ID MLPDropoutNet and DeepEnsembleNet typically attain the lowest MSE; FDNet is slightly less sharp but already shows strong rank calibration (high Spearman). OOD separates methods: on **step** and **quadratic**, IC/LP-FDNet most consistently grow variance in proportion to error—near-unity MSE–Var slopes with small intercepts and high Spearman, yielding large $\Delta\mathrm{Var}$, smaller $\Delta\mathrm{CRPS}$, and competitive AURC, while several baselines remain overconfident (steep fits). On the oscillatory **sine** task, FDNet preserves ranking but *under-scales* variance relative to error (steep slopes, higher AURC), a focused improvement area. For risk-aware use cases that depend on "knowing when you don't know," FDNet's uncertainty that generalizes under shift is often preferable to marginal gains in ID sharpness.

5 Limitations

FDN's input-conditioned *weight* stochasticity, like similar stochastic layers, can overfit spurious cues if β is too small or the prior is too loose; careful KL scheduling and priors are important. LP-FDN samples weights layer-by-layer, adding latency vs. a deterministic pass; sampling at test time also incurs a compute/latency trade-off with K. Our study focuses on regression: extending to classification requires discrete predictive mixtures and calibration beyond CRPS (e.g., ECE/Brier). Finally, while our Hypernetworks are lightweight, scaling to very deep backbones may benefit from structured or low-rank generators and variance-temperature/floor controls, particularly to address under-scaling on oscillatory OOD.

6 Conclusion

We introduced Functional Distribution Networks (FDN), which amortize input-conditioned *distributions* over weights to produce predictive densities that remain sharp in-distribution yet expand appropriately under shift. Trained with a Monte Carlo objective and a β-weighted KL to a simple prior, FDN delivers strong *rank* calibration across tasks and near-ideal *scale* calibration on smooth/piecewise-smooth shifts (step, quadratic tasks), as evidenced by Spearman, MSE–Var slope/intercept, $\Delta\mathrm{Var}$, CRPS, and AURC. Under a fair protocol that matches parameters, updates, and predictive-sample budgets, FDN provides uncertainty that is practically useful for abstention and risk-aware inference. Future work includes stronger variance scaling (temperature/floors, layer-wise β scheduling), structured/priors for deep backbones, frequency-aware conditioning to handle oscillatory OOD (sine task), and calibrated extensions to classification.

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathsf{\mathbf{A}}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	The parents of x_i in $\mathcal{G}$

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{\mathbf{A}}_{i,j,k}$	Element (i, j, k) of a 3-D tensor $\mathsf{\mathbf{A}}$
$\mathsf{\mathbf{A}}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor containing derivatives of y with respect to $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E} f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\frac{1}{1 + \exp(-x)}$
$\zeta(x)$	Softplus, $\log(1 + \exp(x))$
$\Vert \boldsymbol{x} \Vert_p$	L^p norm of $\boldsymbol{x}$
$\Vert \boldsymbol{x} \Vert$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$
$\mathbf{1}_{\mathrm{condition}}$	is 1 if the condition is true, 0 otherwise

REFERENCES

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations (ICLR)*, 2017.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *International Conference on Machine Learning (ICML)*, 2015.

Andrew Brock, Theodore Lim, J. M. Ritchie, and Nick Weston. Smash: One-shot model architecture search through hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/forum?id=rydeCEhs-.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations (ICLR)*, 2016.

Bert De Brabandere, Xu Jia, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 667–675, 2016.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks via pac-bayes. In *International Conference on Learning Representations (ICLR)*, 2017.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017. URL https://proceedings.mlr.press/v70/finn17a.html.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1050–1059. PMLR, 2016.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018. URL https://arxiv.org/abs/1807.01622.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.

Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

David Ha, Andrew Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=rkpACe1lx.

Irina Higgins et al. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations (ICLR)*, 2017.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, 2017.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, S. M. Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=SkE6PjC9KX.

David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian hypernetworks. *arXiv preprint arXiv:1710.04759*, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

Wesley Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, 2019.

Radford M. Neal. *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer, 1996.

David A. Nix and Andreas S. Weigend. Estimating the mean and variance of the target probability distribution. In *IEEE International Conference on Neural Networks*, 1994.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Joaquin Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence (eds.). *Dataset Shift in Machine Learning*. MIT Press, 2009.

A UNIFIED VIEW VIA $q_\phi(\theta \mid x)$

All methods we consider can be written as

$$p(y \mid x) \;=\; \int p(y \mid x, \theta)\; q_\phi(\theta \mid x)\, d\theta \;\approx\; \frac{1}{K} \sum_{k=1}^{K} p\big(y \mid x, \theta^{(k)}\big), \qquad \theta^{(k)} \sim q_\phi(\theta \mid x).$$

In this paper, architectural layers are specified by the choice of $q_\phi(\theta \mid x)$. Framing models through $q_\phi(\theta \mid x)$ enables apples-to-apples comparisons: (i) how they set the spread of plausible weights, (ii) whether that spread adapts to the input, and (iii) how much compute they expend to form the predictive mixture. FDN's module-level approach directly targets this knob: it provides local, input-aware uncertainty where it is inserted (e.g., the head or later blocks), broadens off-support as inputs drift from the training domain, and leaves the surrounding backbone and training loop unchanged.

FDN (IC/LP) *FDN* makes q_ϕ **input-conditional and stochastic**. A common choice is diagonal-Gaussian, factorized by layer:

$$q_\phi(\theta \mid x) = \prod_{\ell} \mathcal{N}\big(\mu_\ell(x), \operatorname{diag} \sigma_\ell^2(x)\big).$$

This **input-conditional** variant is *IC-FDN*. For **layer-propagated** conditioning (*LP-FDN*), the ℓ-th layer's weight distribution depends *only* on the previous activation:

$$q_\phi(\theta_\ell \mid x) = \mathcal{N}\big(\mu_\ell(a_{\ell-1}), \operatorname{diag} \sigma_\ell^2(a_{\ell-1})\big), \qquad a_0 := x,$$

and sampling proceeds sequentially across layers along the same Monte Carlo sample path. This induces a first-order Markov structure in depth, allowing uncertainty to expand as signals propagate—later layers can broaden even when early layers remain sharp. We regularize with a per-layer KL:

$$\beta \sum_{\ell=1}^{L} D_{\mathrm{KL}}(q_\phi(\theta_\ell \mid x) \| p_0(\theta_\ell)).$$

More generally, one could condition longer histories $a_{0:\ell-1}$; in this paper, we restrict to first-order (one-step) conditioning. Note, in the limit $\sigma_\ell \to 0$ for all ℓ, the model collapses to a deterministic layer-conditioned Hypernetwork.

Deterministic Hypernetwork. A deterministic Hypernetwork G_ϕ maps the input to weights, yielding a **degenerate** q:

$$q_\phi(\theta \mid x) = \delta\big(\theta - G_\phi(x)\big), \qquad p(y \mid x) = p\big(y \mid x, G_\phi(x)\big).$$

Training typically uses NLL or MSE; weight decay on ϕ can be interpreted as a MAP prior on the Hypernetwork parameters. Because q_ϕ is a Dirac-Delta, there is no weight-space uncertainty: any predictive uncertainty must come from the observation model (e.g., a heteroscedastic head) or post-hoc calibration. Compared to stochastic variants, this adds no KL term and no MC averaging, but can increase per-example compute due to generating weights via G_ϕ.

Gaussian HyperNetwork A *Stochastic* Hypernetwork outputs a **global** posterior (or context-only):

$$q_\phi(\theta \mid x) \equiv q_\phi(\theta \mid h) = \prod_{\ell} \mathcal{N}(\mu_\ell(h), \operatorname{diag} \sigma_\ell^2(h)),$$

i.e., independent of the query x (but dependent on a learnable latent task vector h). This is variational BNN with parameters produced by a Hypernetwork.

Bayesian Neural Network (Bayes-by-Backprop). A standard variational BNN uses an x-**independent** approximate posterior:

$$q_\phi(\theta \mid x) \equiv q_\phi(\theta) = \prod_{\ell} \mathcal{N}\big(\mu_\ell, \operatorname{diag} \sigma_\ell^2\big),$$

and the same β-ELBO objective with closed-form diagonal-Gaussian KL. Because $q_\phi(\theta|x)$ is global, predictive uncertainty does not adapt to x except via the likelihood term, which can under-react off-support compared to input-conditional alternatives. On the other hand, the objective is simple and sampling cost is amortized across inputs, though matching ensemble-like diversity typically requires larger posterior variances or multiple posterior samples at test time.

MLP with Dropout. MLP with dropout induces a distribution over effective weights via random masks m:

$$q_\phi(\theta \mid x) \equiv q_\phi(\theta) \quad \text{(implicit via dropout masks, independent of } x\text{)},$$

and inference averages predictions over sampled masks (Gal & Ghahramani, 2016).

Deep Ensembles. An M-member ensemble of MLPs corresponds to a **finite mixture of deltas**:

$$q(\theta \mid x) \;\equiv\; \frac{1}{M}\sum_{m=1}^{M} \delta\big(\theta - \theta_m\big), \qquad p(y \mid x) \;=\; \frac{1}{M}\sum_{m=1}^{M} p\big(y \mid x, \theta_m\big),$$

where each θ_m is trained independently from a different initialization (and typically a different data order/augmentation). There is no explicit KL regularizer; diversity arises implicitly from independent training trajectories. Inference cost scales linearly with M (one forward pass per member), and for fair comparisons we match total update or compute budgets by reducing epochs.

B Training objective

FDN is amortized variational inference with the latent weights θ and input-conditional posterior $q_\phi(\theta \mid x)$ realized by small hypernetworks via the reparameterization $\theta^{(k)} = g_\phi(x, \varepsilon^{(k)})$ with $\varepsilon^{(k)} \sim \mathcal{N}(0, I)$. For a single (x, y) the ELBO is

$$\log p(y \mid x) \;\geq\; \underbrace{\mathbb{E}_{q_\phi(\theta|x)}\big[\log p(y \mid x, \theta)\big]}_{\text{data term}} \;-\; \underbrace{D_{\mathrm{KL}}\big(q_\phi(\theta \mid x)\,\|\,p_0(\theta)\big)}_{\text{regularizer}}, \tag{2}$$

with a simple prior $p_0(\theta) = \prod_\ell \mathcal{N}(0, \sigma_0^2 I)$.

(A) β–ELBO (mean of logs). We minimize the negative β–ELBO with K Monte Carlo draws:

$$\mathcal{L}_{\beta\text{-ELBO}} = -\frac{1}{K}\sum_{k=1}^{K} \log p\big(y \mid x, \theta^{(k)}\big) \;+\; \beta\, D_{\mathrm{KL}}\big(q_\phi(\theta \mid x)\,\|\,p_0(\theta)\big), \qquad \theta^{(k)} \sim q_\phi(\theta \mid x). \tag{3}$$

Here β=1 recovers standard VI; $\beta \neq 1$ implements capacity control / tempered VI (Higgins et al., 2017; Alemi et al., 2017; Dziugaite & Roy, 2017). We use simple warm-ups for β early in training.

(B) IWAE variant (log of means; tighter bound). As a reference, the importance-weighted bound is

$$\mathcal{L}_{\mathrm{IWAE}} = -\log\left(\frac{1}{K}\sum_{k=1}^{K} \frac{p_0(\theta^{(k)})\, p(y \mid x, \theta^{(k)})}{q_\phi(\theta^{(k)} \mid x)}\right), \qquad \theta^{(k)} \sim q_\phi(\theta \mid x), \tag{4}$$

which implicitly accounts for the KL via the weights and typically needs no extra β (Burda et al., 2016). We report main results with (A) for simplicity and stability.

KL decomposition (IC vs. LP). For *IC-FDN*, layer posteriors condition directly on x, so the KL sums over layers and averages over the minibatch. For *LP-FDN*, layer ℓ conditions on a sampled hidden state $a_{\ell-1}^{(k)}(x)$; the KL is therefore averaged over this upstream randomness:

$$D_{\mathrm{KL}}\big(q_{\phi,\ell}(\theta_\ell \mid a_{\ell-1}^{(k)}(x))\,\|\,p_0\big) \quad \text{with} \quad \mathbb{E}_k[\cdot] \text{ across samples } k.$$

With diagonal Gaussians, each layer's closed-form term is

$$D_{\mathrm{KL}}\big(\mathcal{N}(\mu, \operatorname{diag}\sigma^2)\,\|\,\mathcal{N}(0, \sigma_0^2 I)\big) = \tfrac{1}{2}\sum_j \left(\frac{\sigma_j^2 + \mu_j^2}{\sigma_0^2} - 1 - \log\frac{\sigma_j^2}{\sigma_0^2}\right),$$

and we implement the variance floor via $\sigma = \varepsilon + \mathrm{softplus}(\rho)$ (no hard clamp).

Remark. Future work should investigate *layer-specific* β schedules to control where uncertainty is expressed across depth (e.g., larger β in early layers for stability, smaller β near the output to permit output-scale variance), with the aim of tightening scale calibration ($b \to 1$, $a \to 0$) and improving AURC/CRPS under oscillatory OOD.

C HETEROSCEDASTIC LIKELIHOOD AND VARIANCE DECOMPOSITION

Variance decomposition (scalar case). Let $\theta \sim q_\phi(\theta \mid x)$ and, given (x, θ), let

$$Y \mid x, \theta \;\sim\; \mathcal{N}\big(\mu_\theta(x),\, \sigma_\theta^2(x)\big).$$

Then the predictive (marginal) variance decomposes as

$$\mathrm{Var}[Y \mid x] \;=\; \underbrace{\mathbb{E}_{\theta\sim q_\phi}\big[\sigma_\theta^2(x)\big]}_{\text{aleatoric}} \;+\; \underbrace{\mathrm{Var}_{\theta\sim q_\phi}\big[\mu_\theta(x)\big]}_{\text{epistemic}}.$$

Proof. By the law of total expectation, $\mathbb{E}[Y \mid x] = \mathbb{E}_\theta[\mathbb{E}[Y \mid x, \theta]] = \mathbb{E}_\theta[\mu_\theta(x)]$. By the law of total variance,

$$\mathrm{Var}[Y \mid x] = \mathbb{E}_\theta\big[\mathrm{Var}(Y \mid x, \theta)\big] + \mathrm{Var}_\theta\big(\mathbb{E}[Y \mid x, \theta]\big) = \mathbb{E}_\theta\big[\sigma_\theta^2(x)\big] + \mathrm{Var}_\theta\big[\mu_\theta(x)\big]. \quad \square$$

Vector-output version. For $Y \in \mathbb{R}^{d_y}$ with $Y \mid x, \theta \sim \mathcal{N}(\mu_\theta(x), \Sigma_\theta(x))$,

$$\mathrm{Cov}[Y \mid x] \;=\; \underbrace{\mathbb{E}_\theta\big[\Sigma_\theta(x)\big]}_{\text{aleatoric}} \;+\; \underbrace{\mathrm{Cov}_\theta\big[\mu_\theta(x)\big]}_{\text{epistemic}}.$$

The proof is identical, replacing variance by covariance and using the matrix form of the law of total variance.

Special cases. (i) *Homoscedastic, isotropic noise* ($\sigma_\theta^2(x) \equiv \sigma^2$): $\mathrm{Var}[Y \mid x] = \sigma^2 + \mathrm{Var}_\theta[\mu_\theta(x)]$. (ii) *Heteroscedastic noise* ($\sigma_\theta^2(x)$ depends on x, θ): the first term becomes an average $\mathbb{E}_\theta[\sigma_\theta^2(x)]$.

Monte Carlo estimators. With samples $\theta^{(k)} \sim q_\phi(\theta \mid x)$,

$$\hat{\mu}(x) = \frac{1}{K}\sum_{k=1}^{K} \mu_{\theta^{(k)}}(x), \qquad \widehat{\mathrm{Var}}_{\text{epi}}(x) = \frac{1}{K}\sum_{k=1}^{K} \big(\mu_{\theta^{(k)}}(x) - \hat{\mu}(x)\big)^2,$$

$$\widehat{\mathrm{Var}}[Y \mid x] = \frac{1}{K}\sum_{k=1}^{K} \sigma^2_{\theta^{(k)}}(x) \;+\; \widehat{\mathrm{Var}}_{\text{epi}}(x).$$

For $d_y > 1$, replace squares by outer products to estimate covariances.

Derivation of the β-ELBO for $d_y{=}1$ and homoscedastic noise. Consider the latent-weight model

$$\theta \sim p_0(\theta), \qquad y \mid x, \theta \sim \mathcal{N}\big(f_\theta(x),\, \sigma^2\big),$$

with a variational family $q_\phi(\theta \mid x)$ (IC-/LP-FDN). For one datum (x_i, y_i) the β-ELBO is

$$\log p(y_i \mid x_i) \;\geq\; \underbrace{\mathbb{E}_{q_\phi}\big[\log p(y_i \mid x_i, \theta)\big]}_{\text{data term}} \;-\; \underbrace{D_{\mathrm{KL}}\big(q_\phi(\theta \mid x_i) \,\|\, p_0(\theta)\big)}_{\text{regularizer}}.$$

Using the Gaussian likelihood,

$$\log p(y_i \mid x_i, \theta) = -\frac{1}{2\sigma^2}\big(y_i - f_\theta(x_i)\big)^2 \;-\; \tfrac{1}{2}\log(2\pi\sigma^2).$$

Plugging into the bound and negating yields the per-example loss

$$\mathcal{L}^{(i)}_{\beta\text{-ELBO}} = \frac{1}{2\sigma^2}\,\mathbb{E}_{q_\phi(\theta|x_i)}\big[(y_i - f_\theta(x_i))^2\big] \;+\; D_{\mathrm{KL}}\big(q_\phi(\theta \mid x_i) \,\|\, p_0(\theta)\big) \;+\; \tfrac{1}{2}\log(2\pi\sigma^2).$$

Using K re-parameterized samples $\theta^{(k)} \sim q_\phi(\theta \mid x_i)$ gives the unbiased MC estimator

$$\boxed{\mathcal{L}^{(i)}_{\beta\text{-ELBO}} \approx \frac{1}{2K\sigma^2}\sum_{k=1}^{K}\big(y_i - f_{\theta^{(k)}}(x_i)\big)^2 \;+\; D_{\mathrm{KL}}\big(q_\phi(\theta \mid x_i) \,\|\, p_0(\theta)\big) \;+\; \tfrac{1}{2}\log(2\pi\sigma^2)}.$$

Since $\frac{1}{2}\log(2\pi\sigma^2)$ does not depend on ϕ or θ, it can be dropped during optimization. If σ^2 is fixed, the data term is just a rescaled MSE. Equivalently,

$$(2\sigma^2)\,\mathcal{L}^{(i)}_{\beta\text{-ELBO}} \doteq \frac{1}{K}\sum_{k=1}^{K}\big(y_i - f_{\theta^{(k)}}(x_i)\big)^2 \;+\; \underbrace{(2\sigma^2)}_{\beta}\, D_{\mathrm{KL}}\big(q_\phi(\theta \mid x_i)\,\|\,p_0(\theta)\big),$$

showing that choosing constant σ^2 is equivalent to training with β-ELBO, hence, it simply rescales the effective KL weight (capacity control). In this paper we will utilize β-ELBO for training, the heteroscedastic case is discussed below.

Heteroscedastic observation model (general form). If we allow the observation variance to depend on x and the sampled weights θ,

$$Y \mid x, \theta \sim \mathcal{N}\big(f_\theta(x),\, \sigma^2_\theta(x)\big) \qquad (d_y = 1),$$

the per-example β-ELBO becomes

$$\mathcal{L}^{(i)}_{\text{het}} = \frac{1}{2K}\sum_{k=1}^{K}\left[\frac{\big(y_i - f_{\theta^{(k)}}(x_i)\big)^2}{\sigma^2_{\theta^{(k)}}(x_i)} \;+\; \log\big(2\pi\,\sigma^2_{\theta^{(k)}}(x_i)\big)\right] \;+\; D_{\mathrm{KL}}\big(q_\phi(\theta \mid x_i)\,\|\,p_0(\theta)\big).$$

Thus, the data term is *weighted least squares* (WLS) plus a variance penalty, with weights $w^{(k)}(x_i) = 1/\sigma^2_{\theta^{(k)}}(x_i)$ learned jointly.

Parameterization and stability. We parameterize

$$\sigma_\theta(x) \;=\; \varepsilon + \mathrm{softplus}\,\big(\rho_\theta(x)\big), \qquad \varepsilon = 10^{-3},$$

which guarantees positivity and avoids numerical collapse. To mitigate variance blow-up in early training, one can (i) apply gentle weight decay on ρ_θ, (ii) clip $s_\theta(x) = \log\sigma^2_\theta(x)$ to a reasonable range, or (iii) use a short β warm up so the likelihood term dominates initially.

Predictive variance. With heteroscedastic noise, the predictive variance decomposes as

$$\mathrm{Var}[Y \mid x] = \underbrace{\mathbb{E}_\theta\big[\sigma^2_\theta(x)\big]}_{\text{aleatoric}} + \underbrace{\mathrm{Var}_\theta\big[f_\theta(x)\big]}_{\text{epistemic}},$$

so both terms adapt with x; the first is averaged over the sampled weights.

Table 1: Monte Carlo and training Hyperparameters (defaults unless noted).

Component	**Symbol**	**Setting**
MC samples (train)	K_{train}	1
MC samples (validate)	K_{val}	100
MC samples (test)	K_{test}	100
Epochs	ϵ	400
Optimizer	—	ADAM
Learning rate	η	1×10^{-3}
Batch size	B	64
Weight prior std	σ_0	1.0
Variance floor	ε	10^{-3} in $\sigma = \varepsilon + \text{softplus}(\rho)$
KL schedule	β_t	cosine
Maximum β	$\beta_{\max}$	0.01
Warm up updates	—	200
Seeds	—	$[7, 8, 9]$
Stochastic Checkpoint	—	minimum MSE in interpolation
Deterministic Checkpoint	—	minimum MSE in interpolation

Table 2: Model configurations and compute. Columns: base hidden width d_{hid}; hypernetwork hidden width d_{hyper}; latent dim d_h (for Gaussian Hypernetwork); ensemble size M; parameter count P (per model). Use "—" where not applicable.

Model	d_{hid}	d_{hyper}	d_h	M	P
MLPDropoutNet	333	—	—	1	1000
Deep Ensemble	64	—	—	10	1000
BayesNet	166	—	—	1	998
Gaussian HyperNet	24	5	9	1	994
IC-FDNet	23	6	—	1	1004
LP-FDNet	24	5	—	1	1011

Notes. We can see that the parameter count is roughly equal. In order to keep a fair comparison we scale the number of epochs by the ensemble size so the number of updates is roughly the same.

D TABLES

Table 3: Step function: unified calibration/uncertainty summary. Lower is better for AURC and deltas (Δ=OOD–ID); ideal MSE–Var fit has $a \approx 0$, $b \approx 1$.

Model	ρ	b	a	AURC ↓	ΔVar (OOD–ID) ↑	ΔMSE (OOD–ID) ↓	ΔCRPS (OOD–ID) ↓
MLPDropoutNet	0.904	0.995	0.041	0.223	0.561	0.516	0.033
LP-FDNet	0.955	1.250	0.032	0.320	1.200	1.440	0.147
IC-FDNet	0.848	1.180	0.303	0.484	0.643	0.990	0.252
DeepEnsembleNet	0.911	19.600	0.412	0.632	0.093	2.150	0.883
BayesNet	0.940	201.000	−2.750	0.704	0.014	2.760	1.170
GaussHyperNet	0.942	3.690	−0.356	1.010	1.060	3.700	0.780

Table 4: Sine function: unified calibration/uncertainty summary. Lower is better for AURC and deltas (Δ=OOD–ID); ideal MSE–Var fit has $a \approx 0$, $b \approx 1$.

Model	ρ	b	a	AURC ↓	ΔVar (OOD–ID) ↑	ΔMSE (OOD–ID) ↓	ΔCRPS (OOD–ID) ↓
DeepEnsembleNet	0.283	2.120	1.160	1.090	0.131	0.381	−0.022
GaussHyperNet	0.508	1.200	1.120	1.460	0.689	0.840	−0.068
BayesNet	0.875	14.100	−4.760	3.690	1.500	18.000	2.440
LP-FDNet	0.933	29.400	−17.200	6.970	3.600	87.100	5.320
MLPDropoutNet	0.974	32.200	−4.850	16.900	3.190	99.000	7.700
IC-FDNet	0.996	99.400	−36.800	40.100	3.420	310.000	13.700

Table 5: Quadratic function: unified calibration/uncertainty summary. Lower is better for AURC and deltas (Δ=OOD–ID); ideal MSE–Var fit has $a \approx 0$, $b \approx 1$.

Model	ρ	b	a	AURC ↓	ΔVar (OOD–ID) ↑	ΔMSE (OOD–ID) ↓	ΔCRPS (OOD–ID) ↓
MLPDropoutNet	0.994	70.700	−0.003	0.002	0.000	0.014	0.083
DeepEnsembleNet	0.994	18.200	−0.003	0.005	0.002	0.040	0.133
BayesNet	0.928	23.500	−0.125	0.012	0.003	0.047	0.114
IC-FDNet	1.000	1.040	−0.007	0.201	0.619	0.637	0.111
GaussHyperNet	1.000	1.060	−0.013	0.229	0.828	0.865	0.134
LP-FDNet	1.000	1.110	−0.012	0.242	0.715	0.790	0.133

LLM DISCLOSURE

ChatGPT assisted with minor copy-editing, LaTeX phrasing, and bibliography chores (suggesting candidate references and drafting BibTeX). The authors independently reviewed the literature and verified all citation metadata (titles, authors, venues, DOIs/arXiv).