HYAN: STATE-ADAPTIVE HIDDEN PERTURBATIONS FOR EFFICIENT ENSEMBLE WEATHER FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reliable weather forecasting requires both accurate predictions and calibrated uncertainty estimates, yet diffusion-based ensemble methods often incur high inference costs due to iterative sampling. We propose Hyan, an efficient ensemble weather forecasting model with state-adaptive noise injection into intermediate representations, and Hyan DE, its deep-ensemble variant. At each injection layer, a lightweight bottleneck network uses local latent features to predict mixing coefficients that control the relative contributions of latent features and stochastic noise. This mechanism produces spatially varying, state-dependent perturbations and requires only one forward pass per ensemble member at each forecast step. On WeatherBench2 at 1.5° resolution, Hyan DE reduces CRPS by an average of 7.92% relative to IFS ENS over the first five forecast days and outperforms ArchesWeatherGen at only 10.76% of its inference cost. Ablation experiments show that explicit state conditioning improves probabilistic forecast skill and the agreement between ensemble spread and forecast error. Visualizations further reveal distinct spatial modulation patterns across perturbation layers, vertical groups, and forecast lead times.

1 INTRODUCTION

Recent advances in AI-based weather prediction (AIWP) have led to substantial improvements in deterministic forecasting. On several benchmarks, AIWP models have matched or surpassed the forecast skill of operational numerical weather prediction (NWP) systems, such as the European Centre for Medium-Range Weather Forecasts' Integrated Forecasting System (ECMWF IFS) (Bi et al., 2023; Lam et al., 2023). Nevertheless, atmospheric evolution is inherently nonlinear and chaotic, making forecast uncertainty unavoidable (Lorenzelli, 2014). Models trained with deterministic losses tend to learn the conditional mean rather than the full conditional distribution, resulting in increasingly smooth forecasts that may miss small-scale signals and extreme events at longer lead times (Ben Bouallegue et al., 2024; Bonavita, 2024). Ensemble forecasting, which generates multiple plausible future trajectories, is therefore essential for representing forecast uncertainty and supporting risk-sensitive decisions (Bauer et al., 2015).

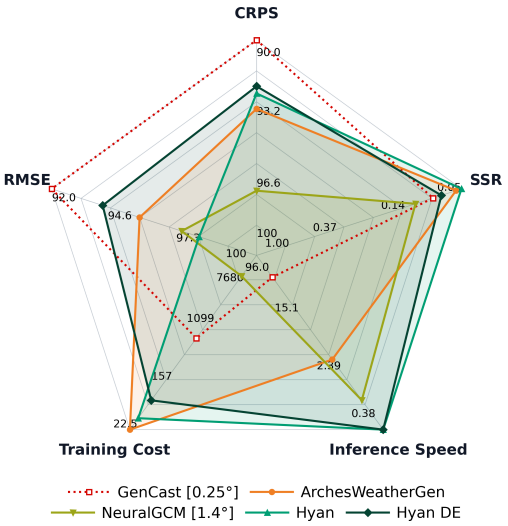


Figure 1: **Performance–efficiency comparison.** GenCast, ArchesWeatherGen, NeuralGCM, Hyan, and Hyan DE are evaluated in terms of RMSE, CRPS, SSR, inference speed, and training cost.

Generative ensemble forecasting represents forecast uncertainty through stochastic sampling. Diffusion models learn conditional distributions of future atmospheric states from reanalysis data, but iterative denoising makes sampling costly for high-dimensional global fields, especially for large ensembles (Song et al., 2023; Price et al., 2025). To reduce this cost, recent studies inject random

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

noise directly into forecasting networks, generating each member in a single forward pass per forecast step. For example, Alet et al. (2025) use adaptive layer normalization (AdaLN) (Peebles & Xie, 2023) to apply spatially shared, noise-driven modulation. However, most existing methods do not explicitly account for the state-dependent and spatially heterogeneous structure of atmospheric uncertainty (Zhang et al., 2003; Selz & Craig, 2015), which may limit their ability to represent the conditional distribution and produce skillful probabilistic forecasts.

In this work, we propose a state-conditioned noise injection method for efficient ensemble weather forecasting. At each injection layer, a lightweight low-dimensional bottleneck network uses local latent features to predict mixing coefficients for combining intermediate representations with stochastic noise. This design produces state-dependent and spatially varying perturbations while adding only a small number of parameters. Building on this method, we develop Hyan and its deep-ensemble variant, Hyan DE. Both generate each ensemble member with a single forward pass per forecast step, without iterative diffusion sampling. On WeatherBench2 at 1.5° resolution, Hyan DE reduces CRPS by an average of 7.92% relative to IFS ENS over the first five forecast days and outperforms ArchesWeatherGen in CRPS at only 10.76% of its inference cost. As summarized in Fig. 1, Hyan DE achieves a favorable balance across forecast accuracy, spread–error consistency, inference speed, and training cost.

The main contributions are summarized as follows:

- We propose Hyan, an efficient ensemble weather forecasting architecture that uses state-conditioned, layer-wise noise injection to generate each ensemble member with a single forward pass per forecast step.
- We demonstrate that state-adaptive perturbations improve probabilistic forecast skill and spread–error consistency. Layer-wise visualizations further reveal distinct spatial modulation patterns across injection layers and vertical groups.
- On WeatherBench2 at 1.5° resolution, Hyan DE reduces CRPS by an average of 7.92% relative to IFS ENS over the first five forecast days and outperforms ArchesWeatherGen at only 10.76% of its inference cost.

2 Related Work

Deep learning has substantially advanced deterministic weather forecasting in recent years. Representative models include FourCastNet (Pathak et al., 2022), Pangu-Weather (Bi et al., 2023), GraphCast (Lam et al., 2023), FuXi (Chen et al., 2023), FengWu (Chen et al., 2025b), Aurora (Bodnar et al., 2025), AIFS-Single (Moldovan et al., 2026), and others (Nguyen et al., 2023; 2024; 2026; Chen et al., 2025a). In their deterministic formulations, these models achieve strong forecast skill but provide point predictions, rather than explicitly characterizing the conditional distribution of future atmospheric states.

Complementing these advances, physically structured models incorporate physical constraints into the modeling of atmospheric evolution. ClimODE (Verma et al., 2024) and PASSAT (Zheng et al., 2025) integrate physical constraints into continuous-time formulations while retaining deterministic point predictions. NeuralGCM (Kochkov et al., 2024) combines a differentiable dynamical core with learned physical parameterizations, with its stochastic variant further extending this hybrid modeling approach to ensemble forecasting.

Generative models characterize forecast uncertainty by sampling plausible future atmospheric states. GenCast (Price et al., 2025), ArchesWeatherGen (Couairon et al., 2026), and ERDM (Cachay et al., 2025) employ diffusion-based approaches for ensemble generation and demonstrate strong probabilistic forecast skill. However, their iterative denoising procedures increase inference costs, particularly when generating large ensembles or performing long autoregressive rollouts.

Recent methods, including AIFS-CRPS (Lang et al., 2026), FGN (Alet et al., 2025), FourCastNet 3 (Bonev et al., 2025), Aurora 1.5 (Weyn et al., 2026), and UCast (Cachay et al., 2026), enable efficient ensemble forecasting without iterative diffusion sampling. These methods improve the practical efficiency of probabilistic forecasting while retaining the ability to represent forecast uncertainty. Hyan follows this direction by adapting latent perturbations to local atmospheric states, requiring only a single forward pass per ensemble member at each forecast step.

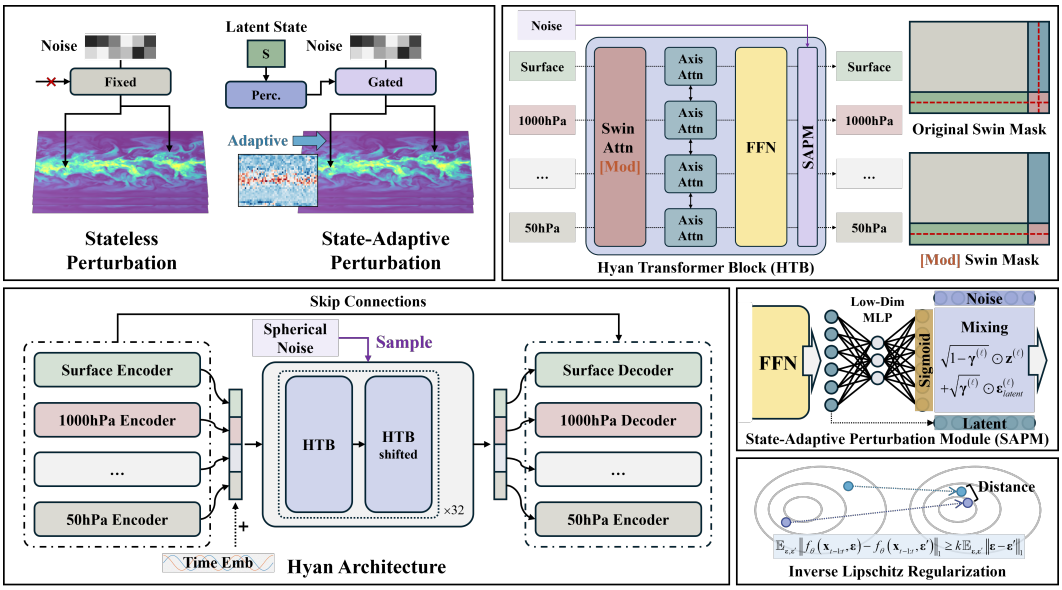


Figure 2: **Overview of the Hyan architecture.** Hyan introduces a state-adaptive perturbation paradigm, using a low-dimensional bottleneck MLP (Perc.) to extract latent information and guide adaptive perturbation injection. The backbone combines modified SwinV2 and AxisAttn modules, with separate encoders and decoders for 14 vertical levels and a shared 32-layer processor.

3 Preliminaries

3.1 Problem Formulation

Let $\mathbf{x}_t \in \mathbb{R}^{C\times H\times W}$ denote the atmospheric state at time t, where C is the number of atmospheric channels and H and W are the spatial grid dimensions. We use a 121×240 grid at $1.5°$ resolution. Given two consecutive atmospheric states, $\mathbf{x}_{t-1:t} = (\mathbf{x}_{t-1}, \mathbf{x}_t)$, our goal is to learn the conditional distribution

$$p(\mathbf{x}_{t+1} \mid \mathbf{x}_{t-1:t}).$$

This formulation represents uncertainty in future atmospheric states conditioned on the available inputs.

At inference time, the model generates forecast trajectories autoregressively. It first samples $\hat{\mathbf{x}}_{t+1}$ conditioned on $\mathbf{x}_{t-1:t}$, then samples $\hat{\mathbf{x}}_{t+2}$ conditioned on $(\mathbf{x}_t, \hat{\mathbf{x}}_{t+1})$, and continues this process over subsequent forecast steps.

3.2 Noise-Conditioned Ensemble Forecasting

We consider a stochastic forecasting model that maps the input atmospheric states and an independent noise variable to a next-state forecast. For an ensemble of M members,

$$\boldsymbol{\varepsilon}_{t+1}^{(m)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}_K), \qquad m = 1, \ldots, M, \tag{1}$$

where K is the noise dimension. Each member is generated as

$$\hat{\mathbf{x}}_{t+1}^{(m)} = f_\theta\left(\mathbf{x}_{t-1:t}, \boldsymbol{\varepsilon}_{t+1}^{(m)}\right). \tag{2}$$

The ensemble provides an empirical representation of the model's predictive distribution, which aims to approximate $p(\mathbf{x}_{t+1} \mid \mathbf{x}_{t-1:t})$. For multistep forecasting, each member is advanced using its own forecast history, with noise sampled independently at each step.

An effective ensemble should represent conditional uncertainty through meteorologically plausible and appropriately dispersed forecasts. If, for a fixed input, the model output varies little across

noise realizations, the predictive distribution collapses toward a deterministic forecast. We refer to this behavior as *conditional noise insensitivity*. This behavior is analogous to latent-variable neglect associated with posterior collapse in variational autoencoders (Wang et al., 2021).

4 METHODOLOGY

Hyan combines a Transformer backbone with state-conditioned noise injection to generate each ensemble member in a single forward pass. We describe the backbone, perturbation module, and training strategy below.

4.1 LEVEL-WISE TRANSFORMER BACKBONE

As illustrated in Figure 2, the Hyan backbone consists of level-specific encoders, a shared latent-space processor, and level-specific decoders. Atmospheric variables are organized into $V = 14$ groups: 1 surface group and 13 pressure-level groups. Each group is encoded separately, while the processor exchanges information across groups through axial attention. UNet-style skip connections link each encoder to its corresponding decoder at multiple spatial scales.

Let $\mathbf{x}_{t-1:t}^{(v)}$ denote the input states for group v at two consecutive time steps. The backbone computation is formulated as

$$\left(\mathbf{z}_t^{(v)}, \mathcal{F}_t^{(v)}\right) = \mathcal{S}_{\phi_v}\left(\mathbf{x}_{t-1:t}^{(v)}\right), \tag{3}$$

$$\mathbf{Z}_t = \mathrm{Stack}_{v=1}^{V}\left(\mathbf{z}_t^{(v)}\right) + \mathbf{T}_{emb}, \tag{4}$$

$$\hat{\mathbf{Z}}_{t+1} = \mathcal{P}_\theta\left(\mathbf{Z}_t\right), \tag{5}$$

$$\hat{\mathbf{x}}_{t+1}^{(v)} = \mathcal{D}_{\psi_v}\left(\hat{\mathbf{z}}_{t+1}^{(v)}, \mathcal{F}_t^{(v)}\right), \tag{6}$$

where $\mathbf{z}_t^{(v)}$ is the latent representation of group v, $\mathcal{F}_t^{(v)}$ collects its multiscale encoder features, and $\mathbf{T}_{\mathrm{emb}}$ denotes the temporal embedding inspired by (Bodnar et al., 2025).

The encoders and decoders have separate parameters for each group, allowing level-specific mappings between atmospheric variables and latent features. The shared processor combines spatial processing with axial attention along the level dimension. The backbone uses Swin Transformer V2 blocks (Liu et al., 2022) with SwiGLU feed-forward networks (Shazeer, 2020).

4.2 STATE-ADAPTIVE LATENT PERTURBATION

Spherical Noise Generation. We construct random noise fields on the sphere using a white angular power spectrum truncated at degree 120, with equal expected power across the retained spherical harmonic modes. We sample one spherical noise field per level group on a 120×240 grid and map it to the latent feature space. Applying pixel unshuffle n times rearranges spatial entries into channels, yielding a noise tensor $\boldsymbol{\varepsilon}$ of shape $(4^n V,\ 120/2^n,\ 240/2^n)$. A channel-wise linear projection followed by batch normalization maps this tensor to the latent feature space:

$$\boldsymbol{\varepsilon}_{\mathrm{latent}} = \mathrm{BatchNorm}\left(\mathbf{W}_{\mathrm{proj}}\boldsymbol{\varepsilon}\right), \tag{7}$$

where $\mathbf{W}_{\mathrm{proj}} \in \mathbb{R}^{D_{\mathrm{latent}} \times 4^n V}$ and D_{latent} is the latent channel dimension.

State Conditioning via a Low-Dimensional Bottleneck. We use the intermediate features to adapt the perturbation strength to the atmospheric state. At injection layer ℓ, a bottleneck mapping takes the latent representation $\mathbf{z}^{(\ell)}$ as input and predicts an element-wise mixing coefficient $\boldsymbol{\gamma}^{(\ell)}$:

$$\boldsymbol{\gamma}^{(\ell)} = \sigma\Big(\mathbf{W}_{\mathrm{up}}^{(\ell)}\kappa\Big(\mathbf{W}_{\mathrm{down}}^{(\ell)}\mathbf{z}^{(\ell)}\Big)\Big), \tag{8}$$

where $\mathbf{W}_{\mathrm{down}}^{(\ell)} \in \mathbb{R}^{r_\ell \times D_\ell}$, $\mathbf{W}_{\mathrm{up}}^{(\ell)} \in \mathbb{R}^{D_\ell \times r_\ell}$, and $r_\ell = D_\ell/4$. Here, D_ℓ is the feature dimension, κ is the bottleneck activation, and σ is the sigmoid function. Bias terms are omitted for clarity.

The mapping is applied independently at each spatial location with shared weights, allowing the mixing coefficients to vary with local features. The sigmoid constrains each coefficient to $(0, 1)$, while the bottleneck limits the additional parameter cost.

Adaptive Perturbation Injection. At layer ℓ, let $\boldsymbol{\varepsilon}^{(\ell)}_{\text{latent}}$ denote the noise features with the same shape as $\mathbf{z}^{(\ell)}$. Inspired by the forward noising formulation of DDPM (Ho et al., 2020), we combine latent features and noise using state-dependent mixing coefficients:

$$\tilde{\mathbf{z}}^{(\ell)} = \sqrt{1-\boldsymbol{\gamma}^{(\ell)}} \odot \mathbf{z}^{(\ell)} + \sqrt{\boldsymbol{\gamma}^{(\ell)}} \odot \boldsymbol{\varepsilon}^{(\ell)}_{\text{latent}}, \tag{9}$$

where $\odot$ denotes element-wise multiplication and the square roots are applied element-wise. Unlike the prescribed noise schedule in DDPM, $\boldsymbol{\gamma}^{(\ell)}$ is predicted from the current latent features.

4.3 Two-Stage Training

Hyan is trained in two stages. We first pretrain the deterministic backbone without perturbation modules, then introduce the perturbation modules and fine-tune the model with a probabilistic objective and progressively longer autoregressive rollouts.

Stage 1: Deterministic Pretraining

We pretrain the backbone using an area- and pressure-weighted L_1 loss. For atmospheric fields $\mathbf{a}$ and $\mathbf{b}$, define

$$d_w(\mathbf{a}, \mathbf{b}) = \frac{1}{VHW} \sum_{v,h,w} \omega_h \omega_v \left\| \mathbf{a}(v,h,w) - \mathbf{b}(v,h,w) \right\|_1, \tag{10}$$

where the L_1 norm sums over the variables within each level group, ω_h is the latitude-dependent area weight, and ω_v is the level weight. For pressure-level groups,

$$\omega_v = \cos\left(\frac{1000 - p_v}{1000} \frac{\pi}{2} \right), \tag{11}$$

where p_v is the pressure in hPa. The surface-group weight and the normalization of latitude weights are specified in the experimental setup.

The deterministic pretraining objective is

$$\mathcal{L}_{\text{det}} = d_w \left(\mathbf{x}_{t+1}, f^{\text{det}}_\theta \left(\mathbf{x}_{t-1:t} \right) \right), \tag{12}$$

where f^{det}_θ is the backbone without perturbation modules.

Stage 2: Probabilistic Fine-Tuning

We initialize the backbone from the pretrained model, introduce the perturbation modules, and jointly fine-tune the resulting stochastic model. A deterministic loss alone can favor predictions that vary little across noise realizations. To discourage such conditional noise insensitivity, we combine the prediction error with a regularizer that promotes ensemble diversity.

For a fixed input $\mathbf{h} = \mathbf{x}_{t-1:t}$ and target $\mathbf{o} = \mathbf{x}_{t+1}$, let $\mathbf{y} = f_\theta(\mathbf{h}, \boldsymbol{\varepsilon})$ and $\mathbf{y}' = f_\theta(\mathbf{h}, \boldsymbol{\varepsilon}')$ be forecasts generated from independent noise samples. A hard inverse-Lipschitz condition (Kinoshita et al., 2023) with respect to the noise input would require

$$\left\| f_\theta(\mathbf{h}, \boldsymbol{\varepsilon}) - f_\theta(\mathbf{h}, \boldsymbol{\varepsilon}') \right\|_1 \geq k \left\| \boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}' \right\|_1, \qquad k > 0, \tag{13}$$

for every pair of admissible noise realizations. Enforcing this pointwise condition is difficult in practice, as it must hold for all pairs of noise inputs. We therefore consider a relaxed form:

$$\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}'} \left[\left\| f_\theta(\mathbf{h}, \boldsymbol{\varepsilon}) - f_\theta(\mathbf{h}, \boldsymbol{\varepsilon}') \right\|_1 \right] \geq k \, \mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}'} \left[\left\| \boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}' \right\|_1 \right]. \tag{14}$$

This condition requires the model to preserve noise-induced variation on average, rather than for every individual pair of noise samples. Since the noise distribution is fixed, the right-hand side of Eq. 14 is a constant. We therefore encourage this condition by maximizing the expected separation between forecasts generated from different noise realizations.

Accordingly, we use the following objective:

$$\mathcal{L}_{\alpha,\lambda} = \alpha \, \mathbb{E} \left[d_w(\mathbf{y}, \mathbf{o}) \right] - \lambda \, \mathbb{E} \left[d_w(\mathbf{y}, \mathbf{y}') \right], \tag{15}$$

where $\alpha, \lambda > 0$ balance prediction error and ensemble diversity. The second term serves as a soft inverse-Lipschitz regularizer by encouraging the model to maintain noise-dependent variation in its outputs.

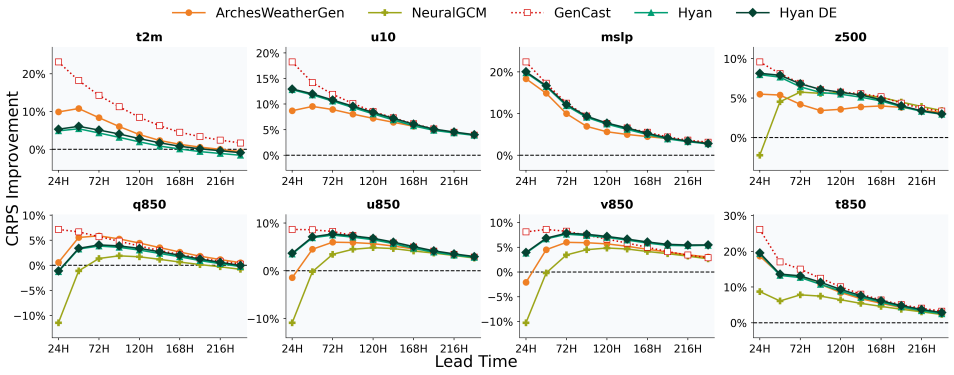


Figure 3: **WeatherBench2 CRPS.** This figure shows the 10-day CRPS trajectories for eight variables. Hyan and Hyan DE substantially outperform the other 1.5° baselines, demonstrating consistently better probabilistic forecast skill throughout the forecast period.

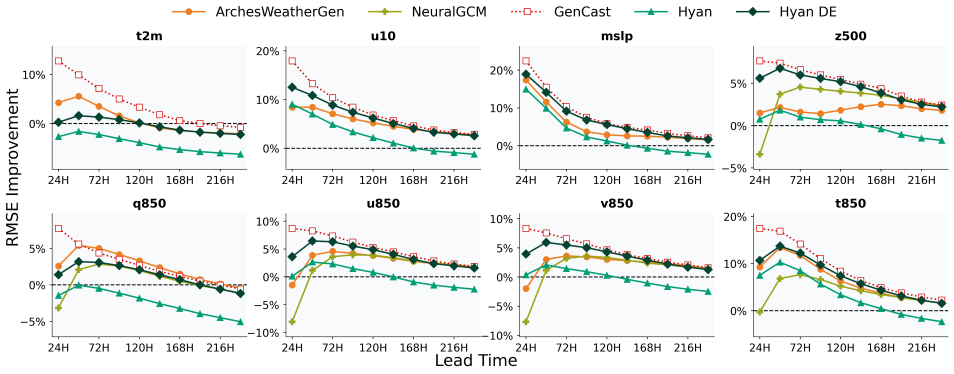


Figure 4: **WeatherBench2 RMSE.** Similar to the CRPS results, Hyan DE achieves a clear advantage over the other baselines across the eight variables and the 10-day forecast horizon. Deep Ensemble also provides a notable improvement in RMSE, indicating that ensemble diversity can further enhance deterministic forecast accuracy.

For $M \geq 2$ independently sampled ensemble members, we estimate this objective as

$$\widehat{\mathcal{L}}_{\alpha,\lambda} = \frac{\alpha}{M} \sum_{m=1}^{M} d_w\left(\mathbf{y}^{(m)}, \mathbf{o}\right) - \frac{\lambda}{M(M-1)} \sum_{m \neq j} d_w\left(\mathbf{y}^{(m)}, \mathbf{y}^{(j)}\right), \tag{16}$$

where the second sum is over ordered pairs of distinct members. By the triangle inequality, when $\alpha \geq 2\lambda$, both the population objective and its finite-ensemble estimator are bounded from below, so increasing ensemble dispersion cannot drive the objective to $-\infty$. At $\alpha = 1$ and $\lambda = \frac{1}{2}$, the estimator equals the weighted sum of componentwise fair CRPS scores (Ferro, 2014; Leutbecher, 2019). More generally, $\alpha = 2\lambda$ recovers this score up to a positive scaling factor. Proofs are provided in Appendix E.

Following a strategy similar to that of Keisler (2022), we progressively increase the autoregressive forecast horizon during fine-tuning, adding one step per training round over R rounds. Each ensemble member is advanced using its own forecast history. We use truncated backpropagation through time to limit the gradient propagation length. The rollout schedule, multistep loss aggregation, and truncation settings are specified in the experimental setup.

5 Experiments

5.1 Experimental Settings

Datasets. We use the ERA5 reanalysis data distributed through WeatherBench2 (Rasp et al., 2024) at 1.5° spatial resolution and 6-hour intervals. The data are split into 1979–2018 for training, 2019 for

Table 1: **Ablation of perturbation modules within Hyan.** Results are averaged over the first five forecast days. **Perf. change** measures the relative change in CRPS gain over IFS ENS compared with full Hyan.

Configuration	Metric	Variables				Average
		$t2m$	$z500$	$q850$	$u850$	
Global AdaLN	CRPS	0.4041	67.18	459.88	1.0434	–
	SSR	0.9399	1.2140	0.9214	0.9932	
	Perf. change	-127.21%	-194.61%	-135.29%	-79.50%	-134.15%
Feature Dropout	CRPS	0.3853	60.49	443.66	0.9875	–
	SSR	0.8535	1.0393	0.8877	0.9751	
	Perf. change	+0.68%	-23.07%	-8.37%	+1.15%	-7.40%
Static Gate	CRPS	0.3869	59.58	445.58	0.9882	–
	SSR	0.8018	0.9788	0.8624	0.9296	
	Perf. change	-10.20%	+0.25%	-23.39%	+0.14%	-8.29%
Hyan (State-conditioned Gate)	CRPS	0.3854	59.59	442.59	0.9883	–
	SSR	0.8778	1.0526	0.9130	0.9845	

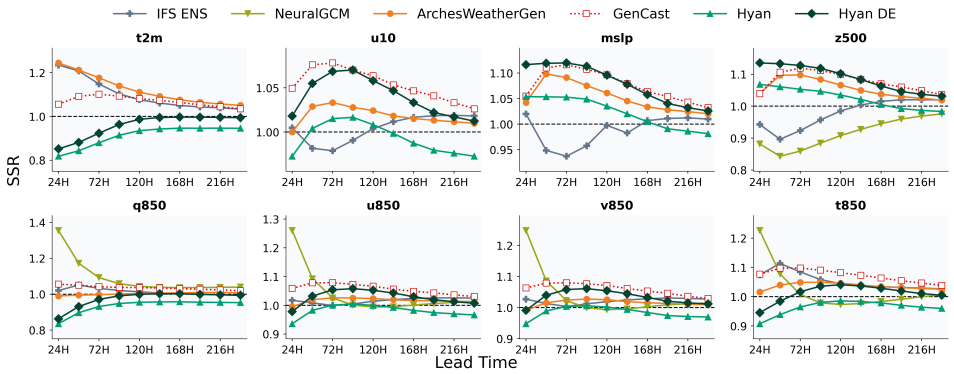


Figure 5: **WeatherBench2 SSR.** This figure compares the SSR of different models on the test set. Except for $t2m$ and $q850$, Hyan and Hyan DE generally achieve competitive spread–error consistency. Hyan DE produces higher SSR values than Hyan across all variables.

validation, and 2020 for testing. Probabilistic fine-tuning uses the 2000–2018 subset of the training data. The complete list of atmospheric variables and pressure levels is provided in Appendix A. All input variables are standardized before training.

Baselines. To evaluate our method against strong, well-established baselines, we use official results reported on WeatherBench2 rather than unofficial reproductions. These extensively optimized models represent leading approaches to ensemble weather forecasting. Our baselines include GenCast at $0.25°$, ArchesWeatherGen at $1.5°$, NeuralGCM at $1.4°$, and the operational NWP ensemble model IFS ENS. For a fairer comparison, we report IFS ENS scores verified against its corresponding analysis. Detailed experimental settings for Hyan are provided in Appendix C.

Optimization. We use Muon (Liu et al., 2025) for matrix-valued parameters and AdamW for scalar- and vector-valued parameters. During probabilistic fine-tuning, we set $\lambda = \alpha/2$ and progressively increase the autoregressive rollout length to a maximum of four forecast steps. Additional optimization settings are reported in Appendix C.

Ensemble Construction. For probabilistic fine-tuning, we use an ensemble size of $M = 3$. At evaluation time, Hyan generates $M = 10$ ensemble members. Hyan DE consists of 4 independently fine-tuned Hyan instances, each generating 10 members with a different random seed, resulting in a total ensemble size of $M = 40$. We use the fair CRPS estimator for ensemble evaluation, which provides an unbiased finite-ensemble estimate and therefore avoids finite-ensemble bias when comparing models with different ensemble sizes.

6 WeatherBench2 1.5° Evaluation

This section evaluates Hyan and its deep-ensemble variant, Hyan DE, on key meteorological variables from WeatherBench2. We evaluate forecasts initialized at 00:00 and 12:00 UTC throughout 2020 over a 10-day forecast horizon. Representative results are shown in Figs. 3, 4, and 5, with complete results provided in Appendix F.

As shown in Fig. 3, Hyan and Hyan DE achieve competitive probabilistic forecast skill across most variables and lead times. Over the first five forecast days, Hyan DE reduces mean CRPS by 7.92% relative to IFS ENS and by 1.46% relative to ArchesWeatherGen, a strong baseline at the same 1.5° resolution. Its CRPS increases more gradually with lead time than those of the other low-resolution baselines, indicating better retention of forecast skill at longer lead times. Beyond five days, both Hyan variants remain competitive with GenCast despite its finer 0.25° resolution.

Figure 4 shows that Hyan DE consistently lowers the RMSE of the ensemble mean relative to Hyan across a range of variables and lead times. The improvement is more pronounced for RMSE than for CRPS, suggesting that combining independently trained models mainly improves the ensemble-mean forecast, with smaller additional gains in probabilistic skill.

The SSR results in Fig. 5 reveal a complementary effect of deep ensembling on ensemble dispersion. When Hyan is underdispersive, Hyan DE brings SSR closer to one, improving spread–error consistency. When Hyan already has appropriate dispersion, however, Hyan DE can produce slight overdispersion. Deep ensembling therefore improves ensemble-mean accuracy more consistently than it improves spread–error consistency.

7 Ablation Study

We evaluate the proposed state-conditioned perturbation mechanism by replacing the perturbation module within Hyan. All variants retain the same perturbation locations and the remaining network architecture.

We consider four perturbation configurations:

- **Global AdaLN**: spatially shared, noise-driven modulation using adaptive layer normalization, following Alet et al. (2025).
- **Feature Dropout**: stochastic masking of latent features with a fixed dropout probability (Cachay et al., 2026).
- **Static Gate**: learned but state-independent perturbation mixing. We remove the state-dependent bottleneck mapping and retain only a learnable bias, followed by the same sigmoid transformation as in Hyan.
- **Hyan (State-conditioned Gate)**: the proposed module, which predicts spatially varying perturbation mixing coefficients from local latent features.

We report mean CRPS and spread–skill ratio (SSR), both averaged over the first five forecast days. Table 1 compares Hyan with three alternative perturbation mechanisms. Global AdaLN performs worst in terms of CRPS across all evaluated variables, with performance drops ranging from -79.50% to -194.61% and an average drop of -134.15% relative to Hyan. Although its SSR is closer to one for $t2m$ and $u850$, it is farther from one for $z500$ and $q850$. This indicates that globally shared modulation cannot adequately capture the spatially varying structure of forecast uncertainty.

Feature Dropout performs substantially better than Global AdaLN, but still loses part of Hyan's CRPS gain on average. The degradation is most pronounced for $z500$ and $q850$, with performance drops of -23.07% and -8.37%, respectively, while it yields marginally lower CRPS for $t2m$ and $u850$. Its SSR also varies across variables, suggesting that fixed-probability masking provides useful stochasticity but cannot adapt the perturbation to the atmospheric state.

Static Gate removes the state-dependent mapping while retaining learnable perturbation coefficients. It has an average performance drop of -8.29%, with the largest degradations for $t2m$ and $q850$ (-10.20% and -23.39%). Although its CRPS for $z500$ and $u850$ is nearly unchanged, its SSR is lower than Hyan's for all variables, indicating greater underdispersion. Overall, the results show

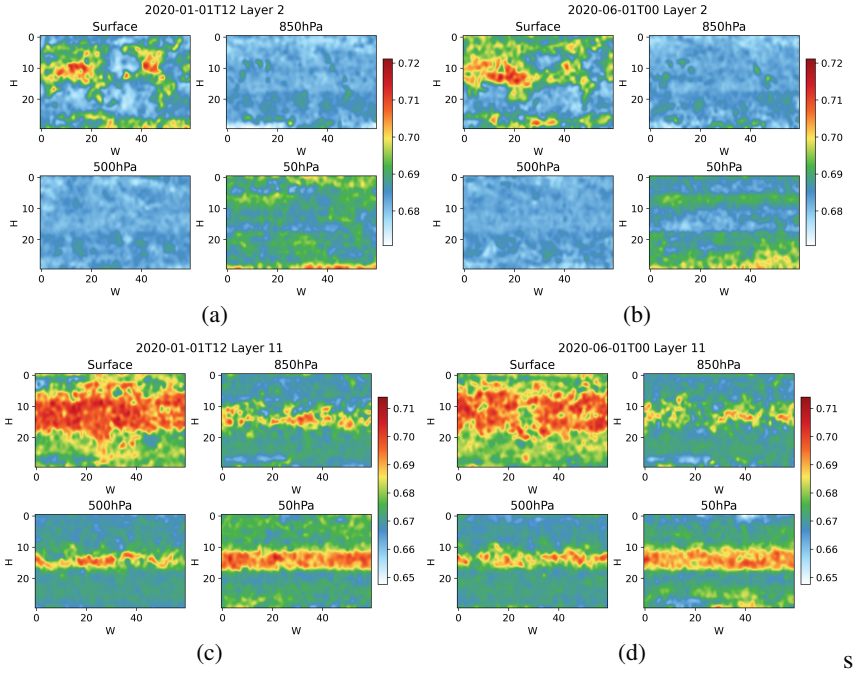


Figure 6: **Perturbation intensity at different init times/layers.** Additional results are provided in Appendix H.

that learnable perturbation strength alone is insufficient. Conditioning the perturbation coefficients on local latent features provides better probabilistic forecast skill and more appropriate ensemble dispersion.

8 Visualization Analysis

To visualize the spatial structure of the learned perturbations, we plot the perturbation intensity $\sqrt{r}$ at the second and eleventh of the 31 perturbation layers for four representative vertical groups: Surface, 850 hPa, 500 hPa, and 50 hPa.

As shown in Fig. 6, panels (a) and (c) correspond to forecasts initialized at 12:00 UTC on January 1, 2020, whereas panels (b) and (d) correspond to forecasts initialized at 00:00 UTC on June 1, 2020. The perturbation intensity exhibits distinct spatial distributions across vertical groups, perturbation layers, and initialization times, indicating that the state-conditioned network adapts the spatial structure of its perturbations to both the atmospheric level and the forecast state.

9 Conclusion

We presented Hyan, an efficient ensemble weather forecasting model that generates stochastic forecasts through state-adaptive perturbations of intermediate representations. A lightweight bottleneck network predicts mixing coefficients from local latent features, enabling spatially varying perturbations conditioned on the atmospheric state. Hyan generates each ensemble member with a single forward pass per forecast step, avoiding iterative diffusion sampling. On WeatherBench2 at 1.5° resolution, its deep-ensemble variant, Hyan DE, reduces CRPS by an average of 7.92% relative to IFS ENS over the first five forecast days and outperforms ArchesWeatherGen at only 10.76% of its inference cost. Ablation experiments support the benefits of state conditioning, while visualizations reveal distinct modulation patterns across perturbation layers, vertical groups, and initialization times.

REFERENCES

Ferran Alet, Ilan Price, Andrew El-Kadi, Dominic Masters, Stratis Markou, Tom R Andersson, Jacklynn Stott, Remi Lam, Matthew Willson, Alvaro Sanchez-Gonzalez, et al. Skillful joint probabilistic weather forecasting from marginals. *arXiv preprint arXiv:2506.10772*, 2025.

Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

Zied Ben Bouallegue, Mariana CA Clare, Linus Magnusson, Estibaliz Gascon, Michael Maier-Gerber, Martin Janoušek, Mark Rodwell, Florian Pinault, Jesper S Dramsch, Simon TK Lang, et al. The rise of data-driven weather forecasting: A first statistical assessment of machine learning–based weather forecasts in an operational-like context. *Bulletin of the American Meteorological Society*, 105(6):E864–E883, 2024.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

Cristian Bodnar, Wessel P Bruinsma, Ana Lucic, Megan Stanley, Anna Allen, Johannes Brandstetter, Patrick Garvan, Maik Riechert, Jonathan A Weyn, Haiyu Dong, et al. A foundation model for the earth system. *Nature*, 641(8065):1180–1187, 2025.

Massimo Bonavita. On some limitations of current machine learning weather prediction models. *Geophysical Research Letters*, 51(12):e2023GL107377, 2024.

Boris Bonev, Thorsten Kurth, Ankur Mahesh, Mauro Bisson, Jean Kossaifi, Karthik Kashinath, Anima Anandkumar, William D Collins, Michael S Pritchard, and Alexander Keller. Fourcastnet 3: A geometric approach to probabilistic machine-learning weather forecasting at scale. *arXiv preprint arXiv:2507.12144*, 2025.

Salva Rühling Cachay, Miika Aittala, Karsten Kreis, Noah Brenowitz, Arash Vahdat, Morteza Mardani, and Rose Yu. Elucidated rolling diffusion models for probabilistic forecasting of complex dynamics. *arXiv preprint arXiv:2506.20024*, 2025.

Salva Rühling Cachay, Duncan Watson-Parris, and Rose Yu. U-cast: A surprisingly simple and efficient frontier probabilistic ai weather forecaster. *arXiv preprint arXiv:2604.09041*, 2026.

Hao Chen, Han Tao, Guo Song, Jie Zhang, Yonghan Dong, Yunlong Yu, and Lei Bai. Va-moe: Variables-adaptive mixture of experts for incremental weather forecasting. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7915–7924. IEEE, 2025a.

Kang Chen, Tao Han, Fenghua Ling, Junchao Gong, Lei Bai, Xinyu Wang, Jing-Jia Luo, Ben Fei, Wenlong Zhang, Xi Chen, et al. The operational medium-range deterministic weather forecasting can be extended beyond a 10-day lead time. *Communications Earth & Environment*, 6(1):518, 2025b.

Lei Chen, Xiaohui Zhong, Feng Zhang, Yuan Cheng, Yinghui Xu, Yuan Qi, and Hao Li. Fuxi: A cascade machine learning forecasting system for 15-day global weather forecast. *npj climate and atmospheric science*, 6(1):190, 2023.

Guillaume Couairon, Renu Singh, Anastase Charantonis, Christian Lessig, and Claire Monteleoni. Archesweathergen: Skillful and compute-efficient probabilistic weather forecasting with machine learning. *Science Advances*, 12(17):eadx2372, 2026.

Christopher AT Ferro. Fair scores for ensemble forecasts. *Quarterly Journal of the Royal Meteorological Society*, 140(683):1917–1923, 2014.

Vincent Fortin, Mabrouk Abaza, Francois Anctil, and Raphael Turcotte. Why should ensemble spread match the rmse of the ensemble mean? *Journal of Hydrometeorology*, 15(4):1708–1713, 2014.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Ryan Keisler. Forecasting global weather with graph neural networks. *arXiv preprint arXiv:2202.07575*, 2022.

Yuri Kinoshita, Kenta Oono, Kenji Fukumizu, Yuichi Yoshida, and Shin-ichi Maeda. Controlling posterior collapse by an inverse lipschitz constraint on the decoder network. In *International Conference on Machine Learning*, pp. 17041–17060. PMLR, 2023.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.

Simon Lang, Mihai Alexe, Mariana CA Clare, Christopher Roberts, Rilwan Adewoyin, Zied Ben Bouallègue, Matthew Chantry, Jesper Dramsch, Peter D Dueben, Sara Hahner, et al. Aifs-crps: ensemble forecasting using a model trained with a loss function based on the continuous ranked probability score. *npj Artificial Intelligence*, 2(1):18, 2026.

Martin Leutbecher. Ensemble size: How suboptimal is less than infinity? *Quarterly Journal of the Royal Meteorological Society*, 145:107–128, 2019.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11999–12009. IEEE, 2022.

Flavio Lorenzelli. *The essence of chaos*. CRC Press, 2014.

Gabriel Moldovan, Ewan Pinnington, Ana Prieto Nemesio, Simon Lang, Zied Ben Bouallègue, Jesper Dramsch, Mihai Alexe, Mario Santa Cruz, Sara Hahner, Harrison Cook, et al. Aifs single 1.1. 0: an update to ecmwf's machine-learned weather forecast model aifs. *Geoscientific Model Development*, 19(10):4703–4724, 2026.

Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. Climax: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

Tung Nguyen, Rohan Shah, Hritik Bansal, Troy Arcomano, Romit Maulik, Veerabhadra Kotamarthi, Ian Foster, Sandeep Madireddy, and Aditya Grover. Scaling transformer neural networks for skillful and reliable medium-range weather forecasting. *Advances in Neural Information Processing Systems*, 37:68740–68771, 2024.

Tung Nguyen, Tuan Pham, Troy Arcomano, Rao Kotamarthi, Ian Foster, Sandeep Madireddy, and Aditya Grover. Omnicast: A masked latent diffusion model for weather forecasting across time scales. *Advances in Neural Information Processing Systems*, 38:172182–172212, 2026.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Gencast: Diffusion-based ensemble forecasting for medium-range weather. In *105th Annual AMS Meeting 2025*, volume 105, pp. 449275, 2025.

Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russell, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6):e2023MS004019, 2024.

Tobias Selz and George C Craig. Upscale error growth in a high-resolution simulation of a summertime weather event over europe. *Monthly Weather Review*, 143(3):813–827, 2015.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Yogesh Verma, Markus Heinonen, and Vikas Garg. Climode: Climate and weather forecasting with physics-informed neural odes. In *International Conference on Learning Representations*, volume 2024, pp. 8408–8430, 2024.

Yixin Wang, David Blei, and John P Cunningham. Posterior collapse and latent variable nonidentifiability. *Advances in neural information processing systems*, 34:5443–5455, 2021.

Jonathan A Weyn, Zekun Ni, Amit Misra, Will Fein, Haiyu Dong, Wessel P Bruinsma, Richard E Turner, Matt Corey, Kit Thambiratnam, Kevin White, et al. Aurora 1.5: Fine-tuning a foundation model for medium-range ensemble weather prediction. 2026.

Fuqing Zhang, Chris Snyder, and Richard Rotunno. Effects of moist convection on mesoscale predictability. *Journal of the Atmospheric Sciences*, 60(9):1173–1185, 2003.

Jiaqi Zheng, Qing Ling, and Yerong Feng. Physics-assisted and topology-informed deep learning for weather prediction. *arXiv preprint arXiv:2505.04918*, 2025.

APPENDIX

A DATASET DETAILS

Table 2 summarizes the variables used by Hyan. The model takes atmospheric and single-level variables as inputs and predicts their corresponding future states. The atmospheric variables include geopotential, specific humidity, zonal and meridional wind components, temperature, and vertical velocity. These variables are provided at the 13 standard pressure levels used in WeatherBench2 (Rasp et al., 2024), namely 50, 100, 150, 200, 250, 300, 400, 500, 600, 700, 850, 925, and 1000 hPa. The single-level variables include 2-metre temperature, 10-metre zonal and meridional wind components, and mean sea-level pressure. The model also receives temporal information and static geographical fields, including time, surface geopotential, and the land–sea mask. These auxiliary variables are used as inputs but are not predicted.

Table 2: **Description of data variables used.** The table summarizes the meteorological variables included in this study, together with their corresponding names and vertical levels.

Type	**Variable name**	**Short name**	**Role**
Atmospheric	geopotential	z	Input/Predicted
Atmospheric	specific humidity	q	Input/Predicted
Atmospheric	u component of wind	u	Input/Predicted
Atmospheric	v component of wind	v	Input/Predicted
Atmospheric	temperature	t	Input/Predicted
Atmospheric	vertical velocity	w	Input/Predicted
Single	2 metre temperature	$2t$	Input/Predicted
Single	10 metre u wind component	$10u$	Input/Predicted
Single	10 metre v wind component	$10v$	Input/Predicted
Single	mean sea level pressure	msl	Input/Predicted
Clock	time	N/A	Input
Static	geopotential at surface	z_s	Input
Static	land–sea mask	lsm	Input

B EVALUATION METRICS

For metric computation, we follow the evaluation protocol of WeatherBench2. The details are provided below.

Latitude-Dependent Area Weight. All spatial metrics use latitude-dependent area weighting. This weighting accounts for the varying grid-cell areas on the regular latitude–longitude grid and prevents high-latitude regions from being overrepresented. The latitude weights ω_h are defined as

$$\omega_h = \frac{\sin\theta_h^{\mathrm{u}} - \sin\theta_h^{\mathrm{l}}}{\frac{1}{H}\sum_{h'=1}^{H}\left(\sin\theta_{h'}^{\mathrm{u}} - \sin\theta_{h'}^{\mathrm{l}}\right)}, \tag{17}$$

where θ_h^{u} and θ_h^{l} denote the upper and lower latitude boundaries of the h-th grid cell, respectively, and H is the number of latitude grid cells.

Root Mean Squared Error (RMSE). For a fixed atmospheric variable or pressure level v, the RMSE at lead time i is defined as

$$\mathrm{RMSE}_i^{(v)} = \sqrt{\frac{1}{THW}\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{w=1}^{W}\omega_h\left(y_{t,i}^{(v)}(h,w) - o_{t+i}^{(v)}(h,w)\right)^2}, \tag{18}$$

where $y_{t,i}^{(v)}(h,w)$ denotes the forecast for variable or pressure level v, initialized at time t with lead time i, at grid location (h,w). The quantity $o_{t+i}^{(v)}(h,w)$ denotes the corresponding observation at the valid time $t+i$. The quantities T, H, and W denote the numbers of initialization times,

latitude points, and longitude points, respectively. The latitude-dependent area weight ω_h is defined in Eq. 17.

Continuous Ranked Probability Score (CRPS). For a fixed atmospheric variable or pressure level v, we first define the area-weighted spatial L_1 distance between two fields as

$$\|\mathbf{g}\|_{t,i}^{(v)} := \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} \omega_h \left| g_{t,i}^{(v)}(h,w) \right|, \tag{19}$$

where ω_h is the latitude-dependent area weight.

For an ensemble of $M \geq 2$ members $\{y^{(1,v)}, \ldots, y^{(M,v)}\}$, the finite-ensemble CRPS at lead time i is computed as

$$\begin{aligned} \mathrm{CRPS}_i^{(v)} = \frac{1}{T} \sum_{t=1}^{T} \Bigg[& \frac{1}{M} \sum_{m=1}^{M} \left\| y_{t,i}^{(m,v)} - o_{t+i}^{(v)} \right\|_{t,i}^{(v)} \\ & - \frac{1}{2M(M-1)} \sum_{m=1}^{M} \sum_{\substack{n=1 \\ n \neq m}}^{M} \left\| y_{t,i}^{(m,v)} - y_{t,i}^{(n,v)} \right\|_{t,i}^{(v)} \Bigg]. \end{aligned} \tag{20}$$

Spread–Skill Ratio (SSR). For a fixed atmospheric variable or pressure level v and lead time i, we evaluate the error of the ensemble mean separately for each initialization time:

$$\mathrm{RMSE}_{t,i}^{\prime(v)} = \sqrt{\frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} \omega_h \left(\frac{1}{M} \sum_{m=1}^{M} y_{t,i}^{(m,v)}(h,w) - o_{t+i}^{(v)}(h,w) \right)^2}. \tag{21}$$

Let $\sigma_{t,i,h,w}^{2,(v)}$ denote the variance of the ensemble predictions at grid location (h,w):

$$\sigma_{t,i,h,w}^{2,(v)} = \frac{1}{M} \sum_{m=1}^{M} \left(y_{t,i}^{(m,v)}(h,w) - \frac{1}{M} \sum_{n=1}^{M} f_{t,i}^{(n,v)}(h,w) \right)^2. \tag{22}$$

The ensemble spread is computed for each initialization time as the spatially weighted root mean square of the ensemble standard deviation:

$$\mathrm{Spread}_{t,i}^{(v)} = \sqrt{\frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} \omega_h \sigma_{t,i,h,w}^{2,(v)}}. \tag{23}$$

The SSR for each initialization time is then defined as

$$\mathrm{SSR}_i^{(v)} = \frac{1}{T} \sum_{t=1}^{T} \sqrt{\frac{M+1}{M}} \frac{\mathrm{Spread}_{t,i}^{(v)}}{\mathrm{RMSE}_{t,i}^{\prime(v)}}. \tag{24}$$

The factor $\sqrt{(M+1)/M}$ accounts for the finite-ensemble correction. An SSR value close to 1 indicates agreement between the ensemble spread and the ensemble-mean forecast error. Values below 1 indicate underdispersion, whereas values above 1 indicate overdispersion (Fortin et al., 2014).

C Hyperparameter Settings

Model Architecture. The detailed architectural hyperparameters of the Hyan backbone, including its feature dimensions and number of layers, are summarized in Fig. 7.

Optimization. We use AdamW for one-dimensional parameters and Muon for two-dimensional parameters. The learning rates are shared between the two optimizers, while the weight decay is adjusted for each training stage. The main optimization settings are summarized in Table 3.

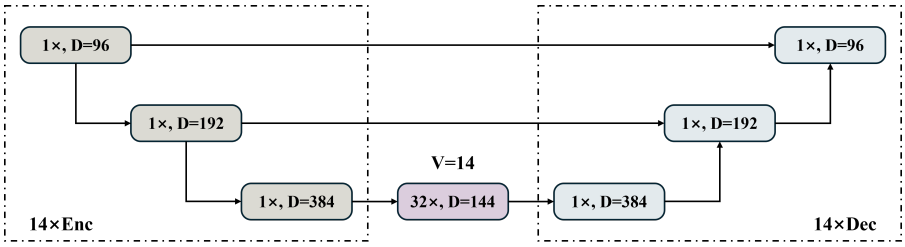


Figure 7: **Hyan architectural hyperparameters.** The diagram summarizes the number of layers and latent dimensions of the main network components.

Table 3: **Optimization settings.** The table summarizes the training hyperparameters used during pretraining and fine-tuning.

Configuration	Pretraining	Fine-tuning
Optimizer for 1D parameters	AdamW	AdamW
Optimizer for 2D parameters	Muon	Muon
Learning rate	1×10^{-4}	5×10^{-5}
Weight decay for 1D parameters	1×10^{-5}	1×10^{-5}
Weight decay for 2D parameters	1×10^{-5}	1×10^{-3}
Batch size	4	4
Optimization steps	18,000	3,500

D Training and Inference Budgets

Based on the information reported in Cachay et al. (2026), we summarize the corresponding training and inference budgets below:

- **ArchesWeatherGen:** Training requires approximately 45 V100 GPU-days. The inference time is approximately 3.5 seconds per forecast step on an A100 GPU.
- **NeuralGCM:** Training takes approximately 10 days on 128 TPU v5e devices. A 10-day forecast requires approximately 12.4 seconds on a single TPU v4 device.
- **GenCast:** Pretraining requires approximately 112 TPU v5-days, while fine-tuning requires approximately 48 TPU v5-days. A 15-day forecast takes approximately 8 minutes on a single device.
- **Hyan:** Training takes approximately 4 days, followed by 1.5 days of fine-tuning, on an NVIDIA RTX PRO 6000 GPU. The inference time is approximately 62.8 ms per forecast step.

Based on the conversion tables provided by Couairon et al. (2026); Cachay et al. (2026) and the theoretical peak computational capabilities of the respective accelerators, we obtain the following approximate equivalences. These estimates reflect theoretical compute capacity rather than measured end-to-end performance:

$$2\,\mathrm{V100} \approx 1\,\mathrm{A100}, \qquad 1\,\mathrm{TPU\ v4} \approx 1.5\,\mathrm{A100},$$
$$1\,\mathrm{TPU\ v5/v5e} \approx 3\,\mathrm{A100}, \qquad 1\,\mathrm{RTX\ PRO\ 6000} \approx 6\,\mathrm{A100}.$$

Using these approximate equivalences, we estimate the A100-equivalent training consumption and average single-step inference cost for each model. The converted results are shown in Fig. 1.

E Analysis of the Training Objective

Fix an input $\mathbf{x}$ and target $\mathbf{o}$. Let $\mathbf{y}$ and $\mathbf{y}'$ be independent draws from the same predictive distribution conditioned on $\mathbf{x}$, with finite expected weighted absolute error. Define

$$A = \mathbb{E}[d_w(\mathbf{y}, \mathbf{o})], \qquad B = \mathbb{E}[d_w(\mathbf{y}, \mathbf{y}')].$$

For $\alpha, \lambda > 0$, the expected objective is

$$\mathcal{L}_{\alpha,\lambda} = \alpha A - \lambda B.$$

Because the diversity term enters with a negative sign, we seek a condition under which the objective is bounded below.

By the triangle inequality,

$$d_w(\mathbf{y}, \mathbf{y}') \le d_w(\mathbf{y}, \mathbf{o}) + d_w(\mathbf{o}, \mathbf{y}').$$

Taking expectations gives $B \le 2A$. Consequently,

$$\mathcal{L}_{\alpha,\lambda} \ge (\alpha - 2\lambda)A. \tag{25}$$

Since $A \ge 0$, a sufficient condition for the objective to have a nonnegative lower bound is $\alpha \ge 2\lambda$.

The same condition applies to the finite-ensemble objective. For $M \ge 2$ ensemble members, summing the triangle inequality over ordered pairs yields

$$\sum_{m \ne j} d_w\left(\mathbf{y}^{(m)}, \mathbf{y}^{(j)}\right) \le 2(M-1) \sum_{m=1}^{M} d_w\left(\mathbf{y}^{(m)}, \mathbf{o}\right).$$

Substituting this inequality into Eq. 16 gives

$$\widehat{\mathcal{L}}_{\alpha,\lambda} \ge \frac{\alpha - 2\lambda}{M} \sum_{m=1}^{M} d_w\left(\mathbf{y}^{(m)}, \mathbf{o}\right). \tag{26}$$

Thus, $\alpha \ge 2\lambda$ ensures nonnegativity for both the expected objective and its finite-ensemble estimator.

For conditionally i.i.d. ensemble members, the sample averages of the prediction errors and off-diagonal pairwise distances are unbiased estimators of A and B, respectively. Therefore,

$$\mathbb{E}\left[\widehat{\mathcal{L}}_{\alpha,\lambda}\right] = \mathcal{L}_{\alpha,\lambda}. \tag{27}$$

To establish the connection to fair CRPS, write

$$d_w(\mathbf{a}, \mathbf{b}) = \sum_c w_c |a_c - b_c|, \qquad w_c \ge 0,$$

where c indexes scalar components and w_c incorporates the spatial and level weights and the normalization in Eq. 10. For component c, the fair CRPS estimator is

$$\begin{aligned} \widehat{\mathrm{CRPS}}_{\mathrm{fair},c} = \frac{1}{M} \sum_{m=1}^{M} |y_c^{(m)} - o_c| \\ - \frac{1}{2M(M-1)} \sum_{m \ne j} |y_c^{(m)} - y_c^{(j)}|. \end{aligned} \tag{28}$$

Substituting the componentwise decomposition of d_w into Eq. 16 gives, when $\alpha = 2\lambda$,

$$\widehat{\mathcal{L}}_{\alpha,\lambda} = \alpha \sum_c w_c \widehat{\mathrm{CRPS}}_{\mathrm{fair},c}. \tag{29}$$

In particular, setting $\alpha = 1$ and $\lambda = \frac{1}{2}$ recovers the weighted sum of componentwise fair CRPS scores exactly.

F Complete Results on WeatherBench2

Table 4: **Complete results on WeatherBench2.** Forecast performance is reported at all 10 lead times.

Variable	Model	Day 1			Day 2			Day 3			Day 4			Day 5			Day 6			Day 7			Day 8			Day 9			Day 10		
		CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR	CRPS	RMSE	SSR
$z500$	IFS ENS	22.38	42.55	0.9425	37.94	80.08	0.8963	58.25	132.20	0.9234	83.99	199.20	0.9568	114.90	277.46	0.9850	148.52	359.67	1.0034	182.07	439.83	1.0150	213.10	511.64	1.0199	239.97	572.14	1.0203	262.43	621.59	1.0195
	ArchesWG	21.15	41.91	1.0394	35.90	78.36	1.0969	55.80	130.08	1.0978	81.12	196.36	1.0839	110.79	272.39	1.0660	142.75	351.64	1.0494	174.79	428.73	1.0373	204.96	499.62	1.0300	231.74	560.75	1.0246	254.06	610.35	1.0189
	NeuralGCM	22.88	43.99	0.8818	36.22	77.10	0.8430	54.90	126.19	0.8591	79.29	190.66	0.8841	108.37	266.20	0.9078	140.46	345.86	0.9272	172.81	424.07	0.9451	203.53	495.48	0.9595	230.53	556.35	0.9689	253.46	606.84	0.9758
	GenCast	20.24	39.30	1.0387	34.87	74.17	1.1061	54.25	123.48	1.1189	78.91	187.27	1.1114	108.3	262.36	1.0992	140.27	342.04	1.0841	172.68	420.47	1.0699	203.83	493.80	1.0602	231.12	556.24	1.0487	253.63	606.52	1.0361
	Hyan	20.60	42.23	1.0682	35.04	78.62	1.0607	54.51	130.88	1.0529	79.20	197.80	1.0466	108.60	275.91	1.0350	140.93	359.22	1.0211	173.65	441.49	1.0050	204.78	517.03	0.9921	232.02	580.64	0.9868	254.68	632.44	0.9833
	Hyan DE	20.56	40.17	1.1353	34.95	74.65	1.1330	54.29	124.29	1.1276	78.87	188.10	1.1184	108.31	263.06	1.1019	140.53	343.14	1.0834	173.33	422.81	1.0631	204.53	496.05	1.0458	231.84	557.77	1.0371	254.57	607.82	1.0312
$q850$	IFS ENS	295.81	646.32	1.0200	397.90	857.63	1.0515	467.98	998.98	1.0323	529.09	1117.91	1.0207	586.08	1224.78	1.0130	638.03	1318.43	1.0069	684.33	1400.10	1.0029	724.34	1469.36	0.9996	757.53	1525.99	0.9966	783.69	1569.87	0.9930
	ArchesWG	294.14	629.53	0.9902	375.80	811.28	0.9947	440.49	948.84	0.9969	501.23	1071.19	1.0010	560.05	1184.19	1.0040	615.48	1287.05	1.0058	666.25	1378.95	1.0075	711.19	1458.46	1.0091	748.86	1523.98	1.0092	779.55	1576.50	1.0088
	NeuralGCM	329.70	666.73	1.3556	402.33	839.65	1.1725	461.62	970.44	1.0927	519.10	1089.03	1.0584	576.13	1200.64	1.0438	630.42	1302.82	1.0381	679.94	1393.25	1.0365	723.29	1470.75	1.0372	759.76	1534.84	1.0380	790.01	1587.38	1.0396
	GenCast	274.65	596.24	1.0561	371.38	809.48	1.0496	441.25	955.29	1.0415	503.69	1078.55	1.0377	563.52	1191.85	1.0365	619.21	1294.04	1.0342	669.64	1383.74	1.0308	714.10	1461.67	1.0277	751.43	1526.46	1.0237	781.90	1578.60	1.0196
	Hyan	299.56	655.74	0.8351	384.99	858.02	0.8965	450.02	1003.70	0.9312	510.13	1130.58	0.9480	568.29	1247.09	0.9545	622.96	1352.34	0.9570	672.74	1445.40	0.9574	717.58	1527.48	0.9550	755.09	1594.62	0.9540	785.67	1649.21	0.9527
	Hyan DE	299.16	637.25	0.8626	384.42	830.28	0.9309	448.94	968.23	0.9707	508.59	1088.34	0.9907	566.29	1199.23	0.9992	620.75	1300.07	1.0023	670.58	1389.96	1.0020	715.34	1469.39	0.9988	753.02	1534.88	0.9963	783.99	1588.31	0.9942
$u850$	IFS ENS	0.5937	1.1463	1.0171	0.8262	1.6278	1.0084	1.0511	2.1111	0.9991	1.2859	2.6185	1.0042	1.5312	3.1360	1.0141	1.7678	3.6175	1.0197	1.9845	4.0450	1.0235	2.1732	4.4066	1.0255	2.3270	4.6950	1.0258	2.4479	4.9168	1.0251
	ArchesWG	0.6023	1.1632	0.9948	0.7890	1.5645	1.0192	0.9882	2.0141	1.0260	1.2099	2.5080	1.0251	1.4443	3.0163	1.0233	1.6761	3.4970	1.0192	1.8936	3.9318	1.0165	2.0858	4.3015	1.0152	2.2465	4.5998	1.0141	2.3738	4.8300	1.0125
	NeuralGCM	0.6584	1.2388	1.2603	0.8278	1.6087	1.0923	1.0149	2.0345	1.0254	1.2281	2.5148	1.0007	1.4571	3.0139	0.9940	1.6857	3.4928	0.9949	1.9024	3.9284	1.0009	2.0928	4.2971	1.0060	2.2525	4.5930	1.0100	2.3828	4.8295	1.0140
	GenCast	0.5423	1.0464	1.0587	0.7552	1.4932	1.0778	0.9647	1.9556	1.0787	1.1905	2.4545	1.0727	1.4294	2.9698	1.0662	1.6646	3.4550	1.0560	1.8856	3.8976	1.0485	2.0836	4.2804	1.0431	2.2474	4.5864	1.0363	2.3776	4.8242	1.0303
	Hyan	0.5729	1.1450	0.9358	0.7691	1.5845	0.9823	0.9726	2.0618	1.0006	1.1954	2.5796	1.0044	1.4317	3.1107	0.9995	1.6674	3.6177	0.9924	1.8911	4.0820	0.9822	2.0886	4.4725	0.9745	2.2508	4.7836	0.9701	2.3796	5.0258	0.9663
	Hyan DE	0.5720	1.1048	0.9780	0.7675	1.5225	1.0318	0.9699	1.9779	1.0538	1.1920	2.4736	1.0577	1.4270	2.9827	1.0521	1.6616	3.4709	1.0426	1.8840	3.9185	1.0298	2.0825	4.3017	1.0190	2.2447	4.6018	1.0125	2.3751	4.8382	1.0071
$v850$	IFS ENS	0.6003	1.1665	1.0268	0.8262	1.6413	1.0128	1.0511	2.1213	1.0034	1.2859	2.6419	1.0117	1.5312	3.1667	1.0190	1.7678	3.6559	1.0242	1.9845	4.0907	1.0284	2.1732	4.4505	1.0307	2.3270	4.7215	1.0294	2.4479	4.9212	1.0270
	ArchesWG	0.6129	1.1895	0.9908	0.7890	1.5918	1.0152	0.9882	2.0446	1.0239	1.2099	2.5510	1.0272	1.4443	3.0710	1.0252	1.6761	3.5533	1.0188	1.8936	3.9870	1.0153	2.0858	4.3480	1.0132	2.2465	4.6303	1.0101	2.3738	4.8480	1.0090
	NeuralGCM	0.6619	1.2560	1.2480	0.8278	1.6224	1.0859	1.0149	2.0534	1.0222	1.2281	2.5471	0.9992	1.4571	3.0605	0.9932	1.6857	3.5513	0.9956	1.9024	3.9927	1.0025	2.0928	4.3582	1.0088	2.2525	4.6402	1.0120	2.3828	4.8523	1.0144
	GenCast	0.5517	1.0693	1.0631	0.7552	1.5172	1.0782	0.9647	1.9822	1.0797	1.1905	2.4916	1.0766	1.4294	3.0191	1.0709	1.6646	3.5160	1.0612	1.8856	3.9610	1.0529	2.0836	4.3370	1.0458	2.2474	4.6233	1.0362	2.3776	4.8433	1.0292
	Hyan	0.5775	1.1622	0.9479	0.7710	1.6074	0.9889	0.9709	2.0907	1.0035	1.1913	2.6178	1.0058	1.4245	3.1574	1.0013	1.6540	3.6704	0.9936	1.8685	4.1332	0.9841	2.0566	4.5236	0.9742	2.2037	4.8206	0.9711	2.3160	5.0429	0.9696
	Hyan DE	0.5767	1.1206	0.9910	0.7697	1.5440	1.0394	0.9688	2.0048	1.0579	1.1879	2.5090	1.0607	1.4212	3.0301	1.0539	1.6503	3.5250	1.0445	1.8641	3.9730	1.0324	2.0518	4.3515	1.0202	2.2002	4.6416	1.0150	2.3140	4.8586	1.0116
$t850$	IFS ENS	0.3646	0.6560	1.0743	0.4405	0.8973	1.1136	0.5395	1.1119	1.0847	0.6456	1.3507	1.0585	0.7640	1.6203	1.0439	0.8900	1.9047	1.0375	1.0167	2.1788	1.0343	1.1341	2.4229	1.0314	1.2381	2.6322	1.0295	1.3235	2.8005	1.0263
	ArchesWG	0.2966	0.5953	1.0158	0.3810	0.7774	1.0397	0.4701	0.9818	1.0487	0.5764	1.2320	1.0490	0.6994	1.5183	1.0461	0.8302	1.8138	1.0398	0.9615	2.0980	1.0339	1.0859	2.3559	1.0306	1.1961	2.5758	1.0280	1.2887	2.7549	1.0253
	NeuralGCM	0.3328	0.6578	1.2265	0.4137	0.8361	1.0779	0.4974	1.0271	1.0079	0.5975	1.2615	0.9785	0.7149	1.5361	0.9732	0.8418	1.8246	0.9763	0.9702	2.1039	0.9833	1.0914	2.3562	0.9911	1.2003	2.5759	0.9997	1.2930	2.7559	1.0062
	GenCast	0.2694	0.5416	1.0770	0.3656	0.7461	1.0971	0.4586	0.9549	1.0979	0.5652	1.2017	1.0906	0.6873	1.4845	1.0818	0.8189	1.7826	1.0730	0.9521	2.0721	1.0638	1.0769	2.3317	1.0543	1.1879	2.5560	1.0465	1.2807	2.7374	1.0385
	Hyan	0.2940	0.6068	0.9073	0.3822	0.8051	0.9383	0.4713	1.0177	0.9647	0.5762	1.2744	0.9802	0.6964	1.5657	0.9855	0.8268	1.8729	0.9833	0.9581	2.1695	0.9785	1.0840	2.4420	0.9703	1.1965	2.6756	0.9637	1.2905	2.8662	0.9597
	Hyan DE	0.2933	0.5860	0.9452	0.3805	0.7746	0.9841	0.4687	0.9761	1.0167	0.5725	1.2198	1.0357	0.6928	1.5005	1.0406	0.8226	1.7961	1.0366	0.9542	2.0843	1.0287	1.0797	2.3473	1.0193	1.1920	2.5737	1.0105	1.2859	2.7569	1.0048
$t2m$	IFS ENS	0.2713	0.5655	1.2346	0.3432	0.7215	1.2081	0.3996	0.8624	1.1482	0.4600	1.0228	1.1031	0.5266	1.2074	1.0764	0.5980	1.4011	1.0611	0.6687	1.5851	1.0501	0.7352	1.7504	1.0435	0.7938	1.8922	1.0392	0.8415	2.0066	1.0336
	ArchesWG	0.2445	0.5414	1.2443	0.3062	0.6814	1.2122	0.3662	0.8322	1.1765	0.4323	1.0069	1.1395	0.5061	1.2057	1.1110	0.5841	1.4134	1.0918	0.6600	1.6080	1.0761	0.7311	1.7808	1.0645	0.7940	1.9284	1.0569	0.8468	2.0485	1.0505
	NeuralGCM	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	GenCast	0.2086	0.4934	1.0559	0.2808	0.6499	1.0914	0.3425	0.8010	1.1014	0.4083	0.9717	1.0928	0.4820	1.1672	1.0819	0.5608	1.3758	1.0733	0.6386	1.5757	1.0640	0.7107	1.7505	1.0531	0.7747	1.9010	1.0441	0.8275	2.0222	1.0347
	Hyan	0.2581	0.5806	0.8180	0.3246	0.7331	0.8429	0.3824	0.8817	0.8796	0.4455	1.0544	0.9139	0.5163	1.2545	0.9350	0.5929	1.4681	0.9426	0.6683	1.6694	0.9464	0.7398	1.8508	0.9460	0.8027	2.0059	0.9464	0.8549	2.1323	0.9457
	Hyan DE	0.2569	0.5640	0.8516	0.3224	0.7100	0.8813	0.3793	0.8511	0.9233	0.4414	1.0147	0.9632	0.5118	1.2058	0.9868	0.5878	1.4104	0.9952	0.6634	1.6065	0.9969	0.7344	1.7812	0.9968	0.7969	1.9304	0.9957	0.8492	2.0517	0.9944
$u10$	IFS ENS	0.4055	0.8111	1.0048	0.5431	1.1310	0.9815	0.6861	1.4701	0.9788	0.8426	1.8324	0.9905	1.0074	2.1988	1.0039	1.1661	2.5333	1.0117	1.3099	2.8260	1.0166	1.4338	3.0679	1.0185	1.5335	3.2567	1.0188	1.6108	3.3999	1.0182
	ArchesWG	0.3702	0.7431	1.0000	0.4913	1.0356	1.0290	0.6248	1.3660	1.0333	0.7749	1.7224	1.0279	0.9348	2.0847	1.0242	1.0909	2.4193	1.0182	1.2357	2.7169	1.0152	1.3614	2.9633	1.0133	1.4645	3.1579	1.0116	1.5458	3.3064	1.0098
	NeuralGCM	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	GenCast	0.3316	0.6659	1.0496	0.4661	0.9814	1.0761	0.6044	1.3176	1.0782	0.7574	1.6782	1.0701	0.9214	2.0489	1.0638	1.0808	2.3897	1.0534	1.2293	2.6965	1.0466	1.3598	2.9533	1.0410	1.4648	3.1517	1.0332	1.5474	3.3030	1.0263
	Hyan	0.3536	0.7379	0.9725	0.4791	1.0517	1.0040	0.6136	1.3984	1.0152	0.7641	1.7712	1.0166	0.9260	2.1514	1.0082	1.0857	2.5066	0.9984	1.2347	2.8247	0.9873	1.3640	3.0860	0.9793	1.4667	3.2857	0.9760	1.5483	3.4419	0.9725
	Hyan DE	0.3530	0.7097	1.0181	0.4779	1.0086	1.0550	0.6121	1.3400	1.0687	0.7618	1.6972	1.0702	0.9230	2.0625	1.0581	1.0819	2.4047	1.0468	1.2303	2.7108	1.0333	1.3604	2.9676	1.0221	1.4638	3.1614	1.0171	1.5460	3.3124	1.0123
$v10$	IFS ENS	0.4170	0.8408	1.0165	0.5596	1.1693	0.9846	0.7068	1.5164	0.9804	0.8701	1.8950	0.9934	1.0435	2.2793	1.0052	1.2121	2.6375	1.0124	1.3672	2.9544	1.0163	1.5020	3.2216	1.0197	1.6069	3.4235	1.0194	1.6850	3.5712	1.0172
	ArchesWG	0.3871	0.7753	0.9945	0.5124	1.0776	1.0229	0.6501	1.4157	1.0293	0.8073	1.7902	1.0298	0.9775	2.1753	1.0278	1.1433	2.5312	1.0214	1.2956	2.8460	1.0156	1.4309	3.1151	1.0143	1.5409	3.3252	1.0124	1.6264	3.4846	1.0113
	NeuralGCM	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	GenCast	0.3476	0.6968	1.0449	0.4863	1.0225	1.0685	0.6290	1.3670	1.0735	0.7890	1.7444	1.0714	0.9607	2.1332	1.0664	1.1308	2.4990	1.0578	1.2881	2.8244	1.0489	1.4287	3.1045	1.0433	1.5415	3.3190	1.0355	1.6270	3.4793	1.0282
	Hyan	0.3664	0.7656	0.9869	0.4969	1.0928	1.0126	0.6373	1.4505	1.0189	0.7951	1.8401	1.0160	0.9652	2.2383	1.0082	1.1341	2.6151	0.9992	1.2917	2.9536	0.9899	1.4328	3.2419	0.9786	1.5444	3.4640	0.9741	1.6279	3.6257	0.9728
	Hyan DE	0.3660	0.7360	1.0339	0.4962	1.0478	1.0651	0.6357	1.3895	1.0736	0.7931	1.7629	1.0702	0.9628	2.1475	1.0595	1.1309	2.5101	1.0488	1.2882	2.8372	1.0371	1.4291	3.1178	1.0237	1.5414	3.3331	1.0168	1.6263	3.4916	1.0132
$mslp$	IFS ENS	31.62	62.58	1.0200	47.27	98.60	0.9487	65.94	146.22	0.9372	89.92	208.27	0.9577	118.65	280.52	0.9977	149.19	354.03	0.9827	178.69	423.91	1.0069	205.56	485.29	1.0112	228.43	536.23	1.0123	246.76	575.97	1.0100
	ArchesWG	25.82	51.70	1.0424	40.25	87.19	1.0983	59.37	137.11	1.0913	83.70	200.56	1.0754	111.97	272.41	1.0611	141.75	344.85	1.0455	170.68	413.20	1.0337	197.34	474.39	1.0282	220.65	526.19	1.0245	239.52	567.14	1.0204
	NeuralGCM	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	GenCast	24.55	48.56	1.0548	39.12	83.34	1.1102	57.71	131.16	1.1161	81.47	192.66	1.1073	109.61	264.07	1.0967	139.42	336.95	1.0794	168.86	406.35	1.0636	196.51	469.62	1.0542	220.17	522.12	1.0431	239.01	563.93	1.0324
	Hyan	25.37	53.22	1.0539	39.49	88.83	1.0535	58.11	139.39	1.0529	81.76	203.43	1.0490	109.80	277.04	1.0353	139.95	353.52	1.0203	169.81	426.73	1.0047	197.52	492.33	0.9910	221.01	546.04	0.9864	240.05	588.96	0.9814
	Hyan DE	25.29	50.75	1.1166	39.39	84.68	1.1192	57.92	132.88	1.1198	81.51	194.12	1.1133	109.49	264.78	1.0959	139.44	338.15	1.0782	169.26	409.00	1.0580	197.04	472.80	1.0405	220.63	524.97	1.0323	239.77	566.31	1.0258

Under review as a conference paper at ICLR 2027

G QUALITATIVE ANALYSIS

For the qualitative analysis, we conduct a case study using the forecast initialized at 00:00 UTC on November 1, 2020. Figures 8–11 show the predicted fields, prediction errors, and corresponding spherical-harmonic angular power spectra for $t2m$, $u10$, $z500$, and $q850$, respectively. Across the four variables, the predictions capture the principal spatial structures of the ground-truth fields. The angular power spectra of the predictions remain close to those of the ground truth over the main resolved scales, suggesting that the model reproduces the overall scale-dependent structure of the atmospheric fields.

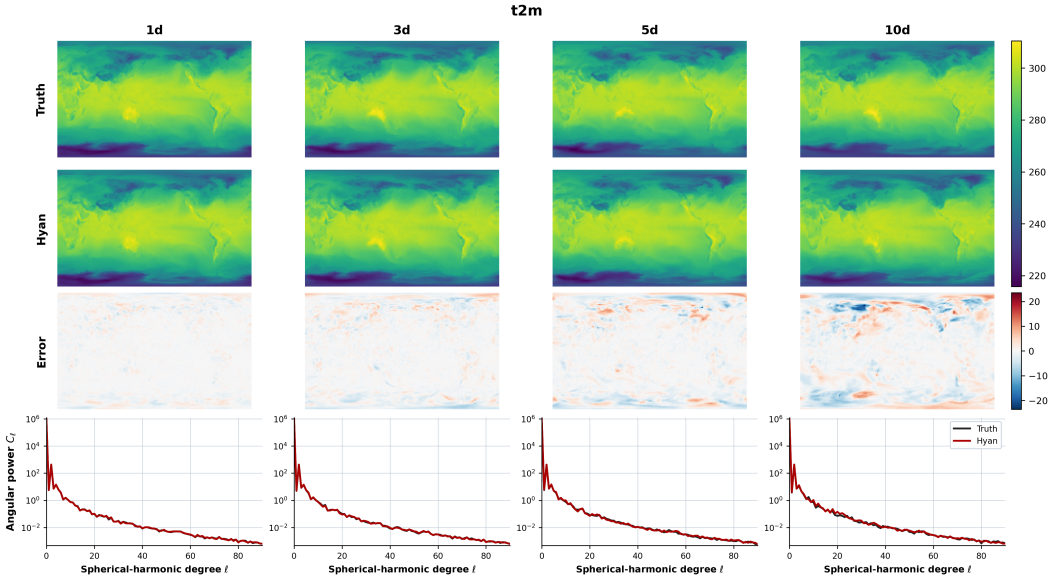


Figure 8: **Qualitative results for $t2m$.** Rows show the ground truth, Hyan forecasts, forecast errors, and spherical-harmonic angular power spectra for the forecast initialized at 00:00 UTC on November 1, 2020.

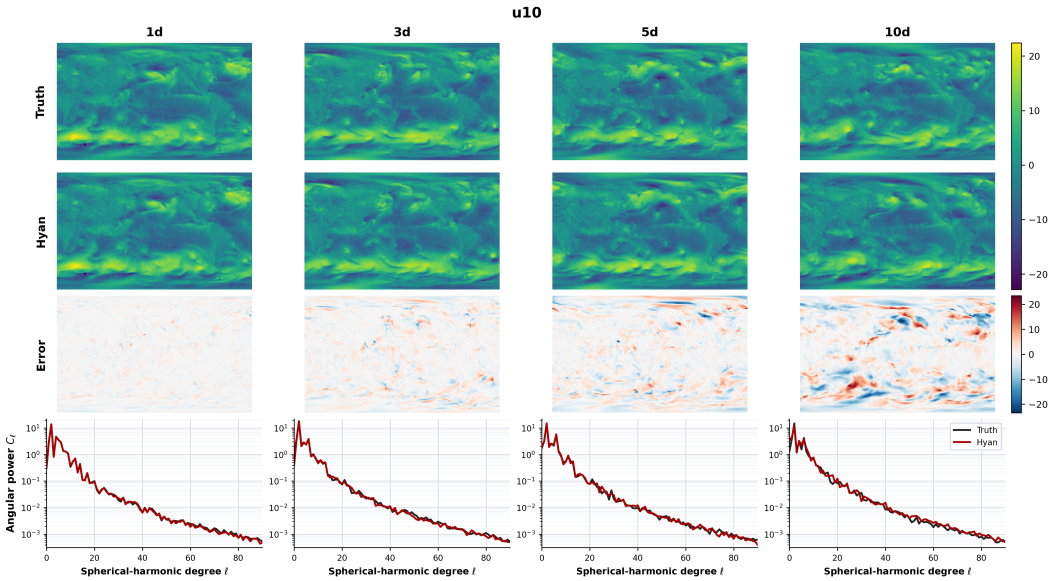


Figure 9: **Qualitative results for $u10$.** Same initialization and row layout as in Fig. 8.

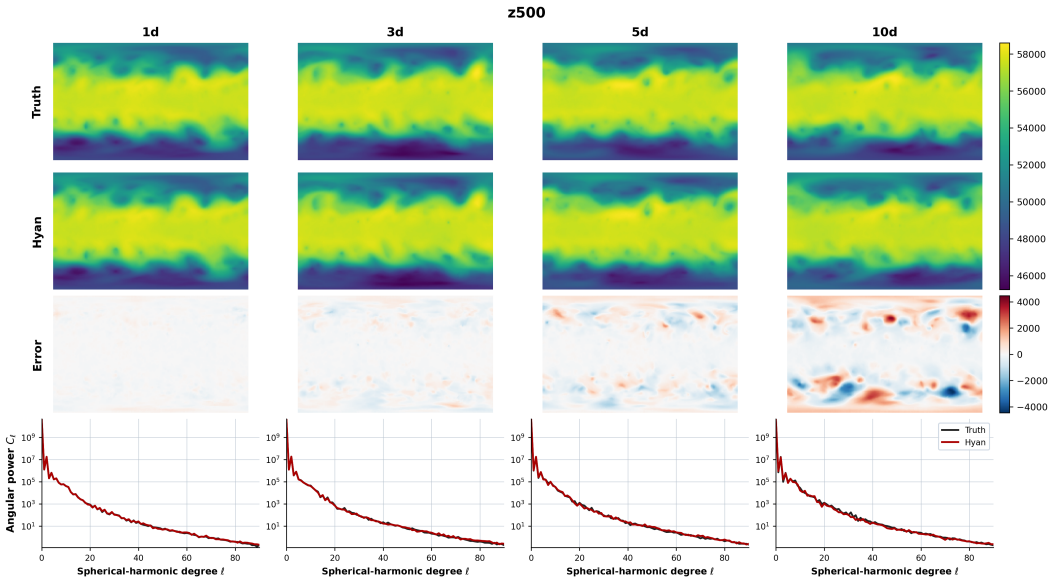


Figure 10: **Qualitative results for** $z500$**.** Same initialization and row layout as in Fig. 8.

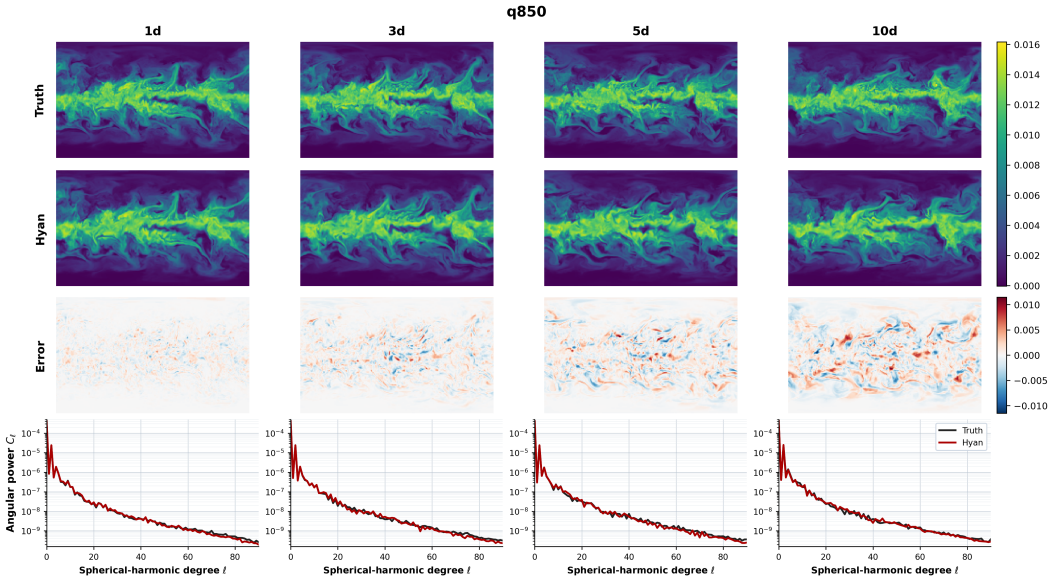


Figure 11: **Qualitative results for** $q850$**.** Same initialization and row layout as in Fig. 8.

H Additional Perturbation Intensity Visualizations

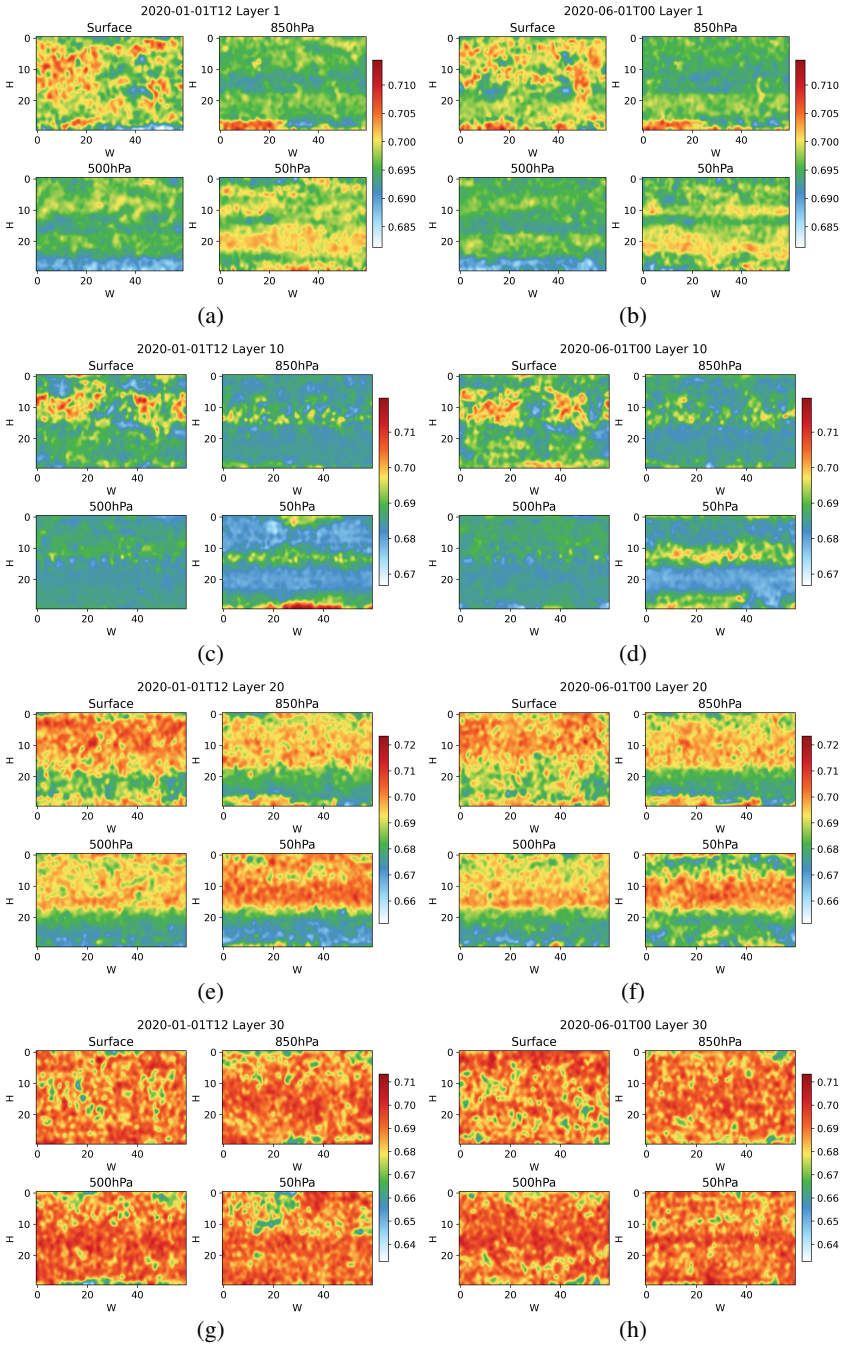


Figure 12: **Additional visualizations of perturbation intensity.** Spatial distributions are shown for different layers.