Probabilistic Weather Forecasting with Hierarchical Graph Neural Networks

Joel Oskarsson
Linköping University
joel.oskarsson@liu.se

Tomas Landelius
Swedish Meteorological and
Hydrological Institute
tomas.landelius@smhi.se

Marc Peter Deisenroth
University College London
m.deisenroth@ucl.ac.uk

Fredrik Lindsten
Linköping University
fredrik.lindsten@liu.se

Abstract

In recent years, machine learning has established itself as a powerful tool for high-resolution weather forecasting. While most current machine learning models focus on deterministic forecasts, accurately capturing the uncertainty in the chaotic weather system calls for probabilistic modeling. We propose a probabilistic weather forecasting model called Graph-EFM, combining a flexible latent-variable formulation with the successful graph-based forecasting framework. The use of a hierarchical graph construction allows for efficient sampling of spatially coherent forecasts. Requiring only a single forward pass per time step, Graph-EFM allows for fast generation of arbitrarily large ensembles. We experiment with the model on both global and limited area forecasting. Ensemble forecasts from Graph-EFM achieve equivalent or lower errors than comparable deterministic models, with the added benefit of accurately capturing forecast uncertainty.

1 Introduction

Forecasting the dynamics of Earth's atmosphere is a scientific problem of utmost importance. Society is dependent on fast and informative weather forecasts for planning in areas such as transportation and agriculture and for balancing the energy system [3]. Especially important is the use of forecasts to issue warnings for extreme weather events [1]. Recent advances in Machine-Learning-based Weather Prediction (MLWP) have enabled models that produce accurate forecasts in a fraction of the time of traditional physics-based systems [37, 4, 23]. So far these developments have largely been focused on deterministic modeling. However, forecasting only one likely weather scenario ignores the many uncertainties in predicting future weather.

Weather is a chaotic system, resulting in high forecast uncertainty [52]. This uncertainty comes from both imperfect representations of initial states and inaccurate descriptions of the function mapping from one time step to the next [26]. Accurately modeling this uncertainty significantly increases the value of weather forecasts. Such uncertainty can be communicated to end-users to improve decision making or be used in downstream products, for example to compute a distribution over solar power generation. Capturing the full forecast uncertainty requires us to predict not just a single likely state trajectory, but a collection of possible future weather states. Due to the complexity and dimensionality of the weather system the feasible way to achieve this is by generating samples from a modeled distribution. Such *ensemble forecasting* is today performed using physics-based methods, where a number of *ensemble members* are simulated as samples from this distribution. The computational cost of this is however massive, often limiting the spatial resolution or size of the ensemble [3].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

MLWP is a promising approach for addressing this limitation and enabling large ensemble forecasts. However, for the ensemble to add value the machine learning model needs to accurately represent the distribution. Initial attempts at MLWP ensemble forecasting either rely on ad-hoc initial state perturbations [10, 37, 4] or have not been scaled to spatial resolutions of interest [19]. Also diffusion models [18] have been applied to the problem, but sampling forecasts from these is computationally expensive and can be prohibitively slow [39]. We propose a Graph-based Ensemble Forecasting Model (Graph-EFM), enabling efficient sampling of ensemble members with only one forward-pass per time step. The method builds on graph-based MLWP [20, 23], which is a flexible framework that can be adapted to different geometries and state grid representations [24]. By combining a latent-variable formulation with a hierarchical Graph Neural Network (GNN) the distribution is modeled in a lower-dimensional space and sampled forecasts are spatially coherent.

MLWP models are typically trained for and evaluated on global weather forecasting [40, 23, 4]. Another common forecasting setup in practice is the use of Limited Area Models (LAMs) to produce high-resolution regional forecasts [11]. Such LAMs are for example used by local weather services in order to provide forecasts tailored to the geographical properties and societal needs of the region [38, 44, 7, 32]. These high-resolution models are also invaluable to various industrial sectors, including energy forecasters, who rely on precise weather predictions to manage supply and demand. This motivates research into also constructing MLWP LAMs, which brings new challenges related to the high resolution and boundary conditions of the limited area. In this work we experiment not just with global forecasting, but consider also how probabilistic LAMs can be trained to produce forecasts for the Nordic region.

Our main contributions are: 1) We develop a hierarchical GNN framework for both deterministic and probabilistic MLWP. The hierarchical construction encourages spatially coherent fields in forecasts. 2) We use this framework to define the probabilistic weather forecasting model Graph-EFM, capable of efficient sampling of arbitrarily large ensemble forecasts. 3) We develop a training method targeting both forecast quality and ensemble calibration. 4) We experiment with both global forecasting on 1.5° resolution and a novel limited-area modeling task at 10 km resolution.

2 Related Work

Deterministic MLWP Multiple machine learning methods have been successfully applied to large-scale weather forecasting. These include graph-based models [20, 23, 24], transformers [4, 8, 10, 33, 25, 34] and neural operators [37, 5]. While large neural network models learn weather dynamics purely from data, there are also parallel developments in building hybrid physics-MLWP models [22, 50].

Ensembles from perturbations Most existing methods for MLWP ensemble forecasting follow closely the physics-based methods, where initial states and model parameters are *perturbed* to create ensemble diversity. A number of MLWP works create ensembles by ad-hoc perturbing initial states with random noise [10, 37, 4, 16, 6]. More informed perturbations have been re-used from physics based ensembles [39, 6] and created based on model-informed singular vectors [43]. Others try to perturb the forecast model itself, rolling out ensemble members using different neural network parameters [51, 43]. Such multi-model approaches require training, or at least fine-tuning, a pre-defined number of MLWP models. Graubner et al. [16] use the SWAG method [30] to allow for constructing multi-model ensembles of arbitrary size.

Generative modeling Probabilistic machine learning approaches aim to directly learn generative models producing ensemble members. Similar to our approach, the SwinVRNN model [19] uses a latent variable formulation, but combined with a Swin Transformer architecture [29]. SwinVRNN is developed for global forecasting at 5° resolution and scales poorly to higher spatial resolutions. Also building on the graph-based framework, Price et al. [39] train a diffusion model [18, 46] to sample each time step. Their Gencast model produces ensemble forecasts of 0.25° global data with 12 h time steps. Diffusion models produce realistic-looking samples, but typically require solving an ordinary differential equation involving multiple passes through the neural network to sample each time step. For GenCast, this results in a sampling time of 8 minutes for a single 15 day forecast on a TPUv5 device [39]. Other works use diffusion models to increase the size of physics-based ensembles [27] or stochastically downscale deterministic forecasts [9, 31].

Hierarchical GNNs Motivated by capturing multiple spatial scales, hierarchical GNNs have been used for modeling general partial differential equations [14, 28]. The overall hierarchical framework shares much of its structure with the popular U-Net architecture [41] for computer vision tasks, but extended to a general graph setting.

3 Background

3.1 Problem Definition

The weather forecasting problem can be summarized as mapping from a set of initial states $X^{-1:0} = (X^{-1}, X^0)$ to the sequence of future states $X^{1:T} = (X^1, \dots, X^T)$. A table of notation is provided in appendix A. Each weather state $X^t \in \mathbb{R}^{N \times d_x}$ here contains d_x variables modeled at N different locations. Geospatial data is often represented as regular grids, in which case these locations correspond to the grid cells. The d_x variables can include both atmospheric variables, modeled at multiple vertical levels, and surface variables. As is common in MLWP we assume the initial states to consist of two time steps, which allows for capturing first-order state dynamics. To produce a forecast, a set of forcing inputs $F^{1:T}$ are also available. These contain known quantities, such as the time of day. There are also static features associated with the grid cells, such as the orography, which we here consider part of the forcing.

Many variables impact the chaotic weather system, all of which are not fully captured in initial states represented on finite grids. This induces forecast uncertainty, which we view as a distribution $p\big(X^{1:T} \big| X^{-1:0}, F^{1:T}\big)$. In deterministic forecasting we seek a model that minimizes the Mean Squared Error (MSE) to the future weather states [23, 33, 34]. This is equivalent to modeling only the mean of the distribution. In probabilistic forecasting we instead aim to model the full distribution. Note that we here specifically model the *conditional* distribution $p\big(X^{1:T} \big| X^{-1:0}, F^{1:T}\big)$, rather than $p\big(X^{1:T} \big| F^{1:T}\big)$. Hence we do not marginalize over uncertainty in initial states.

3.2 Graph-based Weather Forecasting

Graph-based MLWP models use an autoregressive mapping $\hat{X}^t = f(X^{t-2:t-1}, F^t)$ consisting of a sequence of GNNs [20, 23, 24]. Starting from the initial states, this mapping can be iteratively applied to roll out a full forecast $X^{1:T}$. Central to the graph-based framework is the idea of mapping from the original N grid locations to a *mesh graph* $\mathcal{G}_M = (\mathcal{V}_M, \mathcal{E}_M)$. In the graph-context we refer to the grid locations as a set $\mathcal{V}_G$ of *grid nodes*. By choosing $|\mathcal{V}_M| < |\mathcal{V}_G| = N$ it becomes efficient to perform the majority of computations on the mesh. Such a mesh graph can also be tailored to the forecasting setting, for example to respect the spherical geometry in global forecasting [20]. The mapping f realizes a single-step prediction by passing $X^{t-2:t-1}$ and F^t through a series of GNN layers. In sequence, these layers: 1) map grid inputs to representations on the mesh graph; 2) perform a number of processing steps on the mesh; 3) map back to the grid to produce the prediction for X^t. Steps 1 and 3 use bipartite graphs $\mathcal{G}_{\text{G2M}} = (\mathcal{V}_G \cup \mathcal{V}_M, \mathcal{E}_{\text{G2M}})$ and $\mathcal{G}_{\text{M2G}} = (\mathcal{V}_G \cup \mathcal{V}_M, \mathcal{E}_{\text{M2G}})$ with edges connecting the grid and mesh nodes. The GNN layers in each step compute updates for node representations $H \in \mathbb{R}^{|\mathcal{V}| \times d_z}$ and edge representations $E \in \mathbb{R}^{|\mathcal{E}| \times d_z}$ in the graphs. For simplicity all representation vectors have dimensionality d_z.

Interaction Networks The specific GNN layers used in previous works are *Interaction Networks* [2, 23]. The layers in these networks pass messages from a set of sender nodes along directed graph edges to a set of receiver nodes. Based on these messages the edge and receiver node representations are then updated. For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ let $\boldsymbol{e}_{\alpha \to \beta} \in \mathbb{R}^{d_z}$ be the row of E corresponding to the edge $(\alpha, \beta) \in \mathcal{E}$. Let H^S be the matrix with rows containing sender node representations and H^R the corresponding matrix for receiver nodes. Interaction Networks then implement the representation update $H^R, E \leftarrow \text{GNN}(\mathcal{G}, H^S, E, H^R)$ as

$$\tilde{\boldsymbol{e}}_{\alpha \to \beta} \leftarrow \text{MLP}\big(\boldsymbol{e}_{\alpha \to \beta}, H^S_\alpha, H^R_\beta\big) \tag{1a}$$

$$\boldsymbol{e}_{\alpha \to \beta} \leftarrow \boldsymbol{e}_{\alpha \to \beta} + \tilde{\boldsymbol{e}}_{\alpha \to \beta} \qquad H^R_\beta \leftarrow H^R_\beta + \text{MLP}\Big(H^R_\beta, \textstyle\sum_{\alpha \in \text{Ne}(\beta)} \tilde{\boldsymbol{e}}_{\alpha \to \beta}\Big) \tag{1b}$$

where $\text{Ne}(\beta) = \{\alpha : (\alpha, \beta) \in \mathcal{E}\}$ are the incoming neighbors of node β. Parameters in Multi-Layer Perceptrons (MLPs) are shared across nodes and edges in the graph, but not between GNN layers.

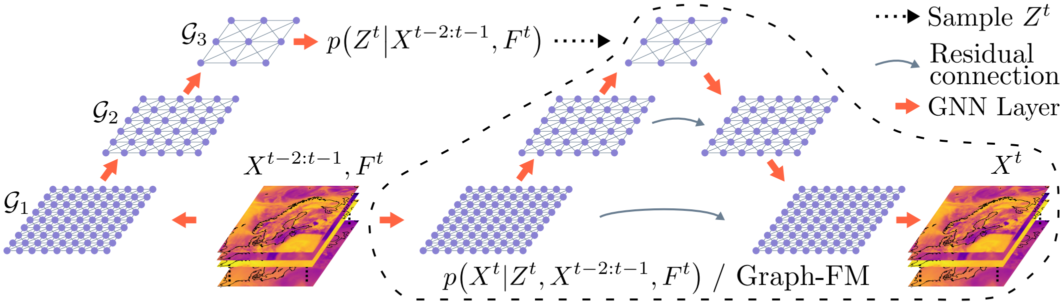


Figure 1: Overview of our Graph-EFM model, with example data and graphs for a Limited Area Model. The corresponding overview for the global setting is given in fig. 6 in appendix C.

Global mesh graphs Keisler [20] proposed to construct a mesh graph for global MLWP as an icosahedral grid covering the globe. This approach was extended in the GraphCast model [23] by introducing a multi-scale mesh graph with edges of varying length. Such multi-scale edges are capable of propagating information and capturing statistical dependencies both locally and over long distances in the graph. The multi-scale mesh graph is created by sequentially splitting the faces of an icosahedron into a sequence of graphs $\mathcal{G}_L, \dots, \mathcal{G}_1$ with node sets satisfying $\mathcal{V}_L \subset \cdots \subset \mathcal{V}_1$ by construction. The original icosahedron $\mathcal{G}_L$ has the longest edges $\mathcal{E}_L$, stretching far across the globe, whereas the final graph $\mathcal{G}_1$ has short edges $\mathcal{E}_1$ only connecting nodes locally. The final multi-scale mesh graph is constructed as $\mathcal{G}_{\text{MS}} = (\mathcal{V}_1, \mathcal{E}_L \cup \cdots \cup \mathcal{E}_1)$, taking the nodes from the final graph but connecting these using edges of all different lengths [23].

4 Weather Forecasting with Hierarchical Graph Neural Networks

Two great challenges in weather forecasting is to accurately capture processes unfolding over different spatial scales and modeling the uncertainty in the chaotic system [52]. To tackle these challenges, we propose to construct a hierarchical mesh graph, working with different length scales at each level in the hierarchy. We use a sequence $\mathcal{G}_1, \dots, \mathcal{G}_L$ of graphs as the different levels in the hierarchy, additionally adding connections between the nodes of adjacent levels. This construction is also highly suitable as a basis for building probabilistic forecasting models, as discussed below. Figure 1 shows an overview of the hierarchical mesh used in our model. See figs. 12 and 14 in the appendix for illustrations of how this differs from the multi-scale graph.

There are multiple benefits to such a hierarchical mesh construction for MLWP. By keeping the graphs at different levels separate, we can define GNN layers with independent parametrizations at each level. This adds flexibility by allowing the model to learn different representation updates for edges of different spatial scales. A hierarchical mesh graph also offers a natural, spatially-aware dimensionality reduction, as the state in the grid is encoded into a few nodes at the top level. Such a representation can capture the general structure of each weather state, with finer details added as this is propagated down through the hierarchy. We leverage this property to construct a probabilistic model by imposing a distribution over these lower-dimensional representations at the top level. This allows for efficiently drawing spatially coherent samples from the distribution of future weather states.

4.1 Hierarchical Graph

Our hierarchical mesh graph consists of L graph levels $\mathcal{G}_1, \dots, \mathcal{G}_L$ with $\mathcal{G}_l = (\mathcal{V}_l, \mathcal{E}_l)$. Only level 1 of the hierarchy is connected to the grid, so we re-define $\mathcal{G}_{\text{G2M}} = (\mathcal{V}_G \cup \mathcal{V}_1, \mathcal{E}_{\text{G2M}})$ and $\mathcal{G}_{\text{M2G}} = (\mathcal{V}_G \cup \mathcal{V}_1, \mathcal{E}_{\text{M2G}})$. The number of nodes $|\mathcal{V}_l|$ decreases with the level l. The smallest set of nodes are found at the top level L.

To pass information between the levels of the hierarchy we introduce additional graphs connecting the different levels. Let $\mathcal{G}_{l,l+1} = (\mathcal{V}_l \cup \mathcal{V}_{l+1}, \mathcal{E}_{l,l+1})$ be a graph containing directed edges from mesh level l to level $l+1$. We make use of a graph sequence $\mathcal{G}_{1,2}, \dots, \mathcal{G}_{L-1,L}$ to propagate information up through the hierarchy and similarly a sequence $\mathcal{G}_{L,L-1}, \dots, \mathcal{G}_{2,1}$ in the downward direction. The exact layout of nodes and edges at and in-between levels are design choices that should be tailored to the specific forecasting setting. Examples for global and limited-area forecasting are given in section 5.

4.2 Graph-FM: Deterministic Forecasting

The hierarchical graph allows for defining GNN layers both on and in-between the different levels. By sequentially updating node and edge representations at different levels in the hierarchy, information can be propagated up from the grid to the different levels. As these levels have edges of different lengths, the processing at each level happens on different spatial scales. Note that this differs from the multi-scale graph approach, where information processing over all different spatial scales happen in the same GNN layer [23]. As a step towards our probabilistic model, we define an alternative deterministic Graph-based Forecasting Model *Graph-FM*[1], operating on the hierarchical graph.

In Graph-FM one processing step on the mesh graph is defined as a complete sweep through the hierarchy. GNNs are applied sequentially to the inter-level and intra-level graphs in the order $\mathcal{G}_1, \mathcal{G}_{1,2}, \mathcal{G}_2, \dots, \mathcal{G}_{L-1,L}, \mathcal{G}_L$, updating edge and node representations at the different levels. Processing steps going up the hierarchy are alternated with similar steps going down from level L to 1. The single step mapping f consists of multiple such sweeps up and down (see appendix C.2).

4.3 Graph-EFM: Probabilistic Forecasting

To capture the uncertainty in the chaotic weather system we next aim to construct a probabilistic model from the ground up to capture the full distribution $p\big(X^{1:T}|X^{-1:0}, F^{1:T}\big)$. We start by assuming the weather system to satisfy a second-order Markov assumption, decomposing

$$p\big(X^{1:T}|X^{-1:0}, F^{1:T}\big) = \textstyle\prod_{t=1}^{T} p\big(X^t \big| X^{t-2:t-1}, F^t\big). \tag{2}$$

Factoring the distribution over time steps allows us to work with forecasts of varying length. Specifying the model for single-step prediction avoids having to learn separate parameters for different lead times. Next, we seek a flexible, but computationally efficient parametrization for the distribution $p\big(X^t \big| X^{t-2:t-1}, F^t\big)$. This can be achieved by introducing a latent random variable Z^t, and letting

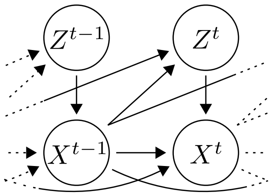


Figure 2: Graphical model for eq. (3).

$$p\big(X^t \big| X^{t-2:t-1}, F^t\big) = \int p\big(X^t | Z^t, X^{t-2:t-1}, F^t\big) p\big(Z^t \big| X^{t-2:t-1}, F^t\big) dZ^t. \tag{3}$$

Here the stochasticity in Z^t should capture the uncertainty over X^t at each time step. The corresponding graphical model is shown in fig. 2. We impose a spatial structure over the latent variable by letting Z^t be $|\mathcal{V}_L| \times d_z$ matrix-valued, with each row a d_z-dimensional vector associated with one node in the top level $\mathcal{G}_L$ of the mesh graph.

The single-step model consists of two components, a latent map $p\big(Z^t \big| X^{t-2:t-1}, F^t\big)$ and predictor $p\big(X^t | Z^t, X^{t-2:t-1}, F^t\big)$. The latent map is parametrized using GNNs, mapping the conditioning variables to parameters of a Gaussian distribution. We consider the predictor to be concentrated around its mean, and realize $p\big(X^t | Z^t, X^{t-2:t-1}, F^t\big)$ as a deterministic mapping of a similar form as Graph-FM. By sampling Z^t and passing this through the predictor we can draw a sample of X^t from eq. (3). This sample can then be conditioned on at the next time step, continuing this sampling process to roll out a forecast following eq. (2). This forecast constitutes one ensemble member, and the process can be repeated to sample an ensemble of arbitrary size. We call our Graph-based Ensemble Forecasting Model *Graph-EFM*. Full details about the model are given in appendix C.

Latent map We let the latent map be an isotropic Gaussian

$$p\big(Z^t \big| X^{t-2:t-1}, F^t\big) = \textstyle\prod_{\alpha \in \mathcal{V}_L} \mathcal{N}\big(Z^t_\alpha \big| \mu_Z \big(X^{t-2:t-1}, F^t\big)_\alpha, I\big) \tag{4}$$

with the mean as a function of the conditioning variables. The variance is fixed, imposing a fixed scale for the learned latent space. The mean function μ_Z consists of a sequence of GNNs. These take the inputs at the grid, propagate representations up through the hierarchical mesh graph, and finally predicts the mean of Z^t_α at each node α at level L. In appendix L.2 we verify empirically the importance of using the latent map over a static distribution for Z^t.

[1] The deterministic Graph-FM model was first proposed in a preliminary version of this work [35], but there only for the LAM setting under the name *Hi-LAM*.

Predictor The predictor is a deterministic mapping

$$\hat{X}^t = g\big(Z^t, X^{t-2:t-1}, F^t\big) = X^{t-1} + \tilde{g}\big(Z^t, X^{t-2:t-1}, F^t\big). \tag{5}$$

With the small time steps used in MLWP, X^t does not change dramatically in a single step. We thus follow the common practice of including a skip connection to the previous state [23, 5, 19]. The predictor takes both inputs $X^{t-2:t-1}, F^t$ at the grid and Z^t at the top of the mesh graph. To incorporate both we design g similar to Graph-FM, performing sweeps up and down through the mesh hierarchy. At the top of the hierarchy Z^t is added to node representations H^L through the residual connections in the GNN layers. A sampled value of Z^t then affects the prediction $\hat{X}^t$ through the downward sweep. While multiple such sweeps are possible, we found one to be sufficient in practice.

Spatial dependencies We want each sample of X^t to contain spatially coherent atmospheric fields. One approach would be to impose spatial dependencies in the joint distribution over Z^t. However, learning and sampling from such a distribution typically comes with computational challenges [19]. Instead, we impose spatial dependencies by integrating the latent variable formulation with the hierarchical graph. We argue that as the independent components of Z^t are propagated down through the mesh graph, gradually increasing the spatial resolution, spatial dependencies are introduced by the model in the GNN layers. The hierarchical graph is key to this property, as the stochasticity in Z^t is necessarily spread out over the forecast region, rather than only affecting the output locally.

4.4 Training Objective

Deterministic forecasting models can be straightforwardly trained by minimizing a weighted MSE [23] or Negative Log-Likelihood (NLL) loss [8] for rolled out forecasts. To train Graph-EFM we instead leverage the fact that the single-step model has a structure similar to a (conditional) Variational AutoEncoder (VAE) [21, 45], allowing us to use a variational objective. We introduce a variational approximation $q\big(Z^t \big| X^{t-2:t-1}, X^t, F^t\big)$ at each time step, approximating the true posterior $p\big(Z^t \big| X^{t-2:t-1}, X^t, F^t\big)$ over Z^t. This variational distribution is parametrized in a similar way as the latent map, with GNN layers mapping to a Gaussian over Z^t. Note however that q also depends on X^t, since it approximates the posterior. Using q, we can then define

$$\begin{aligned}\mathcal{L}_{\text{Var}}\big(X^{t-2:t-1}, X^t, F^t\big) = \lambda_{\text{KL}} D_{\text{KL}}\big(q\big(Z^t \big| X^{t-2:t-1}, X^t, F^t\big) \big\| p\big(Z^t \big| X^{t-2:t-1}, F^t\big)\big) \\ -\mathbb{E}_{q(Z^t | X^{t-2:t-1}, X^t, F^t)}\Big[\textstyle\sum_{\alpha \in \mathcal{V}_G} \sum_{j=1}^{d_x} \log \mathcal{N}\Big(X^t_{\alpha,j} \Big| g\big(Z^t, X^{t-2:t-1}, F^t\big)_{\alpha,j}, \sigma^2_{\alpha,j}\Big)\Big]\end{aligned} \tag{6}$$

which is equal to the (negative) Evidence Lower Bound (ELBO) when the weighting is $\lambda_{\text{KL}} = 1$. While the predictor g is a deterministic mapping, we introduce a Gaussian likelihood in eq. (6) to get a well-defined learning problem. This setup corresponds to the common practice in VAEs of assuming Gaussian observation noise, but not adding this to samples from the model [42]. The standard deviation $\sigma_{\alpha,j}$ can either be a second output from the predictor or manually chosen (see appendix D for details). As with deterministic models [23, 20, 8, 34], we found it crucial to fine-tune on rolled out forecasts of multiple time steps. This improves stability and performance for longer lead times. In the final fine-tuning we include also a Continuous Ranked Probability Score (CRPS) loss term $\mathcal{L}_{\text{CRPS}}$ [15, 22]. The full objective function is then $\mathcal{L} = \mathcal{L}_{\text{Var}} + \lambda_{\text{CRPS}} \mathcal{L}_{\text{CRPS}}$, with λ_{CRPS} a weighting hyperparameter. Including this CRPS loss improves the calibration of ensemble forecasts.

4.5 Improved GNN Layers: Propagation Networks

In Graph-EFM there is a large amount of information that needs to be propagated between the grid and Z^t. However, the Interaction Network GNNs are biased towards keeping old representations of receiver nodes, rather than updating this with new information from incoming edges. Note in eq. (1) that if the MLPs are initialized to give outputs close to 0, there will be no change to $\boldsymbol{e}_{\alpha\to\beta}$ and H^R_β.

In practice the model has a hard time learning to propagate useful information up from the grid to Z^t. Even when trained purely as an auto-encoder ($\lambda_{\text{KL}} = 0$), Z^t easily ends up being ignored. To remedy this we propose an alternative GNN formulation that we call *Propagation Network*, defined by

$$\tilde{\boldsymbol{e}}_{\alpha\to\beta} \leftarrow H^S_\alpha + \text{MLP}\big(\boldsymbol{e}_{\alpha\to\beta}, H^S_\alpha, H^R_\beta\big) \qquad \boldsymbol{e}_{\alpha\to\beta} \leftarrow \boldsymbol{e}_{\alpha\to\beta} + \tilde{\boldsymbol{e}}_{\alpha\to\beta} \tag{7a}$$

$$\tilde{H}^R_\beta \leftarrow \frac{1}{|\text{Ne}(\beta)|} \textstyle\sum_{\alpha \in \text{Ne}(\beta)} \tilde{\boldsymbol{e}}_{\alpha\to\beta} \qquad H^R_\beta \leftarrow \tilde{H}^R_\beta + \text{MLP}\Big(H^R_\beta, \tilde{H}^R_\beta\Big). \tag{7b}$$

For MLPs initialized with outputs close to 0, Propagation Networks reduce to averaging the values of neighboring nodes. This encourages the propagation of information from H^S to H^R by construction. Propagation Networks were found to perform better also in the deterministic model (see comparison in appendix L.1), so we employ these in both Graph-FM and Graph-EFM.

5 Experiments

To evaluate our models we conduct experiments on both global and limited area forecasting. The models are implemented[2] in PyTorch and trained on 8 A100 80 GB GPUs in a data-parallel configuration. Training takes 700–1400 total GPU-hours for the global models, and around half of that for the limited area models. The computational demands prevent us from re-training multiple models for statistical analysis. Once trained, sampling from Graph-EFM is highly efficient. Using batched sampling on a single GPU, 80 ensemble members are produced in 200 s (2.5 s per member) for global forecasting.

Metrics We measure the skill of deterministic models by Root Mean Squared Error (**RMSE**). For probabilistic models we compute the RMSE for the ensemble mean. Good skill in terms of RMSE is however not enough for ensemble forecasts, where we want to capture the full distribution. For these we also assess the ensemble calibration by computing the Spread-Skill-Ratio (**SpSkR**). Calibrated uncertainty corresponds to SpSkR ≈ 1 [13]. We additionally use **CRPS** to measure how well the marginal distributions of the model matches the data. For deterministic models the CRPS reduces to Mean Absolute Error (MAE). Complete definitions of all metrics are given in appendix E.

Models Achieving a fair comparisons of the actual machine learning methodology in MLWP is challenging due to models using different spatial resolution, variables and initial states. We here train an illustrative set of models on the same data and with comparable training setups. Our full **Graph-EFM** model is compared to: 1) **Graph-EFM (ms)**, a version of Graph-EFM using a multi-scale mesh graph instead of the hierarchical one. 2) **Graph-FM**, our deterministic model using the hierarchical graph. 3) **GraphCast***, a reimplementation of GraphCast [23], adapted and trained on our datasets. 4) **GraphCast*+SWAG**, a multi-model ensemble created by applying Stochastic Weight Averaging Gaussian (SWAG) [30] to GraphCast*. Inspired by Graubner et al. [16], this represents a simple way to augment a deterministic model to perform ensemble forecasting. Further details about the baseline models are given in appendix C.5. For ensemble models we sample 80 members for the global experiments and 100 members for limited area forecasting. In appendix L.3 we investigate the impact of ensemble size on the evaluation. We find that improvements in metric values quickly saturate when increasing the ensemble size. This shows that sampling even more members would have negligible impact on the results of our experiments.

5.1 Global Forecasting with ERA5

Data and graphs We experiment on global weather forecasting up to 10 days with 6 h time steps. The dataset used for training and evaluation is a 1.5° version of the global ERA5 reanalysis[3] [17], provided through the WeatherBench 2 benchmark [40]. The models forecast $d_x = 83$ different variables in total, including both surface-level variables and atmospheric variables at 13 different pressure levels. We use the years 1959–2017 for training, 2018–2019 for validation and 2020 as a test set. Forecasts are always started from initial conditions taken directly from ERA5, both during training and evaluation. For global forecasting we use the graph generation process from GraphCast [23]. The multi-scale graph $\mathcal{G}_{\text{MS}}$ is created by refining the icosahedron 4 times. The hierarchical graph contains 4 levels of such icosahedral grids. More details on the global experiments are given in appendix H.

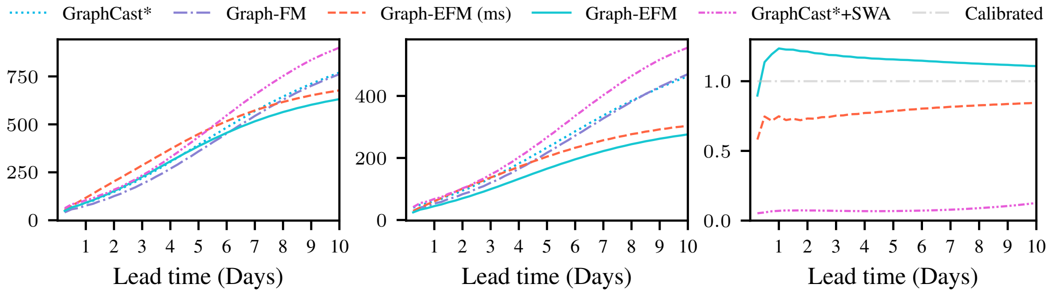
Results As the models forecast many different variables we present only a selection of results in the main paper. Metric values for geopotential (`z500`) and 2 m temperature (`2t`) are listed in table 1 and results for mean sea level pressure (`msl`) plotted in fig. 3. Line plots for all metrics and a large number of variables are given in appendix J.1. In the appendix we also show comparisons to additional models from the literature, trained on different data, as well as the physics-based IFS-ENS model [12]. The

[2] Our code is available at `https://github.com/mllam/neural-lam/tree/prob_model_global` (global forecasting) and `https://github.com/mllam/neural-lam/tree/prob_model_lam` (LAM).

[3] Provided by the Copernicus Climate Change Service under the ECMWF Copernicus License.

Table 1: Selection of results for global forecasting, including geopotential at 500 hPa (`z500`) and 2 m temperature (`2t`). For RMSE and CRPS lower values are better, and SpSkR should be close to 1 for a calibrated ensemble. The best metric values are marked with **bold** and second best underlined.

		Lead time 5 days			Lead time 10 days		
Variable	Model	RMSE	CRPS	SpSkR	RMSE	CRPS	SpSkR
`z500`	GraphCast*	387	236	-	808	498	-
	Graph-FM	**363**	223	-	825	510	-
	GraphCast*+SWAG	437	269	0.07	960	590	0.12
	Graph-EFM (ms)	472	211	0.77	756	333	0.83
	Graph-EFM	399	**169**	**1.18**	**695**	**299**	**1.15**
`2t`	GraphCast*	1.65	1.00	-	2.82	1.69	-
	Graph-FM	**1.57**	0.94	-	2.82	1.66	-
	GraphCast*+SWAG	2.03	1.20	0.06	3.58	2.04	0.13
	Graph-EFM (ms)	1.76	0.77	0.75	2.55	1.09	0.82
	Graph-EFM	1.64	**0.71**	**0.98**	**2.32**	**1.00**	**0.99**



(a) RMSE

(b) CRPS

(c) SpSkR

Figure 3: Results for global forecasting of mean sea level pressure (`msl`) at all lead times.

ensemble mean from Graph-EFM often shows improvements in RMSE over the deterministic models, especially for longer lead times. Across the ensemble models, Graph-EFM achieves lower CRPS values, better capturing the distribution of the weather data. Without any perturbations to initial states Graph-EFM reaches a SpSkR close to 1. We note that GraphCast*+SWAG does not produce useful ensemble forecasts, as these are poorly calibrated and in general do not lead to improved forecast errors. Figure 4 shows an example forecast from Graph-EFM for specific humidity (`q700`) at 10 days lead time. Examples for other variables are given in appendix J.2.

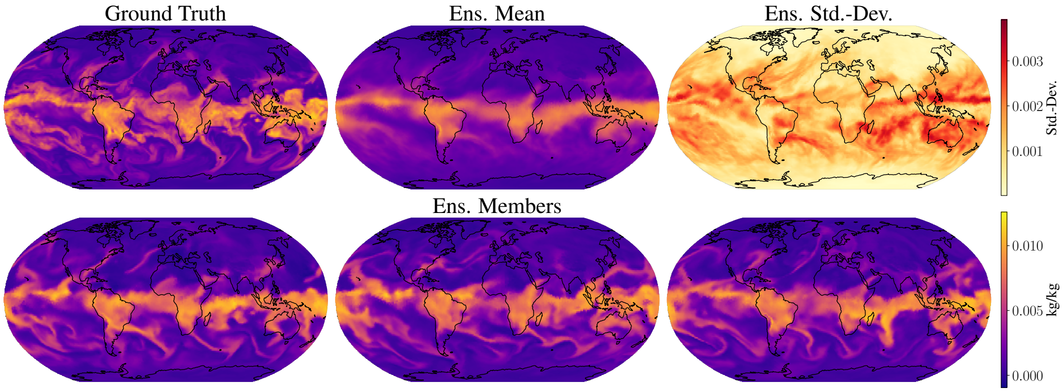


Figure 4: Example Graph-EFM ensemble forecast for specific humidity at 700 hPa (`q700`), for lead time 10 days. The bottom row shows 3 ensemble members, randomly chosen out of the 80.

Table 2: Selection of results for LAM forecasting, including geopotential at 500 hPa (`z500`) and integrated column of water vapor (`wvint`).

		Lead time 24 h			Lead time 57 h		
Variable	Model	RMSE	CRPS	SpSkR	RMSE	CRPS	SpSkR
`z500`	GraphCast*	**153**	108	-	**201**	138	-
	Graph-FM	230	162	-	354	238	-
	GraphCast*+SWAG	219	136	0.08	376	206	0.10
	Graph-EFM (ms)	400	261	0.22	711	470	0.23
	Graph-EFM	172	**91**	**0.84**	219	**115**	**0.75**
`wvint`	GraphCast*	**1.51**	1.01	-	**2.06**	1.32	-
	Graph-FM	1.64	1.08	-	2.48	1.58	-
	GraphCast*+SWAG	1.78	1.17	0.05	2.34	1.50	0.05
	Graph-EFM (ms)	2.39	1.43	0.16	3.51	2.12	0.13
	Graph-EFM	1.61	**0.79**	**0.57**	2.08	**1.00**	**0.53**

Extreme weather case study An important use case for ensemble forecasting is modeling extreme weather events. While higher resolutions than 1.5° are generally desirable for accurately capturing such extremes, we conduct one case study on using Graph-EFM for forecasting hurricane Laura. The full case study with visualized forecasts is available in appendix F. For this example we show that there exists ensemble members accurately predicting the landfall location of the hurricane at 7 days lead time, while the deterministic models still show no sign of the hurricane in the region. Closer to the landfall event the ensemble forecast from Graph-EFM indicates uncertainties associated with the landfall location and wind intensity. This demonstrates the added value of a probabilistic forecasting model.

5.2 Limited Area Modeling with MEPS Data

In LAMs weather forecasts are produced for a bounded region of the globe. LAM forecasting allows for higher resolution modeling and regionally tailored model configurations [11], properties that can be inherited by MLWP models by training on LAM data. To model weather over a limited domain, boundary conditions need to be taken into account. In physics-based LAMs these are typically given by a global forecast [38, 44, 7, 32]. We adapt a similar approach for MLWP LAMs, by taking boundary conditions as additional forcing along the boundary of the forecast area. The problem of LAM forecasting is thus about simulating physics not just based on the initial state, but also consistent with these boundary inputs. In the models we introduce N_b additional grid nodes along the area boundary, for the boundary forcing $B^t \in \mathbb{R}^{N_b \times d_x}$. Boundary forcing B^t is always fed together with X^t to the model. Grid nodes on the boundary and within the area are treated identically by the GNN layers. We perform this adaptation to all models in our experiment.

Data and graphs We experiment with a dataset containing 6069 forecasts from the MetCoOp Ensemble Prediction System (MEPS) LAM. Training on forecasts, the goal is here to learn a fast surrogate model for MEPS. We use forecasts started during April 2021 – Jun 2022 for training and validation, and forecasts from July 2022 – March 2023 as a test set. The data is laid out in a 238×268 grid with spatial resolution 10 km, covering the Nordic region. This dataset contains in total $d_x = 17$ weather variables, some repeated on multiple vertical levels. Forecasts are rolled out with 3 h time steps up to lead time 57 h. In this experiment we take also the boundary forcing directly from the MEPS dataset. We define the boundary as the outermost 10 grid positions. Using the same dataset for the area and boundary allows us to investigate the modeling choices in a controlled experimental setup. In an operational scenario the boundary forcing would instead come from a re-gridded global forecast. In the LAM setting we define our graphs as regular quadrilateral meshes covering the MEPS forecasting area, but with far fewer nodes than the original grid. The graph hierarchy $\mathcal{G}_1, \dots, \mathcal{G}_L$ is created by constructing such meshes at different resolutions. By placing each node in $\mathcal{G}_l$ at the center of 3×3 nodes in $\mathcal{G}_{l-1}$, we can merge 4 such graph levels to create $\mathcal{G}_{\text{MS}}$. In the hierarchical graph we instead introduce edges from each node in $\mathcal{G}_l$ to the 3×3 nodes in the level below. More details about the MEPS data and experiment can be found in appendix I.

Results A selection of metrics are shown in table 2 and full results given in appendix K. At these shorter lead times there is no clear benefit of probabilistic modeling in terms of RMSE. Still,

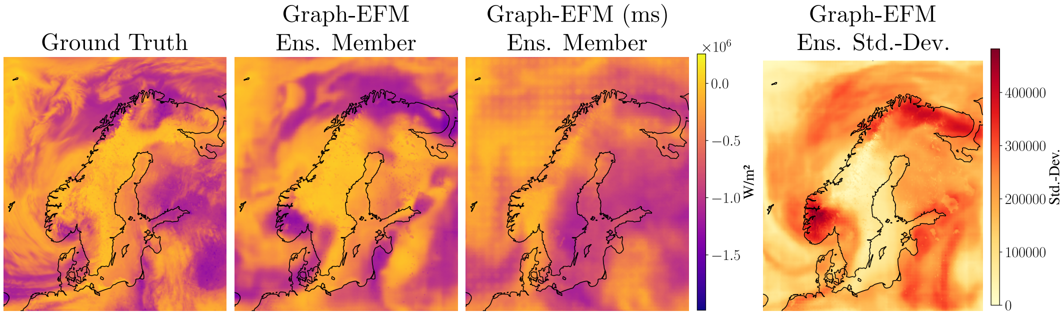


Figure 5: Example forecasts for net solar longwave radiation (`nlwrs`) at lead time 57 h.

as exemplified by the standard-deviation plotted in fig. 5, probabilistic modeling provides useful information about the forecast uncertainty. Comparing the ensemble members in fig. 5 highlights the improved spatial coherency of the hierarchical graph in Graph-EFM. In contrast, the Graph-EFM (ms) forecast looks patchy and lacks physically intuitive features. There are also clear visual artifacts, that can be traced to the multi-scale graph structure. We discuss this more in-depth in appendix G. In the LAM setting all models are under-dispersed, with SpSkR < 1. One explanation for this is that the boundary forcing constrains the space of plausible forecasts, hindering the ensemble spread.

6 Discussion

In this paper we have explored MLWP ensemble weather forecasting using graph-based latent variable models. Our Graph-EFM model is capable of efficiently producing accurate ensemble forecasts. This paves the way for large-scale MLWP ensemble forecasting both in operational use and research settings. In appendix B we further discuss the societal impact of this research. With this work we hope to emphasize that MLWP models are not just deterministic mappings, but parametrize distributions of weather states. It follows that ensemble forecasting should not be achieved by perturbing models, but by directly modeling the distribution of interest.

Limitations The training process comes with some complications in terms of choosing a training schedule and hyperparameters λ_{KL} and λ_{CRPS}. While the CRPS fine-tuning is an important training step, we have found that choosing a too high λ_{CRPS} can introduce visual artifacts, especially for the Graph-EFM (ms) model (see appendix G). While Graph-EFM produces diverse and physically plausible ensemble members, the forecasts still suffer from some of the blurriness common to deterministic models [23, 40]. We here trade off some of the visual fidelity achieved for example by diffusion models [39] for more efficient sampling of ensemble members.

Future work Interesting avenues for future work include learning probabilistic weather models based on other types of autoencoders [48, 49], or by directly optimizing scoring rules [36, 22]. Another approach for achieving efficient ensemble forecasting is to explore techniques for speeding up diffusion model sampling [47].

Acknowledgments and Disclosure of Funding

This research is financially supported by the Swedish Research Council via the project *Handling Uncertainty in Machine Learning Systems* (contract number: 2020-04122), the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, the Excellence Center at Linköping–Lund in Information Technology (ELLIIT), and the project *OWGRE*, funded by partners of the ERA-Net Smart Energy Systems and Mission Innovation through the Joint Call 2020. As such, this project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 883973. Our computations were enabled by the Berzelius resource at the National Supercomputer Centre, provided by the Knut and Alice Wallenberg Foundation.

References

[1] M. Astitha and E. Nikolopoulos. Definition of extreme weather events (subchapter 1.1). In *Extreme Weather Forecasting*, pages 1–7. Elsevier, 2023.

[2] P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende, and K. Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[3] P. Bauer, A. Thorpe, and G. Brunet. The quiet revolution of numerical weather prediction. *Nature*, 2015.

[4] K. Bi, L. Xie, H. Zhang, X. Chen, X. Gu, and Q. Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 2023.

[5] B. Bonev, T. Kurth, C. Hundt, J. Pathak, M. Baust, K. Kashinath, and A. Anandkumar. Spherical Fourier neural operators: Learning stable dynamics on the sphere. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[6] C. Bülte, N. Horat, J. Quinting, and S. Lerch. Uncertainty quantification for data-driven weather models. *arXiv preprint arXiv:2403.13458*, 2024.

[7] M. Bush, I. Boutle, J. Edwards, A. Finnenkoetter, C. Franklin, K. Hanley, A. Jayakumar, H. Lewis, A. Lock, M. Mittermaier, S. Mohandas, R. North, A. Porson, B. Roux, S. Webster, and M. Weeks. The second met office unified model–JULES regional atmosphere and land configuration, RAL2. *Geoscientific Model Development*, 2023.

[8] K. Chen, T. Han, J. Gong, L. Bai, F. Ling, J.-J. Luo, X. Chen, L. Ma, T. Zhang, R. Su, Y. Ci, B. Li, X. Yang, and W. Ouyang. FengWu: Pushing the skillful global medium-range weather forecast beyond 10 days lead. *arXiv preprint arXiv:2304.02948*, 2023.

[9] L. Chen, F. Du, Y. Hu, Z. Wang, and F. Wang. Swinrdm: integrate swinrnn with diffusion model towards high-resolution and high-quality weather forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2023.

[10] L. Chen, X. Zhong, F. Zhang, Y. Cheng, Y. Xu, Y. Qi, and H. Li. Fuxi: a cascade machine learning forecasting system for 15-day global weather forecast. *npj Climate and Atmospheric Science*, 2023.

[11] J. Coiffier. *Fundamentals of numerical weather prediction*. Cambridge University Press, 2011.

[12] ECMWF. Ifs documentation cy46r1 - part v: Ensemble prediction system, 2019. URL `https://www.ecmwf.int/node/19309`.

[13] V. Fortin, M. Abaza, F. Anctil, and R. Turcotte. Why should ensemble spread match the RMSE of the ensemble mean? *Journal of Hydrometeorology*, 2014.

[14] M. Fortunato, T. Pfaff, P. Wirnsberger, A. Pritzel, and P. Battaglia. Multiscale meshgraphnets. In *ICML 2022 2nd AI for Science Workshop*, 2022.

[15] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 2007.

[16] A. Graubner, K. Kamyar Azizzadenesheli, J. Pathak, M. Mardani, M. Pritchard, K. Kashinath, and A. Anandkumar. Calibration of large neural weather models. In *NeurIPS 2022 Workshop on Tackling Climate Change with Machine Learning*, 2022.

[17] H. Hersbach, B. Bell, P. Berrisford, S. Hirahara, A. Horányi, J. Muñoz-Sabater, J. Nicolas, C. Peubey, R. Radu, D. Schepers, A. Simmons, C. Soci, S. Abdalla, X. Abellan, G. Balsamo, P. Bechtold, G. Biavati, J. Bidlot, M. Bonavita, G. De Chiara, P. Dahlgren, D. Dee, M. Diamantakis, R. Dragani, J. Flemming, R. Forbes, M. Fuentes, A. Geer, L. Haimberger, S. Healy, R. J. Hogan, E. Hólm, M. Janisková, S. Keeley, P. Laloyaux, P. Lopez, C. Lupu, G. Radnoti, P. de Rosnay, I. Rozum, F. Vamborg, S. Villaume, and J.-N. Thépaut. The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 2020.

[18] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[19] Y. Hu, L. Chen, Z. Wang, and H. Li. SwinVRNN: A data-driven ensemble forecasting model via learned distribution perturbation. *Journal of Advances in Modeling Earth Systems*, 2023.

[20] R. Keisler. Forecasting global weather with graph neural networks. *arXiv preprint arXiv:2202.07575*, 2022.

[21] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

[22] D. Kochkov, J. Yuval, I. Langmore, P. Norgaard, J. Smith, G. Mooers, M. Klöwer, J. Lottes, S. Rasp, P. Düben, S. Hatfield, P. Battaglia, A. Sanchez-Gonzalez, M. Willson, M. P. Brenner, and S. Hoyer. Neural general circulation models for weather and climate. *Nature*, 2024.

[23] R. Lam, A. Sanchez-Gonzalez, M. Willson, P. Wirnsberger, M. Fortunato, F. Alet, S. Ravuri, T. Ewalds, Z. Eaton-Rosen, W. Hu, A. Merose, S. Hoyer, G. Holland, O. Vinyals, J. Stott, A. Pritzel, S. Mohamed, and P. Battaglia. Learning skillful medium-range global weather forecasting. *Science*, 2023.

[24] S. Lang, M. Alexe, M. Chantry, J. Dramsch, F. Pinault, B. Raoult, M. C. A. Clare, C. Lessig, M. Maier-Gerber, L. Magnusson, Z. B. Bouallègue, A. P. Nemesio, P. D. Dueben, A. Brown, F. Pappenberger, and F. Rabier. AIFS-ECMWF's data-driven forecasting system. *arXiv preprint arXiv:2406.01465*, 2024.

[25] C. Lessig, I. Luise, B. Gong, M. Langguth, S. Stadler, and M. Schultz. AtmoRep: A stochastic model of atmosphere dynamics using large scale representation learning. *arXiv preprint arXiv:2308.13280*, 2023.

[26] M. Leutbecher and T. Palmer. Ensemble forecasting. *Journal of Computational Physics*, 2008.

[27] L. Li, R. Carver, I. Lopez-Gomez, F. Sha, and J. Anderson. Generative emulation of weather forecast ensembles with diffusion models. *Science Advances*, 2024.

[28] M. Lino, C. Cantwell, A. A. Bharath, and S. Fotiadis. Simulating continuum mechanics with multi-scale graph neural networks. *arXiv preprint arXiv:2106.04900*, 2021.

[29] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[30] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[31] M. Mardani, N. Brenowitz, Y. Cohen, J. Pathak, C.-Y. Chen, C.-C. Liu, A. Vahdat, K. Kashinath, J. Kautz, and M. Pritchard. Residual diffusion modeling for km-scale atmospheric downscaling. *arXiv preprint arXiv:2309.15214*, 2023.

[32] M. Müller, M. Homleid, K.-I. Ivarsson, M. A. Ø. Køltzow, M. Lindskog, K. H. Midtbø, U. Andrae, T. Aspelien, L. Berggren, D. Bjørge, P. Dahlgren, J. Kristiansen, R. Randriamampianina, M. Ridal, and O. Vignes. AROME-MetCoOp: A nordic convective-scale operational weather prediction model. *Weather and Forecasting*, 2017.

[33] T. Nguyen, J. Brandstetter, A. Kapoor, J. K. Gupta, and A. Grover. ClimaX: A foundation model for weather and climate. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[34] T. Nguyen, R. Shah, H. Bansal, T. Arcomano, S. Madireddy, R. Maulik, V. Kotamarthi, I. Foster, and A. Grover. Scaling transformer neural networks for skillful and reliable medium-range weather forecasting. *arXiv preprint arXiv:2312.03876*, 2023.

[35] J. Oskarsson, T. Landelius, and F. Lindsten. Graph-based neural weather prediction for limited area modeling. In *NeurIPS 2023 Workshop on Tackling Climate Change with Machine Learning*, 2023.

[36] L. Pacchiardi, R. A. Adewoyin, P. Dueben, and R. Dutta. Probabilistic forecasting with generative networks via scoring rule minimization. *Journal of Machine Learning Research*, 2024.

[37] J. Pathak, S. Subramanian, P. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z. Li, K. Azizzadenesheli, P. Hassanzadeh, K. Kashinath, and A. Anandkumar. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

[38] T. V. Pham, C. Steger, B. Rockel, K. Keuler, I. Kirchner, M. Mertens, D. Rieger, G. Zängl, and B. Früh. ICON in climate limited-area mode (ICON release version 2.6.1): a new regional climate model. *Geoscientific Model Development*, 2021.

[39] I. Price, A. Sanchez-Gonzalez, F. Alet, T. Ewalds, A. El-Kadi, J. Stott, S. Mohamed, P. Battaglia, R. Lam, and M. Willson. Gencast: Diffusion-based ensemble forecasting for medium-range weather. *arXiv preprint arXiv:2312.15796*, 2023.

[40] S. Rasp, S. Hoyer, A. Merose, I. Langmore, P. Battaglia, T. Russel, A. Sanchez-Gonzalez, V. Yang, R. Carver, S. Agrawal, M. Chantry, Z. B. Bouallegue, P. Dueben, C. Bromberg, J. Sisk, L. Barrington, A. Bell, and F. Sha. WeatherBench 2: A benchmark for the next generation of data-driven global weather models. *arXiv preprint arXiv:2308.15560*, 2023.

[41] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*. Springer International Publishing, 2015.

[42] O. Rybkin, K. Daniilidis, and S. Levine. Simple and effective vae training with calibrated decoders. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

[43] S. Scher and G. Messori. Ensemble methods for neural network-based weather forecasts. *Journal of Advances in Modeling Earth Systems*, 2021.

[44] Y. Seity, P. Brousseau, S. Malardel, G. Hello, P. Bénard, F. Bouttier, C. Lac, and V. Masson. The AROME-france convective-scale operational model. *Monthly Weather Review*, 2011.

[45] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

[46] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[47] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[48] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

[49] A. van den Oord, O. Vinyals, and k. kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[50] Y. Verma, M. Heinonen, and V. Garg. ClimODE: Climate forecasting with physics-informed neural ODEs. In *International Conference on Learning Representations*, 2024.

[51] J. A. Weyn, D. R. Durran, R. Caruana, and N. Cresswell-Clay. Sub-seasonal forecasting with a large ensemble of deep-learning weather prediction models. *Journal of Advances in Modeling Earth Systems*, 2021.

[52] J.-I. Yano, M. Z. Ziemiański, M. Cullen, P. Termonia, J. Onvlee, L. Bengtsson, A. Carrassi, R. Davy, A. Deluca, S. L. Gray, V. Homar, M. Köhler, S. Krichak, S. Michaelides, V. T. J. Phillips, P. M. M. Soares, and A. A. Wyszogrodzki. Scientific challenges of convective-scale numerical weather prediction. *Bulletin of the American Meteorological Society*, 2018.