Machine Learning with Graphs Project
Paper : *Probabilistic Weather Forecasting with Hierarchical Graph Neural Networks*

Abdoul ZEBA [1] **Amine RAZIG** [2]

Abstract

This report will mainly be a significant summary of the the paper *Probabilistic Weather Forecasting with Hierarchical Graph Neural Networks* (Oskarsson et al., 2024). We will provide an outline the main contributions, an overview of the methodology and some key experiments described in the paper.

All our code, theoretical explanations, and experimental results are available in this Github repository[1], ensuring the reproducibility of our work.

1. Motivation

Traditional weather forecasting begins by capturing a snapshot of Earth's current atmospheric conditions. Gathering data from satellites, weather stations, and buoys around the world, capturing cloud images, and measuring temperature, pressure, wind speed, and humidity. These data are then fed into supercomputers that generate a 3D grid twin of the atmosphere. These computers finally perform complex physics calculations to predict how these data interact, resulting in a weather forecast.

Despite this sophisticated process, the initial 3D grid of the atmosphere can never perfectly replicate reality due to data gaps, leading to increasing uncertainty in forecasts over time. To address this challenge, the idea is to employ ensemble forecasting, tweaking the initial data to produce up to many forecasts, ensemble forecasting, thus measuring uncertainty. Still, such a way of generating weather ensemble forecasts is computationally expensive, often leading to restrict the spatial resolution or the ensemble size and, consequently, reducing accuracy in representing the distribution.

(Oskarsson et al., 2024) address this challenge by proposing a probabilistic weather forecasting model. The goal is to model the full distribution $p(X^{1:T}|X^{-1:0}, F^{1:T})$ of future weather states. Where, $X^{1:T}$ represents the sequence of future weather states, $X^{-1:0}$ denotes the initial conditions, and $F^{1:T}$ includes the forcing inputs such as time of day and static geographical features. The model leverages hierarchical graph neural networks to efficiently sample ensemble forecasts, where each ensemble member $X^{1:T} \sim p(X^{1:T}|X^{-1:0}, F^{1:T})$ represents a possible trajectory of weather states.

The primary objective of this work, (Oskarsson et al., 2024), is to develop a framework that not only provides accurate deterministic forecasts but also quantifies the uncertainty in these predictions. By generating ensemble forecasts, the model allows for a more comprehensive understanding of potential weather scenarios, enabling better decision-making and risk assessment. This approach is particularly valuable for investigating extreme weather events, where understanding the range of possible outcomes is crucial.

2. Methodology

In the following sections, we will explain the details of the model used, including the hierarchical graph neural network architecture, the probabilistic formulation of the problem, and the training setup. We will also present experimental results demonstrating the effectiveness of the approach in both global and limited-area forecasting tasks.

2.1. Graph-based Ensemble Forecasting Model (Graph-EFM)

In their work, (Oskarsson et al., 2024) propose a probabilistic weather forecasting model combining a flexible latent-variable formulation with a graph-based forecasting framework, called **Graph-EFM**. They built a probabilistic model from ground up still working on a auto-regressive setup where they decomposed the distribution over the time steps.

[1]Ecole Polytechnique, Palaiseau, France [2]ENSAE, Palaiseau, France. Correspondence to: Abdoul ZEBA <abdoul.zeba@polytechnique.edu>, Amine Razig <amine.razig@polytechnique.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

[1]**https://github.com/arazig/ML-with-Graphs-Project-Weather-Forecasting-with-GNNs**

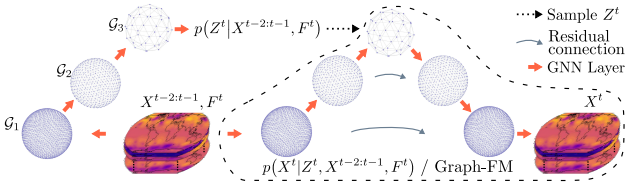

Figure 1. Overview of our Graph-EFM model, with example data and graphs for global forecasting (Oskarsson et al., 2024).

This approach allows for the generation of ensemble forecasts, where each ensemble member represents a possible trajectory of weather states, capturing the inherent uncertainty of what they called in the paper "chaotic weather systems".

The key innovation of Graph-EFM lies in its hierarchical graph neural network architecture, which enables efficient and spatially coherent sampling of ensemble members. By leveraging a hierarchical structure, the model propagates information across multiple spatial scales, capturing both local and global weather dynamics. This design not only improves the accuracy of forecasts but also ensures computational efficiency, requiring only a single forward pass per time step to generate large ensembles.

The paper introduce a latent random variable Z^t at each time step, with the idea that it should capture any uncertainty that is in this single time step prediction. Then, the deep latent variable model is formulated as follows: $p(X^{1:T}|X^{-1:0}, F^{1:T}) = \prod_{t=1}^{T} \int p(X^t|Z^t, X^{t-2:t-1}, F^t) p(Z^t|X^{t-2:t-1}, F^t) dZ^t$,

where the part $p(Z^t|X^{t-2:t-1}, F^t)$ if for the **latent map** and $p(X^t|Z^t, X^{t-2:t-1}, F^t)$, for the **predictor**.

The latent map describes uncertainty in single-step prediction by defining distribution of latent random variable Z^t while the predictor, chosen as a deterministic mapping with skip connection $\hat{X}^t = X^{t-1} + g(Z^t, X^{t-2:t-1}, F^t)$, predicts next weather state X^t given sample of Z^t.

With this formulation and letting the latent map be an isotropic Gaussian, $p(Z^t|X^{t-2:t-1}, F^t) = \prod_{a \in V_L} \mathcal{N}\left(Z_a^t | \mu_{Z^t}(X^{t-2:t-1}, F^t)_a, I\right)$ where The mean function μ_Z consists of a sequence of GNNs, Graph-EFM has an architecture similar to a (conditional) Variational AutoEncoder (VAE), as illustrated in Figure 1.

To efficiently capture multi-scale dependencies, they construct the hierarchical graph as follows. The hierarchy consists of multiple graph levels $G_1, G_2, \ldots, G_L$, where each level $G_l = (V_l, E_l)$ contains a decreasing number of vertices as l increases. Level 1 is the only one that is connected to the input grid, and information propagates upward through a sequence of directed graphs $G_{1,2}, G_{2,3}, \ldots, G_{L-1,L}$, enabling higher-level abstractions. A reverse sequence $G_{L,L-1}, \ldots, G_{2,1}$ allows to pass information and filter it back up to higher scales.

2.2. Training

We have seen in the last section that the model architecture is close to a VAE. Therefore, to train the model, they first define a variational distribution $q(Z^t|X^{t-2:t-1}, X^t, F^t)$ as a Graph Neural Network (GNN) that maps to an isotropic Gaussian and approximates the true posterior $p(Z^t|X^{t-2:t-1}, X^t, F^t)$ over Z^t.

Then, they enter in pre-training phase where they minimize the Evidence Lower Bound (ELBO) for a single time step with the focus is on learning latent space representations while maintaining spatial coherence:

$\mathcal{L}_{\text{Var}}(X^{t-2:t-1}, X^t, F^t) = \lambda_{\text{KL}} D_{\text{KL}}\Big(q(Z^t|X^{t-2:t-1}, X^t, F^t) \Big\| p(Z^t|X^{t-2:t-1}, F^t)\Big) - \mathbb{E}_{q(Z^t|X^{t-2:t-1}, X^t, F^t)}[\sum_{a \in V_G} \sum_{j=1}^{d_x} \log \mathcal{N}\Big(X_{a,j}^t \Big| g(Z^t, X^{t-2:t-1}, F^t)_{a,j}, \sigma_{a,j}^2\Big)]$. The weighted parameter λ_{KL} and the standard deviation $\sigma_{\alpha,j}$ are generally chosen manually.

After the pre-training phase is done, they fine-tune the model by multi-step rollouts including an additional loss term based on the Continuous Ranked Probability Score (CRPS). This helps mitigate blurriness in deterministic components while maintaining ensemble diversity:

$\mathcal{L} = \mathcal{L}_{\text{Var}} + \lambda_{\text{CRPS}} \mathcal{L}_{\text{CRPS}}$, where λ_{CRPS} is a weighting hyperparameter and $\mathcal{L}_{CRPS} = \sum_{t=1}^{T} \sum_{\alpha \in V_G} \sum_{j=1}^{d_z} w_\alpha \frac{1}{2}\Big(\Big|\hat{X}_{\alpha,j}^t - X_{\alpha,j}^t\Big| + \Big|\tilde{X}_{\alpha,j}^t - X_{\alpha,j}^t\Big| - \Big|\hat{X}_{\alpha,j}^t - \tilde{X}_{\alpha,j}^t\Big|\Big)$. $\hat{X}^t$ and $\tilde{X}^t$ coming from two independent ensemble members sampled from the model.

3. GraphCast

In this section, we decided to do a brief deep dive into the paper *GraphCast: Learning Skillful Medium-Range Global Weather Forecasting* (Lam et al., 2023), which introduces the weather forecasting model GraphCast. In (Oskarsson et al., 2024), GraphCast was reimplemented, trained on the considered datasets (ERA5 and MEPS), and used as a state-of-the-art baseline for comparison with Graph-EFM. Based on GNNs, GraphCast predicts the Earth's surface and atmospheric (3D) weather, 10 days head, at 0.25 degree latitude/longitude resolution.

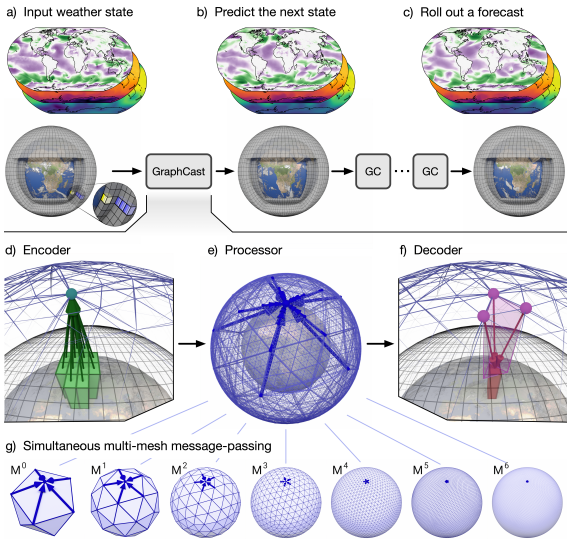

Figure 2. Model schematic (Lam et al., 2023)

3.1. Overview

GraphCast has three main components: an **encoder**, a **processor**, and a **decoder**.

The **encoder** maps local regions of the input (green boxes) to nodes in the multi-mesh graph representation (green upward arrows terminating in green-blue nodes). The **processor** then updates each multi-mesh node using learned message-passing (thick blue arrows pointing to nodes). Finally, the **decoder** maps the processed multi-mesh features (purple nodes) back onto the grid representation (red downward arrows terminating in red boxes). See Figure 2.

The learning process of GraphCast is driven by an autoregressive loss to penalize long-term. It was trained to minimize the mean square error (MSE) between the target output X and predicted output $\hat{X}$. See the appendix A.1 for the loss objective function.

3.2. Comparison of GraphCast and Graph-EFM

GraphCast and Graph-EFM are two advanced machine learning weather forecasting techniques that differ in architecture, prediction methodology, and operational use. GraphCast generates one high-resolution forecast using a graph neural networks architecture with state-of-the-art performance on 10-day forecasts. Graph-EFM, on the other hand, yields ensemble forecasts, which create numerous potential weather sequences to calculate uncertainty—a primary necessity for calculating dangers like excessive heat or cyclones.

The authors to evaluated the models used two datasets: the ERA5 (global reanalysis (Hersbach et al., 2020), 1.5° resolution, 10-day forecasts) and the MEPS (Müller et al., 2017)

Dataset	**ERA5**	**MEPS [6]**
Type	Reanalysis	Forecasts
Region	Global	Nordics
Years	61	2
Variables	83	17
Resolution	1.5°, 6 h	10 km, 3 h
Forecast Length	10 days	57 h

Table 1. Details on the datasets

(Nordic forecasts, 10 km resolution, 57-hour forecasts). See the Table 1 for more details. They compared their proposed Graph-EFM model against several baselines models such as a multi-scale verison of EFM instead of hierachical, the Graph-FM (it is a deterministic hierarchical model), and a more important reimplementation of GraphCast, which we will refer to as GraphCast*. Then, the model performance is evaluated using the metrics RMSE computed on the ensemble mean, the CRPS wich measure how well the model distribution correspond to the data, and finally the SpSkR metric for measuring how well the uncertainty in ensemble forecasts matches the actual forecast errors, with an optimal value of 1 indicating perfect calibration. We present the results of the evaluation in the next section.

4. Results

4.1. Author's experiments

The authors of (Oskarsson et al., 2024) first conducted experiment on limited area modelling. They do this to develop a fast surrogate model for the MEPS Limited Area Model, which provides high-resolution weather forecasts for the Nordic region. The Limited area models require boundary conditions as additional inputs. The goal was to test whether Graph-EFM could efficiently handle regional forecasting while maintaining accuracy and capturing uncertainty. To simulate weather dynamics accurately we need boundary conditions, so the authors adapted an approach for their machine learning-based LAM by introducing additional grid nodes along the boundary to incorporate boundary forcing B^t. These boundary conditions were fed into the model alongside the initial weather states X^t, and the grid nodes (both boundary and internal) were treated identically by the GNN layers.

The Table 2 highlights that Graph-EFM consistently achieves the best CRPS scores and the highest SpSkR values across both lead times (24h and 57h), this indicates superior probabilistic forecasting and better calibration of uncertainty compared to the other models. While GraphCast* has the lowest RMSE for all the lead time and variable, Graph-EFM strikes a balance between accuracy and uncertainty quantification, making it the most effective model for probabilistic limited-area forecasting.

Table 2. Selection of results for LAM forecasting, including geopotential at 500 hPa (`z500`) and integrated column of water vapor (`wvint`). The best metric values are marked with **bold** (Oskarsson et al., 2024).

		Lead time 24h		
Variable	Model	RMSE	CRPS	SpSkR
`z500`	GraphCast*	**153**	108	-
	Graph-EFM	172	**91**	**0.84**
`wvint`	GraphCast*	**1.51**	1.01	-
	Graph-EFM	1.61	**0.79**	**0.57**
		Lead time 57h		
Variable	Model	RMSE	CRPS	SpSkR
`z500`	GraphCast*	**201**	138	-
	Graph-EFM	219	**115**	**0.75**
`wvint`	GraphCast*	**2.82**	1.32	-
	Graph-EFM	2.08	**1.00**	**0.53**

Next, for the global analysis Graph-EFM performed well, producing spatially coherent ensemble members and demonstrating improved calibration compared to baseline models like GraphCast* and Graph-FM. However, the ensemble forecasts were slightly under-dispersed (SpSkR < 1).

Table 3. Selection of results for global forecasting, including geopotential at 500 hPa (`z500`) and 2 m temperature (`2t`). (Oskarsson et al., 2024).

		Lead time 5 days		
Variable	Model	RMSE	CRPS	SpSkR
`z500`	GraphCast*	**387**	236	-
	Graph-EFM	399	**169**	**1.18**
`2t`	GraphCast*	1.65	1.00	-
	Graph-EFM	**1.64**	**0.71**	**0.98**
		Lead time 10 days		
Variable	Model	RMSE	CRPS	SpSkR
`z500`	GraphCast*	808	498	-
	Graph-EFM	**695**	**299**	**1.15**
`2t`	GraphCast*	2.82	1.69	-
	Graph-EFM	**2.32**	**1.00**	**0.99**

Also, an important remark is that, as the lead time increases, Graph-EFM outperforms other models, achieving lower errors and better calibration (SpSkR ≈ 1). This demonstrate its ability to capture uncertainty in long-term forecasts (cf. Table 3) witch is important for weather prediction. Graph-EFM outperforms GraphCast* in most metrics for both `z500` and `2t` at 5-day and 10-day lead times. While GraphCast* has slightly lower RMSE for `z500` at 5 days, Graph-EFM has superior CRPS and spatial skill (SpSkR) in all cases and a notably lower RMSE at 10 days. For `2t`, Graph-EFM consistently produces more precise and trustworthy predictions with lower RMSE and CRPS and is therefore the overall better model.

4.2. Our experiments

In this section, we present our experiments. Our objective was to model the fundamental principles used in the paper's methodology rather than simply reproducing the same results, which would require significant computational resources and be less interesting from an educational perspective.

Inspired by the paper method, we built two forecasting models for the North-West region of France using data from the ERA5 database. The observed variables are the 2-meter temperature and the two wind components.

We propose two graph encoder-decoder architectures:

- The first is a deterministic model.
- The second, probabilistic, introduces a latent variable Z to improve forecasting at time $t + 1$. (cf. figure [4] in Appendix)

Below one of the results we obtained with the deterministic encoder-decoder based GCN.

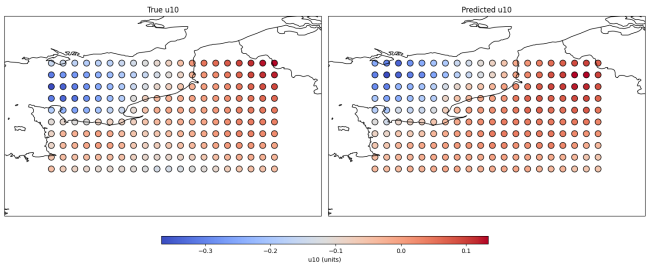

Figure 3. Forecasting with Graph One-Time-Step Convolutional Network Encoder-Decoder Structure (U-Wind Component)

As observed, the predictions are quite accurate. Predictions for other variables and architectures can be found in the appendix. The complete implementation code is available at the following at **https://github.com/arazig/ML-with-Graphs-Project-Weather-Forecasting-with-GNNs**.

5. Conclusion

In conclusion, the paper presents Graph-EFM model for ensemble weather forecasting using graph-based latent variable models. It efficiently makes accurate ensemble forecasts, emphasizing the value of modeling distributions of weather states directly over perturbing models.

However, the training process presents challenges, particularly in selecting a training schedule and balancing hyperpa-

rameters like λ_{KL} and λ_{CRPS}, as excessive CRPS fine-tuning can introduce artifacts.

References

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R. J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., and Thépaut, J.-N. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730): 1999–2049, 2020. doi: https://doi.org/10.1002/qj.3803. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3803.

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., Merose, A., Hoyer, S., Holland, G., Vinyals, O., Stott, J., Pritzel, A., Mohamed, S., and Battaglia, P. Graphcast: Learning skillful medium-range global weather forecasting, 2023. URL https://arxiv.org/abs/2212.12794.

Müller, M., Homleid, M., Ivarsson, K.-I., Køltzow, M. A. , Lindskog, M., Midtbø, K. H., Andrae, U., Aspelien, T., Berggren, L., Bjørge, D., Dahlgren, P., Kristiansen, J., Randriamampianina, R., Ridal, M., and Vignes, O. Arome-metcoop: A nordic convective-scale operational weather prediction model. *Weather and Forecasting*, 32(2):609 – 627, 2017. doi: 10.1175/WAF-D-16-0099.1. URL https://journals.ametsoc.org/view/journals/wefo/32/2/waf-d-16-0099_1.xml.

Oskarsson, J., Landelius, T., Deisenroth, M. P., and Lindsten, F. Probabilistic weather forecasting with hierarchical graph neural networks, 2024. URL https://arxiv.org/abs/2406.04759.

A. Appendix

A.1. GraphCast Loss Function

$$\mathcal{L}_{\text{MSE}} = \frac{1}{|D_{\text{batch}}|} \sum_{d_0 \in D_{\text{batch}}} \frac{1}{T_{\text{train}}} \sum_{\tau \in 1:T_{\text{train}}} \frac{1}{|G_{0.25^\circ}|} \sum_{i \in G_{0.25^\circ}} \sum_{j \in J} s_j w_j a_i \left(x_{i,j}^{d_0+\tau} - \hat{x}_{i,j}^{d_0+\tau} \right)^2$$

where

- $\tau \in 1 : T_{\text{train}}$ represents the lead times associated with the T_{train} autoregressive steps.
- $d_0 \in D_{\text{batch}}$ denotes the forecast initialization date-times within a batch of forecasts used in training.
- $j \in J$ is the index for variables, which, in the case of atmospheric data, corresponds to pressure levels.
- $i \in G_{0.25^\circ}$ refers to the grid points defined by latitude and longitude coordinates.
- $x_{i,j}^{d_0+\tau}$ and $\hat{x}_{i,j}^{d_0+\tau}$ denote the actual and predicted values, respectively, for a given variable, location, and lead time.
- s_j represents the inverse variance of time differences for each variable.
- w_j is the weighting factor assigned to each variable in the loss function.
- a_i corresponds to the area of a latitude-longitude grid cell, which varies with latitude and is normalized such that its mean value across the grid is one.

B. Experiments modelisation

The following figures show the probabilistic model used in our experiments for predictions of the U-wind component, V-wind component, and temperature at 2 meters.

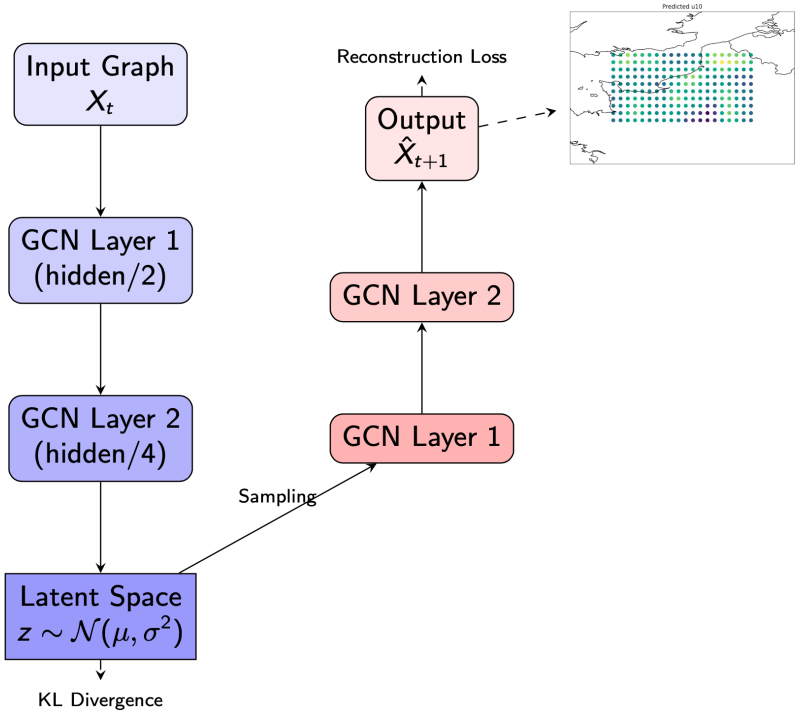

Figure 4. Experimental probabilistic model.

C. Prediction Results

C.1. Graph Convolutional Network (GCN) - Encoder Architecture

The following figures show the one-step-ahead predictions using the Graph Convolutional Network (GCN) with an encoder architecture. Predictions are provided for the U-wind component, V-wind component, and temperature at 2 meters.

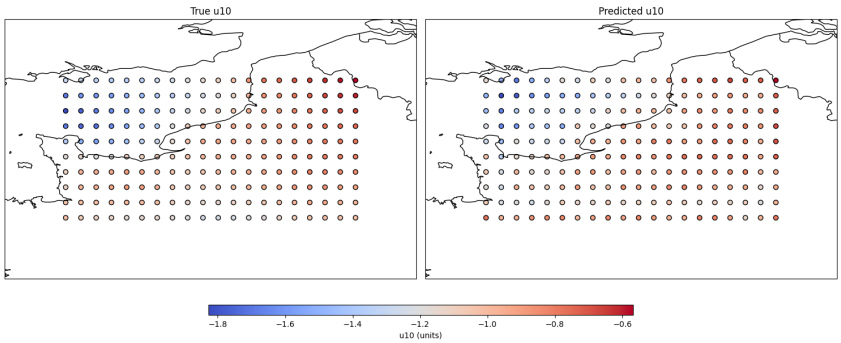

Figure 5. One-step-ahead prediction of the U-wind component using the Graph Convolutional Network.

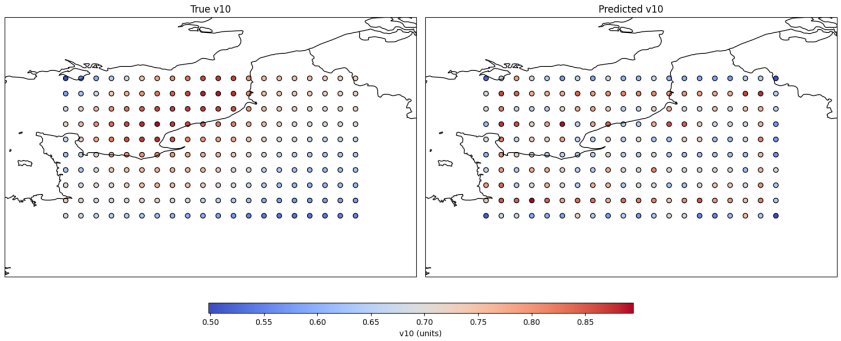

Figure 6. One-step-ahead prediction of the V-wind component using the Graph Convolutional Network.

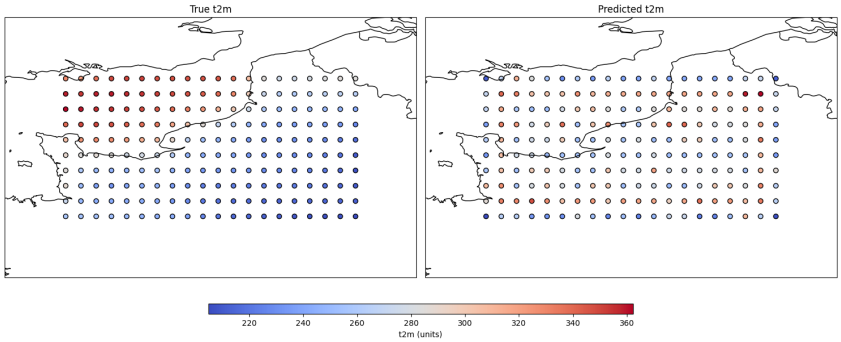

Figure 7. One-step-ahead prediction of the temperature at 2 meters using the Graph Convolutional Network.

C.2. Variational Graph Autoencoder (VGAE)

The following figures show the predictions using the Variational Graph Autoencoder (VGAE). Each image consists of three subplots: (1) the true values, (2) the predicted values, and (3) the difference between them.

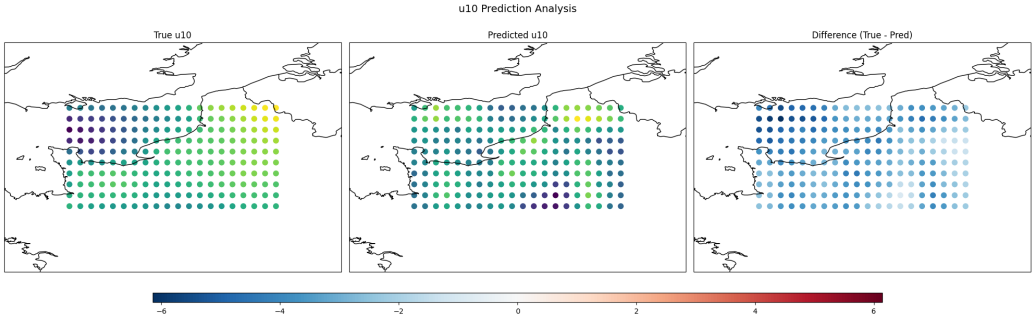

Figure 8. Variational Graph Autoencoder predictions for the U-wind component. The image consists of three subplots: true values, predicted values, and their difference.

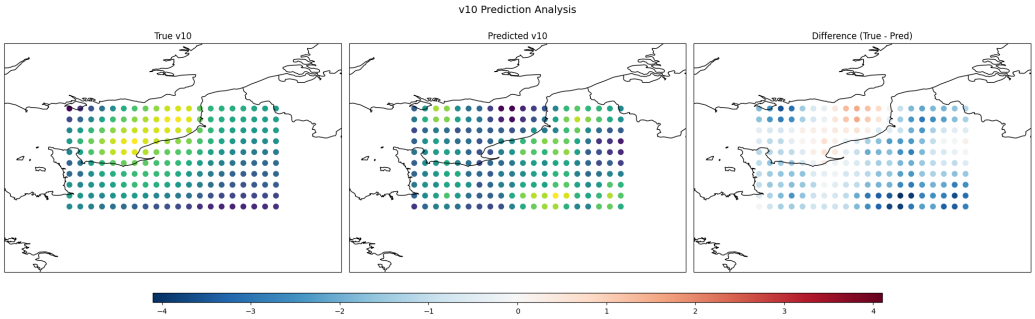

Figure 9. Variational Graph Autoencoder predictions for the V-wind component. The image consists of three subplots: true values, predicted values, and their difference.

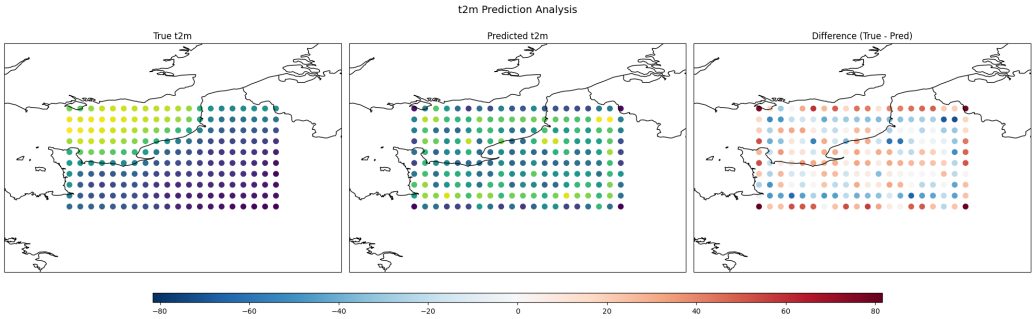

Figure 10. Variational Graph Autoencoder predictions for the temperature at 2 meters. The image consists of three subplots: true values, predicted values, and their difference.