Graph Neural Networks and Spatial Information Learning for Post-Processing Ensemble Weather Forecasts

Moritz Feik [1 2] **Sebastian Lerch** [1 2 *] **Jan Stühmer** [1 2 *]

Abstract

Ensemble forecasts from numerical weather prediction models show systematic errors that require correction via post-processing. While there has been substantial progress in flexible neural network-based post-processing methods over the past years, most station-based approaches still treat every input data point separately which limits the capabilities for leveraging spatial structures in the forecast errors. In order to improve information sharing across locations, we propose a graph neural network architecture for ensemble post-processing, which represents the station locations as nodes on a graph and utilizes an attention mechanism to identify relevant predictive information from neighboring locations. In a case study on 2-m temperature forecasts over Europe, the graph neural network model shows substantial improvements over a highly competitive neural network-based post-processing method.

1. Introduction

Modern weather forecasts utilize ensemble simulations from numerical weather prediction (NWP) models with different initial conditions or model physics. Even though NWP ensemble predictions have seen substantial progress over the past decades (Bauer et al., 2015), they often show systematic biases and fail to correctly quantify forecast uncertainty. Therefore, statistical or machine learning methods are required to correct these errors in a process referred to as post-processing, which has become a standard practice in research and operations. Most modern post-processing methods yield forecast distributions as their output, e.g. in the form of parameters of a pre-specified family of probability distributions. A major focus of post-processing research over the past years has been on flexible machine learning (ML) techniques which have demonstrated superior forecast performance, primarily due to their ability to incorporate additional predictor variables beyond ensemble forecasts of the target variable (Haupt et al., 2021; Vannitsem et al., 2021). Specifically, neural network (NN)-based distributional regression approaches first proposed by Rasp & Lerch (2018) have shown considerable success. Thereby, NNs enable the data-driven learning of nonlinear relationships between arbitrary predictor variables and forecast distribution parameters. Over the past years, NN-based post-processing methods have been extended in several directions, including non-parametric approaches (Bremnes, 2020), CNN-based methods for two-dimensional gridded forecast fields (Scheuerer et al., 2020; Veldkamp et al., 2021; Chapman et al., 2022; Horat & Lerch, 2024), generative ML methods for multivariate post-processing (Chen et al., 2024), or permutation-invariant set transformer architectures to model interactions between individual ensemble members (Höhlein et al., 2024).

The aforementioned CNN models incorporate spatial information between locations for gridded domains. Most station-based post-processing methods still treat every input data point separately, which prevents the models from sharing information across locations and thereby leveraging spatial structures in the forecast errors. To address this limitation, we propose graph neural network (GNN) architectures for post-processing, where weather stations form the nodes on a graph. By obtaining forecast distribution parameters in a node-level prediction setting, GNN-based post-processing methods are able to leverage spatial dependencies between stations and enable improved sharing of information across locations during model training and inference compared to standard NN approaches.

2. Data

In order to facilitate a fair and standardized comparison to other methods, we use EUPPBench, a benchmark dataset for ensemble post-processing (Demaeyer et al., 2023). The dataset includes medium-range ensemble forecasts from the European Centre for Medium-Range Weather Forecasts

[*]Equal contribution [1]Heidelberg Institute for Theoretical Studies, Heidelberg, Germany [2]Karlsruhe Institute of Technology, Karlsruhe, Germany. Correspondence to: Moritz Feik <moritz.feik@student.kit.edu>.

Proceedings of the 1st Machine Learning for Earth System Modeling workshop at the 41st International Conference on Machine Learning, Vienna, Austria.

(ECMWF) along with corresponding station observations over an extended period for multiple lead times. In total, the data spans from 1997 to 2018 and includes 122 weather stations in Europe, see Figure A.1 for details. Motivated by typical development practices for post-processing methods in operational weather prediction at meteorological services, the dataset contains both reforecasts and forecasts. Reforecasts are NWP model runs for past dates, which are conducted to obtain a large archive of past forecasts for analyzing various properties of the NWP system. The EUPPBench dataset contains 4180 reforecasts with a reduced number of 11 ensemble members from 1997 to 2017. In addition, the EUPPBench dataset includes of 730 daily operational forecasts from 2017–2018, which consist of 51 ensemble members. For both parts, a total of 31 predictor variables is available. We refer to Demaeyer et al. (2023) for details.

We here focus on forecasts of 2-meter temperature (T2M) and report results for lead times of 24 h, 72 h and 120 h in the interest of brevity. Given the structure of the EUPPBench dataset and following Höhlein et al. (2024), we consider two setups for post-processing tasks: "reforecast to reforecast" (*R2R*) and "reforecast to forecast" (*R2F*). The *R2R* task consists of fitting a post-processing models to the reforecast data from 1997–2013, and testing this model on reforecasts from 2014–2017, whereas the *R2F* task aims applying the fitted model to the forecast data from 2017–2018. The *R2F* task can be viewed as a typical pathway for developing a post-processing model in operational weather prediction, and comes with additional technical challenges, e.g., the need to account for varying numbers of ensemble members in the training and test data. Table A.1 lists the sizes of the training, validation and test datasets.

3. Methods

3.1. Forecast Evaluation

The main evaluation metric in the post-processing literature is the continuous ranked probability score (CRPS) given by $\mathrm{CRPS}(F, y) = \int_{-\infty}^{\infty} (F(z) - \mathbb{1}(y \leq z))^2 \, \mathrm{d}z$, where F is the cumulative distribution function of the forecast distribution, y is the realizing observation, and $\mathbb{1}$ denotes the indicator function (e.g., Gneiting & Katzfuss, 2014). The CRPS simultaneously evaluates calibration and sharpness of the forecast distribution, and can be computed in analytical form for ensembles and many parametric families (Jordan et al., 2019). To assess the statistical significance of score differences, we use tests of equal predictive performance (Diebold & Mariano, 1995).

3.2. DRN

We utilize the distributional regression network (DRN) model originally proposed in Rasp & Lerch (2018) as a state-of-the-art benchmark for station-based post-processing, which remains widely used and yields highly competitive benchmark forecasts (Vannitsem et al., 2021; Schulz & Lerch, 2022; Höhlein et al., 2024). The DRN model essentially is a standard fully-connected feed-forward NN which outputs the parameters of a predictive distribution, in our case the location μ and scale σ of a Gaussian distribution which has been demonstrated to be an appropriate choice for T2M prediction. Summary statistics from the NWP ensemble predictions of various meteorological variables serve as inputs to the NN. We estimate a single model jointly for all stations by optimizing the CRPS as a loss function. Thereby, station embeddings which map the station identifiers to a vector of latent features are used as additional inputs to generate local adaptivity. Our specific implementation of DRN follows Höhlein et al. (2024), see their Section 3 for details.

3.3. GNN

Graph neural networks (GNNs) are specialized deep learning models for graph-structured data, recognizing the value of representing problems in graph form rather than fixed grids or sequences (Gori et al., 2005; Scarselli et al., 2008). Unlike traditional architectures, GNNs enable the modelling of complex interactions between nodes and edges within the graph. Figure 1 provides an overview of the proposed GNN model architecture. In a first step, the graph $\mathcal{G}$ is created, and, for each node, the station identifier is replaced with its embedding, akin to the station embeddings in the DRN approach. The graph is then passed to K GNN-blocks, which iteratively refine the hidden representations $\mathbf{h}_{s,n}$, where s denotes the station and n the member of the NWP ensemble. Using K blocks, each node incorporates information from K hops away. Each of these blocks has skip connections inspired by the ResNet model (He et al., 2015). The residual learning approach helps to combat learning instabilities and leverage information from nodes multiple hops away. After the hidden features are created, they are aggregated using the *Deep Set* aggregation scheme (Zaheer et al., 2017). For each station, the hidden features of the different ensemble members are used to compute the final outputs μ_s and σ_s. The weights of all components of the GNN model are optimized jointly using the CRPS as a loss function.

3.3.1. Graph Topology

In order for a GNN to process data, the data must be transformed into a graph. For our dataset at hand, a graph $\mathcal{G}_t$ is created for each day t for which a forecast exists. Each node $v_{s,n}$ represents the forecast for a particular station s made by ensemble member n. Additionally, each node carries the forecasts of several meteorological variables generated by the respective ensemble member as attributes. Details are provided in Tables A.2 and A.3. Stations that are closer than a certain threshold $d_{\max}$ and stations with the same identifier

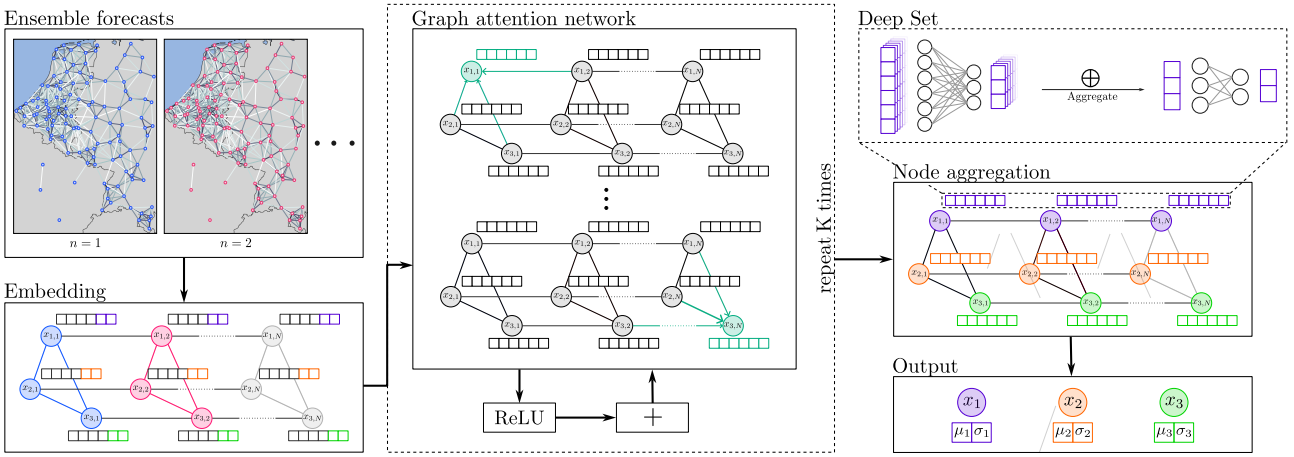

Figure 1. Schematic illustration of the GNN model for ensemble post-processing. The input graph $\mathcal{G}$ is created from the N-member ensemble forecasts at S stations. Next, the embedded station IDs are concatenated and passed to the GNN. The GNN block is repeated K times with residual connections, followed by the node aggregation. Finally, a softplus function is applied to σ to ensure positivity.

are bidirectionally connected. Each edge carries the normalized distance as a feature, while the edges between the ensemble members have a very small value ϵ instead of 0 as an attribute to facilitate training. Accordingly, the set of edges is $\mathcal{E} = \{(v_{i,u}, v_{j,v}) \,|\, i = j \vee d(v_{i,u}, v_{j,v}) < d_{\text{max}}\}$, where $d(\cdot)$ is the geodesic distance.

3.3.2. Graph Neural Networks

GNNs operate on the principle that they can learn and reason about graph-structured data by aggregating information from neighboring nodes and edges iteratively through message passing. One of the many types of GNNs is the graph attentional network (GAT), which weights incoming messages for each node using an attention function a (Veličković et al., 2018; Brody et al., 2022). The hidden representations in GATs are generally computed as $\mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}(i)} a(\mathbf{x}_i, \mathbf{x}_j)\psi(\mathbf{x}_j)\right)$ (Bronstein et al., 2021). For details on ϕ, ψ, and a, see Brody et al. (2022). With the attention mechanism, each node is able to discern important from unimportant neighbors and aggregate only relevant messages. Our implementation uses a GAT with *multi-head attention* to stabilize learning, employing multiple independent attention mechanisms and concatenating their outputs for the new node representation (Vaswani et al., 2017).

3.3.3. Permutation Invariant Node Aggregation

After processing the input graph $\mathcal{G}$, we generate predictions for μ_s and σ_s based on the output of the GNN, which consists of the hidden features $\mathbf{h}_{s,n}, n = 1, ..., N$. Since the ensemble members are interchangeable, the aggregation along the ensemble dimension n should be permutation invariant. Such an aggregation scheme can be achieved by using *Deep Sets* (Zaheer et al., 2017). Specifically, each set of hidden features for a given station $\mathcal{H}_s = \{\mathbf{h}_{s,n}, n = 1, \ldots, N\}$ is aggregated using $(\mu_s, \sigma_s) = \rho\left(\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{h}_{s,n})\right)$ as an aggregation function. In our concrete implementation, ρ and ϕ are both two-layer NNs.

4. Results

For our experiments we implemented the proposed method with the PyTorch Geometric (Fey & Lenssen, 2019) framework[1]. We evaluate the performance of the proposed model by training it on the EUPPBench dataset described in Section 2. Here, we focus on the "reforecast to forecast" (*R2F*) task. Additional, qualitatively similar results for the "reforecast to reforecast" (*R2R*) task are available in the supplemental material. Although the number of ensemble members, N, is arbitrary, we process the 51 ensemble members of the forecast data batch-wise in groups of 4×10 and a remaining group of 11, and average the predictions because the reforecasts used as training data contain only 11 ensemble members. This procedure aims to better account for the different number of ensemble members in the reforecast and forecast data, and results in better forecast performance. Table 1 provides an overview of the results for the *R2F* task. Results for the *R2R* task are available in the supplementary material in Table A.5. We compare the proposed model (GAT) against a GNN model which only operates on one graph based on the summary statistics (i.e., mean and standard deviation) of the ensemble forecasts (SMRY), a pure Deep Set architecture (DS), where all edges from the initial graph except for self loops are removed, a fully-connected, feed-forward DRN model described in Section 3.2, and the unprocessed ensemble forecasts (ENS). For each lead time, we train a separate model. These comparisons enable

[1]The implementation can be downloaded from https://github.com/hits-mli/gnn-post-processing.

Table 1. Scores for the reforecast to forecast task calculated per lead time, with the best CRPS scores highlighted in **bold**. The nominal level of the central prediction interval (PI) is $N - 1/N + 1$, where N is the number of ensemble members. The coverage (PI COVER) is the ratio of how often the observation is contained in the PI and should be close to the nominal level for a calibrated forecast.

LEAD TIME	24 h			72 h			120 h		
METHOD	CRPS	PI LENGTH	PI COVER	CRPS	PI LENGTH	PI COVER	CRPS	PI LENGTH	PI COVER
ENS	1.12	2.66	56.06	1.18	4.72	72.90	1.38	7.14	81.16
DRN	0.61	4.26	94.87	0.79	5.90	96.37	1.11	7.99	95.82
SMRY	0.62	4.45	95.53	0.79	6.17	97.01	1.10	8.31	96.64
DS	0.61	4.41	95.72	**0.78**	4.43	89.87	1.14	4.56	77.79
GAT	**0.60**	4.16	95.04	**0.78**	5.93	96.42	**1.09**	8.27	96.80

us to assess whether there is important information in the distribution of the NWP ensemble members and if the information sharing among weather station enabled by the GNN improves performance.

Not surprisingly, all post-processing methods substantially improve the raw ensemble predictions, which provide the sharpest prediction intervals, but fail to achieve a coverage close to the nominal value and thus clearly lack calibration. The proposed GAT model outperforms all other post-processing models in terms of the mean CRPS across all lead times and tasks. The statistical significance of these improvements is assessed via formal statistical tests following Diebold & Mariano (1995). Detailed results available in the supplemental material indicate that the improvements achieved by the GAT model are significant at the 5% level for a large fraction of the investigated stations and lead times. Interestingly, the DS model produces substantially sharper prediction intervals at longer lead times, but fails to achieve improvements over the DRN model in terms of the CRPS.

In order to investigate local differences, Figure 2 shows the relative improvement in terms of the CRPS, i.e., the station-specific continuous ranked probability skill score, CRPSS ($= 1 - \text{CRPS}_{\text{GAT}}/\text{CRPS}_{\text{DRN}}$), where DRN serves as a reference method and CRPS_{GAT} and CRPS_{DRN} denote the corresponding mean CRPS at a station. The GAT model achieves improvements over DRN for almost all investigated stations, which range up to around 14% in terms of the mean CRPS. While there is no clear geographical pattern, the improvements seem slightly larger at stations which are more centrally located within the graph.

Additional results on the calibration of the forecast distributions are available in the supplemental material. To asses feature importance, we employ a permutation importance procedure with two stage feature shuffling, following Höhlein et al. (2024). The four most important features are all related to temperature variables from the NWP ensemble, followed by the station ID and the station altitude. Together these six features are responsible for roughly 80 % of the total feature importance. Details and graphical illustrations are provided in the supplemental material.

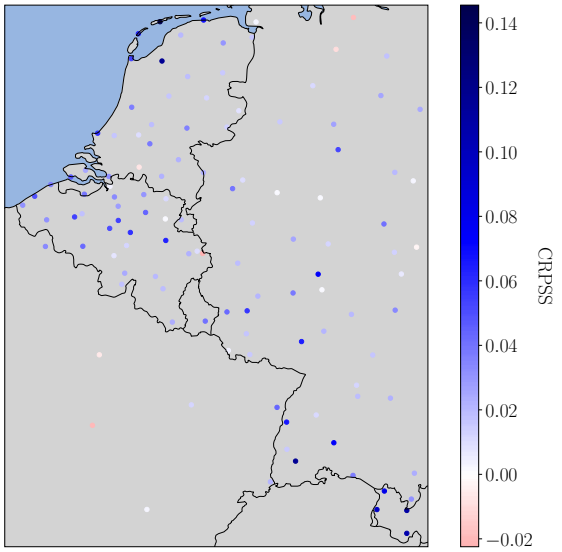

Figure 2. Station-specific improvement in terms of the CRPS of the GAT model over DRN, computed in terms of the CRPSS; where higher values indicate larger improvements by the GAT model.

5. Conclusion

We propose a graph neural network architecture for ensemble post-processing which enables an improved information sharing across station locations and achieves consistent and significant improvements over a highly competitive NN-based post-processing model across lead times and forecasting tasks on a benchmark dataset. Within the proposed GAT architecture, the attention mechanism is a specifically important component to achieving these improvements. Potential future extensions of the GAT model include extensions towards spatio-temporal GNNs (Li & Zhu, 2021) as well as other graph generation methods based on alternative, e.g. meteorologically motivated similarity-based distance metrics (Lerch & Baran, 2017). Further, a more detailed investigation of station-specific benefits of the GAT model and their relation to meteorological factors such as weather patterns or seasonality provides an interesting avenue for further analysis.

References

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

Benjamini, Y. and Hochberg, Y. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B*, 57(1):289–300, 1995.

Bremnes, J. B. Ensemble postprocessing using quantile function regression based on neural networks and bernstein polynomials. *Monthly Weather Review*, 148(1): 403–414, 2020.

Brody, S., Alon, U., and Yahav, E. How Attentive are Graph Attention Networks?, 2022. arXiv:2105.14491.

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, 2021. arXiv:2104.13478.

Chapman, W. E., Monache, L. D., Alessandrini, S., Subramanian, A. C., Ralph, F. M., Xie, S.-P., Lerch, S., and Hayatbini, N. Probabilistic predictions from deterministic atmospheric river forecasts with deep learning. *Monthly Weather Review*, 150(1):215–234, 2022.

Chen, J., Janke, T., Steinke, F., and Lerch, S. Generative machine learning methods for multivariate ensemble postprocessing. *Annals of Applied Statistics*, 18(1):159–183, 2024.

Demaeyer, J. and Stauffer, R. EUPP-benchmark/climetlab-eumetnet-postprocessing-benchmark, April 2024. original-date: 2021-11-25T10:31:59Z.

Demaeyer, J., Bhend, J., Lerch, S., Primo, C., Van Schaeybroeck, B., Atencia, A., Ben Bouallègue, Z., Chen, J., Dabernig, M., Evans, G., Faganeli Pucer, J., Hooper, B., Horat, N., Jobst, D., Merše, J., Mlakar, P., Möller, A., Mestre, O., Taillardat, M., and Vannitsem, S. The EUPP-Bench postprocessing benchmark dataset v1.0. *Earth System Science Data*, 15(6):2635–2653, 2023.

Diebold, F. X. and Mariano, R. S. Comparing predictive accuracy. *Journal of Business and Economic Statistics*, 13:253–263, 1995.

Fey, M. and Lenssen, J. E. Fast graph representation learning with pytorch geometric, 2019.

Gneiting, T. and Katzfuss, M. Probabilistic Forecasting. *Annual Review of Statistics and Its Application*, 1(1):125–151, 2014.

Gori, M., Monfardini, G., and Scarselli, F. A new model for learning in graph domains. In *Proceedings. 2005 IEEE international joint conference on neural networks, 2005.*, volume 2, pp. 729–734. IEEE, 2005.

Haupt, S. E., Chapman, W., Adams, S. V., Kirkwood, C., Hosking, J. S., Robinson, N. H., Lerch, S., and Subramanian, A. C. Towards implementing artificial intelligence post-processing in weather and climate: Proposed actions from the Oxford 2019 workshop. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200091, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition, 2015. arXiv:1512.03385.

Horat, N. and Lerch, S. Deep learning for post-processing global probabilistic forecasts on sub-seasonal time scales. *Monthly Weather Review*, 152:667–687, 2024.

Höhlein, K., Schulz, B., Westermann, R., and Lerch, S. Postprocessing of Ensemble Weather Forecasts Using Permutation-Invariant Neural Networks. *Artificial Intelligence for the Earth Systems*, 3(1):e230070, 2024.

Jordan, A., Krüger, F., and Lerch, S. Evaluating probabilistic forecasts with scoringRules. *Journal of Statistical Software*, 90(12):1–37, 2019.

Lerch, S. and Baran, S. Similarity-based semilocal estimation of post-processing models. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 66(1):29–51, 2017.

Li, M. and Zhu, Z. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4189–4196, 2021.

Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization, 2017. arXiv:1711.05101.

McGovern, A., Lagerquist, R., Gagne, D. J., Jergensen, G. E., Elmore, K. L., Homeyer, C. R., and Smith, T. Making the Black Box More Transparent: Understanding the Physical Implications of Machine Learning. *Bulletin of the American Meteorological Society*, 100(11):2175–2199, 2019.

Rasp, S. and Lerch, S. Neural Networks for Postprocessing Ensemble Weather Forecasts. *Monthly Weather Review*, 146(11):3885–3900, 2018.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

Scheuerer, M., Switanek, M. B., Worsnop, R. P., and Hamill, T. M. Using Artificial Neural Networks for Generating Probabilistic Subseasonal Precipitation Forecasts over California. *Monthly Weather Review*, 148(8):3489–3506, 2020.

Schulz, B. and Lerch, S. Machine Learning Methods for Postprocessing Ensemble Forecasts of Wind Gusts: A Systematic Comparison. *Monthly Weather Review*, 150 (1):235–257, 2022.

Vannitsem, S., Bremnes, J. B., Demaeyer, J., Evans, G. R., Flowerdew, J., Hemri, S., Lerch, S., Roberts, N., Theis, S., Atencia, A., Bouallègue, Z. B., Bhend, J., Dabernig, M., Cruz, L. D., Hieta, L., Mestre, O., Moret, L., Plenković, I. O., Schmeits, M., Taillardat, M., den Bergh, J. V., Schaeybroeck, B. V., Whan, K., and Ylhaisi, J. Statistical postprocessing for weather forecasts: Review, challenges, and avenues in a big data world. *Bulletin of the American Meteorological Society*, 102(3):E681 – E699, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention Is All You Need, 2017. arXiv:1706.03762.

Veldkamp, S., Whan, K., Dirksen, S., and Schmeits, M. Statistical Postprocessing of Wind Speed Forecasts Using Convolutional Neural Networks. *Monthly Weather Review*, 149(4):1141–1152, 2021.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep Sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

A. Supplementary Material

The supplementary material is organized as follows. Appendix A.1 gives an overview of the data and features used, and Appendix A.2 provides additional details on the model architecture and training, as well as additional results.

A.1. Data

The EUPPBench dataset (Demaeyer et al., 2023) includes (re)forecasts and observations of 2-m air temperature and additional auxiliary variables at lead times of 6 to 120 h in 6 h intervals for a total of 122 stations. The stations, along with their altitude, are shown in Figure A.1. The auxiliary variables are listed in Table A.2, and station-specific information included in the dataset is listed in Table A.3. The EUPPBench dataset is available through the CliMetLab API (Demaeyer & Stauffer, 2024)

We focus on the lead times of 24 h, 72 h and 120 h, and define a training, validation and testing datasets for the *R2F* and *R2R* tasks described in Section 2. An overview of the datasets and tasks is provided in Table A.1. Note that for the final model training, the valid set is used for training as well.

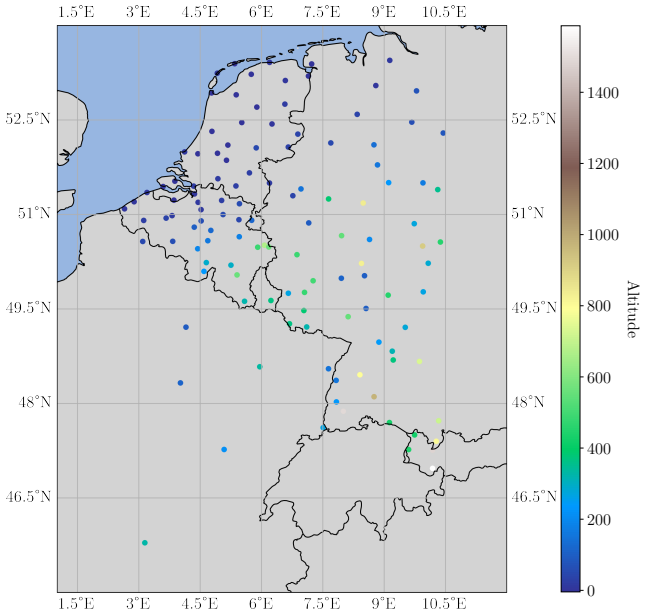

Figure A.1. Weather stations in the EUPPBench dataset with their corresponding altitude.

Table A.1. Sizes of the training, validation and test datasets in terms of the number of days for which a forecast is available. There is a forecast for 122 stations for each day, generated by either 11 or 51 ensemble members for the reforecasts and forecasts, respectively. RF_Test and F_Test denote the test datasets for the *R2R* and *R2F* task.

DATASET	SIZE	YEARS	REFORECAST?
TRAIN	2611	1997-2009	✓
VALID	836	2010-2013	✓
RF_TEST	733	2014-2017	✓
F_TEST	730	2017-2018	×

Table A.2. Description of auxiliary variables, their corresponding units, full name, and levels which they were measured at (Demaeyer et al., 2023). Temperature at 2 m is the target variable of interest for our study. **Processed** indicates if the variable has been accumulated, averaged or filtered over the past 6 h. Note that *cin* is not used in the final dataset since the data is incomplete.

SHORT NAME	UNITS	FULL NAME	LEVELS	PROCESSED?
T	K	TEMPERATURE	2 m, 850 hPa	
MX2T6	K	MAX TEMPERATURE	2 m	✓
MN2T6	K	MIN TEMPERATURE	2 m	✓
Z	$m^2\,s^{-2}$	GEOPOTENTIAL	500 hPa	
U	$m\,s^{-1}$	U COMPONENT OF WIND	10 m, 100 m, 700 hPa	
V	$m\,s^{-1}$	V COMPONENT OF WIND	10 m, 100 m, 700 hPa	
P10FG6	$m\,s^{-1}$	MAX WIND GUST	10 m	✓
Q	$kg\,kg^{-1}$	SPECIFIC HUMIDITY	700 hPa	
R	%	RELATIVE HUMIDITY	850 hPa	
CAPE	$J\,kg^{-1}$	CONVECTIVE AVAILABLE POTENTIAL ENERGY	—	
CIN[1]	$J\,kg^{-1}$	CONVECTIVE INHIBITION	—	
TP6	m	TOTAL PRECIPITATION	—	✓
CP6	m	CONVECTIVE PRECIPITATION	—	✓
TCW	$kg\,m^{-2}$	TOTAL COLUMN WATER	—	
TCWV	$kg\,m^{-2}$	TOTAL COLUMN WATER VAPOR	—	
TCC	$\in [0, 1]$	TOTAL CLOUD COVER	—	
VIS	m	VISIBILITY	—	
SSHF6	$J\,m^{-2}$	SURFACE SENSIBLE HEAT FLUX	—	✓
SLHF6	$J\,m^{-2}$	SURFACE LATENT HEAT FLUX	—	✓
SSR6	$J\,m^{-2}$	SURFACE NET SHORTWAVE (SOLAR) RADIATION	—	✓
SSRD6	$J\,m^{-2}$	SURFACE NET SHORTWAVE (SOLAR) RADIATION DOWNWARD	—	✓
STR6	$J\,m^{-2}$	SURFACE NET LONGWAVE (THERMAL) RADIATION	—	✓
STRD6	$J\,m^{-2}$	SURFACE NET LONGWAVE (THERMAL) RADIATION DOWNWARD	—	✓
SWV	$m^3\,m^{-3}$	VOLUMETRIC SOIL WATER	L1: 0 - 7 cm	
SD	m	SNOW DEPTH-WATER EQUIVALENT	—	
ST	K	SOIL TEMPERATURE	L1: 0 - 7 cm	

[1] Omitted in the final analysis due to missing data.

Table A.3. Further auxiliary variables, which are station specific except for yday, see Schulz & Lerch (2022) for details.

PREDICTOR	TYPE	DESCRIPTION
YDAY	TEMPORAL	COSINE AND SINE TRANSFORMED DAY OF THE YEAR
ID	—	UNIQUE ID ASSIGNED TO EACH STATION
LAT	SPATIAL	LATITUDE OF THE STATION
LON	SPATIAL	LONGITUDE OF THE STATION
ALT	SPATIAL	ALTITUDE OF THE STATION
OROG	SPATIAL	DIFFERENCE OF STATION ALTITUDE AND MODEL SURFACE HEIGHT OF NEAREST GRID POINT

A.2. Additional results

A.2.1. Details on hyperparameter optimization and model training

Following Rasp & Lerch (2018), a collection of 10 models is trained based on different random initalizations to address uncertainty during training and improve overall performance for all investigated post-processing models. The predictions, i.e., the distribution parameters obtained as the output of the resulting 10 models are averaged to generate the final prediction. We use an early stopping algorithm to enable faster training; if the CRPS does not increase for 10 epochs, we revert to the best model iteration and stop training. Model parameters are estimated using adaptive moment estimation with weight decay (*AdamW*) (Loshchilov & Hutter, 2017).

Table A.4 shows the results of a grid search for the GAT model. Note that d_{max} was not included in the grid search, however preliminary testing showed that $100\,\text{km}$ delivered good results. Further, the DS and SMRY models were optimized using the same hyperparameter grid as for the GAT model. Similar to the approach for the graph based models, the DRN model was also optimized using a grid search of the relevant hyperparameters, see also the model descriptions in Rasp & Lerch (2018); Schulz & Lerch (2022), and Höhlein et al. (2024). Training times ranged from a few minutes for the DRN to up to an hour for the GAT-based models on one NVIDIA P40 GPU. Note that in contrast to the computational costs of all post-processing methods are negligible compared to the costs of obtaining the raw forecasts by running ensembles of NWP models.

Table A.4. Choice of hyperparameters of the GAT model. The column 'Optimized?' indicates whether the hyperparameters were optimized based on the validation dataset using a grid search.

Parameter	24 h	72 h	120 h	**Optimized?**
maximal distance (d_{max})	100 km	100 km	100 km	
batch size	8	8	8	
training epochs	31	42	35	✓
learning rate	0.0002	0.0001	0.0005	✓
embedding dimension	20	20	20	
hidden channels (GNN)	265	128	64	✓
GNN Layers	2	2	1	✓
attention heads	8	8	8	✓
deep set layers (in)	3	3	3	
deep set layers (out)	2	2	2	
deep set hidden channels	same as "hidden channels (GNN)"			(✓)

A.2.2. Additional results

To compare the different post-processing models, we report the average CRPS for the two tasks (*R2R* and *R2F*) and all models in Table A.5, along with the average length of the prediction interval (PI length) based on a nominal level of $N - 1/N + 1$, where N is the number of ensemble members. This evaluates to $96.15\,\%$ and $83.33\,\%$ for the *R2F* and *R2R* task, respectively. Overall, qualitatively similar results are obtained for the two tasks, with similar rankings and relative improvements of the GAT model over the alternative specification of GNN models and the DRN model.

To assess the statistical significance of score differences, we use Diebold-Mariano tests (Diebold & Mariano, 1995) of equal predictive performance. The test is conducted for each combination of two models and separately for the considered lead times, with the null hypothesis of equal predictive performance at a given station. The test statistic is

$$t = \sqrt{n}\frac{\bar{S}_n^F - \bar{S}_n^G}{\hat{\sigma}_n}, \text{ where } \hat{\sigma}_n = \frac{1}{n}\sum_{i=1}^{n}\left(S(F_i, y_i) - S(G_i, y_i)\right)^2,$$

and $\bar{S}^F$ and $\bar{S}^G$ denote the corresponding mean scores for a fixed observation station and lead time for the two models' forecast distributions F and G and a corresponding test dataset of size n. Under the assumption of equal predictive performance, the distribution of t approximately follows a standard Gaussian distribution. In order to account for multiple testing, the Benjamini-Hochberg correction is applied (Benjamini & Hochberg, 1995), which corresponds to sorting the

Table A.5. Scores for the reforecast to reforecast (*R2R*) and reforecast to forecast (*R2F*) tasks. Scores are calculated per lead time, with the best CRPS scores highlighted in **bold**.

LEAD TIME	24 h			72 h			120 h		
METHOD	CRPS	PI LENGTH	PI COVER	CRPS	PI LENGTH	PI COVER	CRPS	PI LENGTH	PI COVER
R2R									
ENS	1.20	1.82	38.94	1.28	3.29	54.98	1.54	4.88	61.60
DRN	0.65	2.79	78.38	0.86	3.89	80.32	1.19	5.27	79.53
SMRY	0.66	2.91	79.73	0.87	4.04	82.37	1.18	5.48	81.51
DS	0.64	2.92	81.11	0.87	2.90	68.53	1.25	3.01	54.97
GAT	**0.63**	2.75	79.39	**0.85**	3.90	81.66	**1.17**	5.47	82.32
R2F									
ENS	1.12	2.66	56.06	1.18	4.72	72.90	1.38	7.14	81.16
DRN	0.61	4.26	94.87	0.79	5.90	96.37	1.11	7.99	95.82
SMRY	0.62	4.45	95.53	0.79	6.17	97.01	1.10	8.31	96.64
DS	0.61	4.41	95.72	**0.78**	4.43	89.87	1.14	4.56	77.79
GAT	**0.60**	4.16	95.04	**0.78**	5.93	96.42	**1.09**	8.27	96.80

p-values of the per-station tests in ascending order and selecting the corrected significance level as

$$p^* = \max(p_i \,|\, p_i \leq \frac{\alpha i}{M}).$$

For all p-values smaller or equal to p^* the null hypothesis is rejected. Results are reported in Table A.6 and indicate that the GAT models' scores tend to be significantly better those of DRN at up to 38% of the stations, while the null hypothesis is never rejected in favor of the DRN model. For longer lead times, the fraction of stations with significant score differences tends to decrease, and overall, qualitatively similar results can be observed for the two tasks.

Table A.6. Percentage of combinations of stations showing statistically significant differences in terms of the CRPS after applying the Benjamini–Hochberg correction at a nominal level of 0.05. Two-sided test were conducted, the table shows the ratio of stations for which the null hypothesis of equal predictive performance was rejected in favor of the model in the row, while comparing it to the model in the column.

Lead time	24 h				72 h				120 h			
Method	DRN	SMRY	DS	GAT	DRN	SMRY	DS	GAT	DRN	SMRY	DS	GAT
R2R												
DRN	—	29.2	0.0	0.0	—	11.7	0.8	0.0	—	1.7	84.2	0.0
SMRY	2.5	—	0.0	0.0	5.8	—	0.0	0.0	5.0	—	89.2	0.0
DS	18.3	36.7	—	1.7	3.3	2.5	—	0.0	0.8	0.8	—	0.0
GAT	30.0	70.0	23.3	—	20.8	45.8	51.7	—	16.7	22.5	97.5	—
R2F												
DRN	—	30.0	0.0	0.0	—	22.5	0.0	1.7	—	8.3	50.8	0.0
SMRY	7.5	—	5.0	0.0	10.0	—	0.0	3.3	10.0	—	55.0	0.0
DS	10.8	40.8	—	2.5	20.0	28.3	—	10.0	1.7	3.3	—	0.0
GAT	38.3	50.0	24.2	—	14.2	35.8	0.0	—	10.8	11.7	72.5	—

A.2.3. PIT HISTOGRAMS

To assess the calibration of the different post-processing approaches, we use probability integral transform (PIT) histograms. The PIT $F(y)$ is the value of the predictive CDF F, evaluated at the T2M observation y. In our case, the predictive distribution is Gaussian and thus $F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right)$ is the PIT. If the model is calibrated, meaning the realizing T2M observation is indistinguishable from a random draw from the forecast distribution, the PIT values should follow a uniform distribution $\mathcal{U}(0,1)$, and the visual inspection of histograms of the PIT values can point to different kinds of mis-calibration. For example, histograms that follow a U-shape indicate that the forecast is underdispersive (i.e., the observation too often

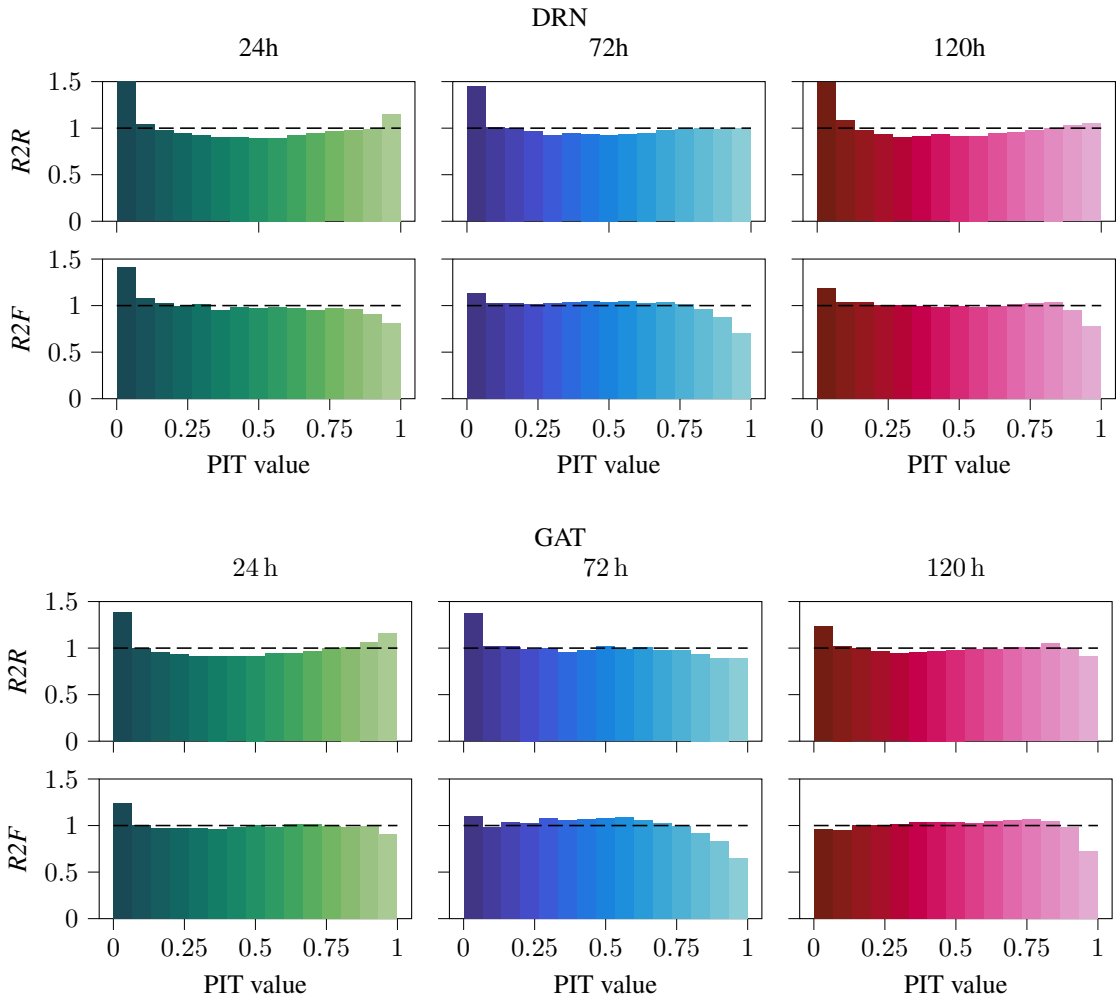

Figure A.2. PIT histograms of the post-processed forecasts of the DRN and GAT model for 24 h, 72 h and 120 h lead times based on the *R2R* and *R2F* tasks.

falls outside a plausible predicted range). Figure A.2 shows PIT histograms of the DRN and GAT models for the different lead times. All PIT histograms resemble an uniform distribution fairly well, however, for the reforecast data there exists a spike for the lower PIT values and larger PIT values are under-presented, specifically for the *R2F* task. Overall, only minor differences between the PIT histograms of the DRN and the GAT model can be observed.

A.2.4. Feature Importance

To identify the most important input features, we employ a permutation importance approach, which operates on the fundamental assumption that an input feature's importance can be determined by measuring the impact of randomly shuffling it on the model's performance. If an input variable is important, the predictive performance deteriorates notably after permuting it, while for unimportant variables, performance remains relatively unchanged. This can be due to the variable being generally unimportant for the task at hand, or the redundancy of the variable, meaning the information of this variable is already captured by other variables through multicollinearities (McGovern et al., 2019). The main advantage is that the model does not have to be retrained each time, saving computational resources. However, colinearities or interactions between variables cannot be captured.

Following Höhlein et al. (2024), we employ a two-step permutation is employed to first permute the feature across the time dimension and subsequently across the station (s) and ensemble member (n) dimension to evaluate the importance of an

input variable i. Let

$$\mathbf{X}_t = \{\mathbf{x}_{t,s,n} | s = 1, \ldots, S;\, n = 1, \ldots, N\}$$

denote the entire dataset at time step t, where $\mathbf{x}_{t,s,n}$ is a vector in $\mathbb{R}^P$ describing the prediction at station s and time t, made by ensemble member n, and P is the total number of input features. To simplify notation, we omit the index for the i-th feature, however note that the following transformations are only applied to the i-th dimension of $\mathbf{x}_{t,s,n}$. First, the data is permuted along the time dimension, according to a permutation π. Second, for each time-stamp t, the feature of interest is permuted along the station and ensemble member dimension together. Therefore,

$$\Pi(\mathbf{X}_t) = \{\mathbf{x}_{\pi(t),\pi_t(s,n)} | s = 1, \ldots, S;\, n = 1, \ldots, N\} \tag{1}$$

is the permuted feature set, which is then used to generate the graphs, as detailed in Section 3.3.1. The two-stage shuffling is designed to maintain certain structural information in the graph, such as ensuring that each station ID appears an equal number of times each day, irrespective of the shuffling. The importance of each feature is calculated by comparing the mean CRPS of the permuted dataset and to the original one computed on the non-permuted data via

$$\mathrm{Imp}(i) = \frac{\overline{\mathrm{CRPS}}(\mathbf{F}|\Pi_i(\mathbf{X}), \mathbf{Y}) - \overline{\mathrm{CRPS}}(\mathbf{F}|\mathbf{X}, \mathbf{Y})}{\overline{\mathrm{CRPS}}(\mathbf{F}|\mathbf{X}, \mathbf{Y})}. \tag{2}$$

The importance of feature i is estimated by evaluating Equation (2) 10 times using a different training run of a single GNN model.

Figure A.3 shows the feature importance for the $24\,\mathrm{h}$, $72\,\mathrm{h}$ and $120\,\mathrm{h}$ lead times for the two tasks. Note that the feature importances are normalized to allow for a better comparison. Not surprisingly, the top 3 most important predictor variables (T2M, MX2T6, MN2T6) all concern the 2-m temperature and account for about $61.8\,\%$ of the total importance together (in the *R2F* task). Even though the distribution of importances across these three variables varies substantially depending on the lead time, the total importance always sums up to $61.8 \pm 1\,\%$. For the *R2F* task, the temperature variables are followed by the level 1 soil temperature (STL1), which is recorded in a a depth of 0 - $7\,\mathrm{cm}$. As the lead time increases, the importance of soil temperature increases as well. Subsequently, two station-specific features follow, where ID refers to the station identifier, which is arbitrarily assigned in the beginning, but is mapped via the embedding layer to a 20-dimensional vector. Using this embedding, the model encodes station specific information in the node id during training. ALT refers to the altitude of the station location. Qualitatively similar results are obtained for the *R2R* task, with a change in the raking between the soil temperature and the station identifier being the most notable difference in the most important predictors. However, note that these feature importances should be interpreted with care, as the quality of the prediction made by the NWP model varies across the features. Thus low importance can also be due to decreased forecasting performance by the NWP model, instead of the variable being irrelevant for the task. For details, see also the corresponding discussions in Rasp & Lerch (2018) and Schulz & Lerch (2022).

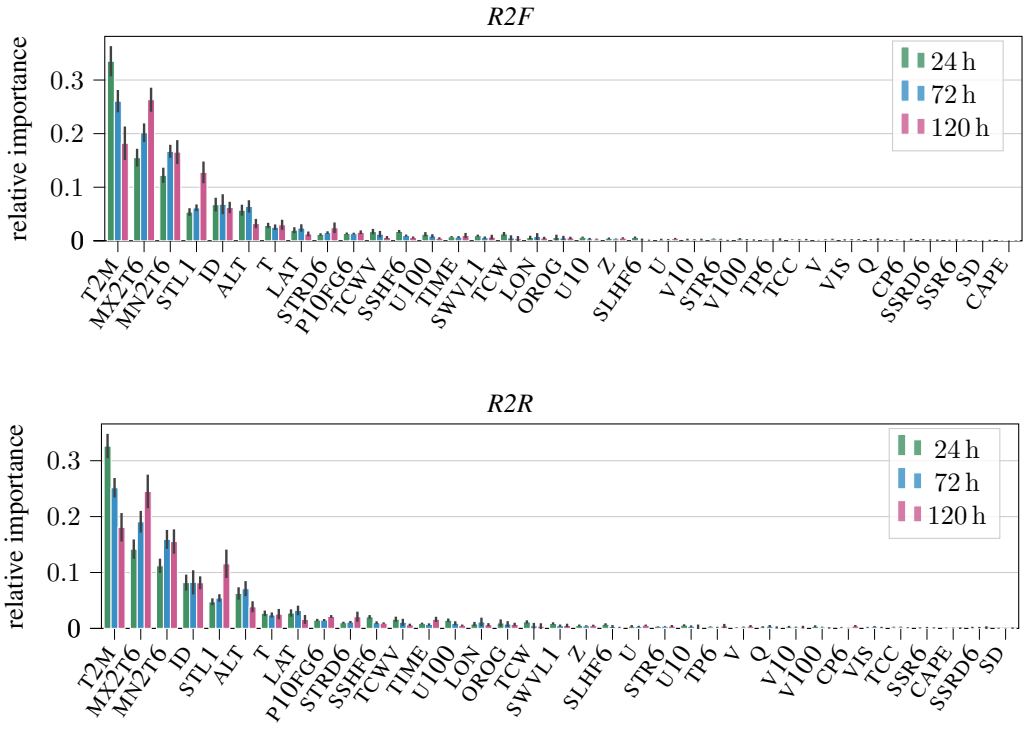

Figure A.3. Relative feature importance of the GAT model for the *R2F* (top) and the *R2R* task (bottom). Error bars show the standard deviation, which is calculated based on 10 training runs of the individual GNNs.