Robust Calibration For Improved Weather Prediction Under Distributional Shift

Sankalp Gilda
ML Collective
sankalp.gilda@gmail.com

Neel Bhandari
Department of Computer Science and Engineering
RV College of Engineering
neelbhandari.cs18@rvce.edu.in

Wendy Mak
ML Collective
wwymak@gmail.com

Andrea Panizza
ML Collective
andrea.panizza75@gmail.com

Abstract

In this paper, we present preliminary results on improving out-of-domain weather prediction and uncertainty estimation as part of the `Shifts Challenge on Robustness and Uncertainty under Real-World Distributional Shift` challenge. We find that by leveraging a mixture of experts in conjunction with an advanced data augmentation technique borrowed from the computer vision domain, in conjunction with robust *post-hoc* calibration of predictive uncertainties, we can potentially achieve more accurate and better-calibrated results with deep neural networks than with boosted tree models for tabular data. We quantify our predictions using several metrics and propose several future lines of inquiry and experimentation to boost performance.

1 Introduction

Machine learning is increasingly being applied across several domains, finding ever new applications in the real world. A key assumption held by most of these models is that training and test data are IID; this assumption rarely holds up to scrutiny in the real-world, where the models must process unseen and unpredictable distributions. This leads to several machine learning models providing unsatisfactory performance in production. The development of models that are robust to distributional shifts, therefore, is an important goal to work towards.

A prime exemplar for a field where distributional shifts are abundant over time is weather prediction. Weather prediction requires that models provide consistently satisfactory performance across both as a function of space (latitude, longitude, climate) and time (of day, month, year). While there have been several works in the field of weather prediction, the `Shifts Challenge on Robustness and Uncertainty under Real-World Distributional Shift` presents a unique opportunity to develop and apply models that are robust to such effects and also yield sensible uncertainty estimates. We present here an empirical study on the effects of regularization, calibration, and data augmentation in a multi-domain training environment.

Our main contributions are thus. First, we leverage a mixture density network (MDN, 3; 6; 11; 12) with $\beta-$likelihoods and demonstrate competitive performance relative to two commonly used boosted-tree models, `NGBoost` (8) and `CatBoost` (21). Second, we demonstrate the necessity of regularization while training the MDN. Specifically, we use moment exchange (MoEx, 17), a data augmentation method originally developed for use in computer vision (CV), to regularize our MDN successfully. Third, we demonstrate the necessity of calibrating predictions, and utilize a state-of-the-art *post-hoc* calibration method (CRUDE, 26) to that end. Fourth, we illustrate that the inverse variance weighing

Preprint. Under review.

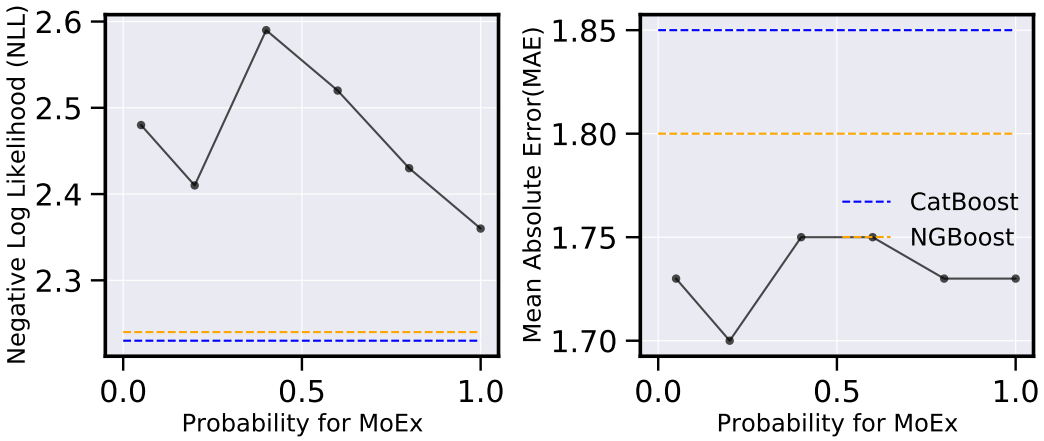

Figure 1: Impact of varying the probability parameter p in MoEx. p is the probability that a given sample will be augmented. **Left:** Negative log likelihood of *robust, post-hoc* predictions on `TEST` vs. p and mean absolute error. **Right:** Mean absolute error instead of NLL. For both plots, lower is better. We denote by dashed orange and blue lines the respective predicted metrics from `NGBoost` and `CatBoost` (see last lines in Tables 2 and 3, respectively).

method, commonly used to combine predictions from members of an ensemble, improves predicted means but results in poorly calibrated uncertainties. Finally, we demonstrate, for the first time with a tabular dataset, the improvement in predicted negative log likelihood (NLL) resulting from *robust*, domain-aware calibration (23).

2 Data

We employ the `Weather Prediction` dataset developed by Yandex Research for the Shifts Challenge. This dataset comprises several meteorological features from three major global weather prediction models, as well as air temperatures 2 metres above ground (`fact_temperature`), for a diverse set of latitudes and longitudes. The goal of this competition is to predict air temperature at 2 meters above ground, given all available weather station measurements and multiple weather forecast model predictions. While the training data has information about location on earth and the timestamp when predictions from the 3 global forecasting systems were made, these meta-data are assumed to be missing for the test set.

There are two specific set of feature categories that make up this dataset: weather-related features and meteorological features. Weather related features consist of sun evaluation at the current location, climate values of temperature, pressure and topography. Meteorological parameters are details based on pressure and surface level data from the weather prediction models. There are five climate types associated with the data: Tropical, Dry, Mild Temperate, Snow and Polar. The data are split into *training*, *development* and *evaluation* sets based on the climate and time of year, in order to provide a clear distributional shift in the weather data. The *development* set is first sub-divided into ID-development and OoD-development (in-distribution an out of distribution, respectively), and is designed to simulate the *evaluation* set. In this paper, we only use the first two datasets as these were the only ones released during the first phase of the competition. The training data consists of features with climate types as Tropical, Dry and Mild Temperature. This data is recorded from the 1^{st} of September 2018, to the 8^{th} of April, 2019. dev-ID has the same climate types and time of year (first and last thirds of the year) as the training data, whereas dev-OoD consists of samples only with the climate type Snow, and during the middle third of the year. The evaluation dataset, of course, has been shifted based on climate and time. It consists of features with Snow and Polar climate types, recorded from the 14^{th} May, 2019 to the 8^{th} of July,2019.[1] The training, dev-ID and dev-OoD datasets show a clear distributional shift, presenting a challenge to develop models robust to these shifts and competent at generalisation to OoD data in the real world.

[1]We mention this only for sake of exposition but remind the reader that we do not actually use the evaluation dataset in this work.

3 Method

1. We divide the given training dataset `T_ALL` with $\sim$3.1 million samples into six disjoint datasets (`T_E1`, `T_E2`, `T_E3`, `T_E4`, `T_E5`, and `T_E6`) based on `climate` and `week_of_year`. We use the ID part of the development dataset as our validation data `V_ALL`, and the OoD part as the test set `TEST`. We divide the former it into `V_E1`, `V_E2`, `V_E3`, `V_E4`, `V_E5`, and `V_E6`. The Es stand for environment, to highlight the fact that each of the six datasets come from a different domain/environment. From each of these 14 datasets (7 training and 7 validation), as well as from `TEST`, we drop the first six columns as these are to be treated as meta-data and not expected to be present in the final evaluation dataset.
2. We now have seven unique sets of training-validation splits upon which we we train separate models. We scale each of these separately. We standardize the input features (to mean of 0 and standard deviation of 1) of the training datasets (`T_E1` through `T_E6`, and `T_ALL`), and then re-scale them to lie between 0 and 1. We use the derived statistics to re-scale `V_E1` through `V_E6`, and `V_ALL` as well as the respective copies of the test set `TEST`. We similarly normalize the six output columns of `fact_temperature`, and then follow this up by min-max scaling them such that the training set values lie between 0.1 and 0.9. This is because of our models, the MDN (visualized in Figure 2) uses a β-likelihood for the output variable, which requires it to lie between 0 and 1. We leave a little buffer zone of 0.1 on each size to allow for OoD predictions.
3. We choose the hyperparameters for `NGBoost`[2] and `CatBoost`[3] by trial-and-error. We use these two models as they allow to derive both aleatoric and epistemic uncertainties easily. For each type of model, we create 10 models with different seeds, and average the predictions from each such that the final mean on a test sample is the mean of the 10 predicted means, the final aleatoric uncertainty is the mean of the 10 predicted aleatoric uncertainties, and the final epistemic uncertainty is the variance of the 10 predicted means.
4. For the MDN, we use 5 mixture components, negative log-likelihood (NLL) as the loss function, LAMB (25) as the optimizer, and a batch size of 512. We use a cosine decay learning rate scheduler (22) varying between $1e-3$ and $1e-4$, with 2 cycles each of length 15 epochs. We save and revert to model weights which result in the lowest loss values on the respective validation sets.
5. For the MDN, we leverage moment exchange (MoEx, 17) to augment training data online. MoEx uses two hyperparameters: *p*, the probability that a given sample will be augmented, and λ, the weight of a sample in a binary mixture of itself with another randomly selected sample (see original paper for details). We pick $p = 0.20$ and draw lambda from a peaked β distribution, such that it is more often than note close to 0.5 ($\lambda \sim \beta(100, 100)$). In addition we use gradient centralization (GC, 24) and stochastic weight averaging (SWA, 15) to smooth the loss landscape and improve generalization.
6. Next, we combine the six sets of predictions from each of the individual training sets (`T_E1` through `T_E6`) by inverse variance scaling.
7. We *post-hoc* calibrate all predicted aleatoric uncertainties using CRUDE (26), a state-of-the-art method for regression problems.
8. When the training set is `T_ALL`, we also calibrate the predicted aleatoric uncertainties on a per-domain basis (23). Specifically, after training a model on `T_ALL`, instead of using `V_ALL` as the calibration set, we instead use `V_E1` through `V_E6` as individual calibration sets. For each of the six runs, we store the Gaussian NLL (7) derived from the calibrated aleatoric uncertainties on the respective validation/calibration sets, then find the `V_E`i with the minimum NLL – this is the calibration set with respect to which *post-hoc* calibration will result in maximum gain.
9. Finally, to test the sensitivity of MDN with MoEx on the probability of augmentation *p*, we run experiments with $p = 0.05, 0.40, 0.60, 0.80, 1.00$; see Figure 1.

[2] `n_estimators`=500, `max_depth`=12, `colsample_by_node`=0.3, `subsample`=0.8, `eta`=0.1, `num_parallel_trees`=3, `min_child_weight`=40, `gamma`=10, `reg_lambda`=5, `reg_alpha`=5, `distribution`='normal'.

[3] `iterations`=500, `l2_leaf_reg`=10, `border_count`=254, `depth`=10, `learning_rate`=0.03, `use_best_model`=True, `loss`='RMSEWithUncertainty'.

Table 1: Relative performance of MDN+MoEx on the OoD test set (TEST), when trained individually on training datasets from the six environments (T_E1 through T_E6), when ensembling these results using inverse variance scaling, and when trianing on the combined dataset T_ALL. The first two lines in each field correspond to raw and calibrated predictions. For the last cell with T_ALL, the last line corresponds to robustly calibrated predictions. 'M.T.' stands for mild temperate. For 'Inverse Variance', the second row is not the results of calibration via CRUDE, but the inverse variance weighted means of the means and standard deviations from the calibrated predictions from the six domains (i.e., a weighted average of the second rows in the top six cells).

Domain	Dataset	MAE (↓)	RMSE (↓)	BE (| ↓ |)	IS (| ↓ |)	ACE (| ↓ |)	NLL (↓)
M.T.	T_E1	2.32	3.11	-0.40	0.04	-0.23	3.00
Early		2.32	3.11	-0.42	0.05	-0.29	3.46
M.T.	T_E2	2.02	2.69	-0.44	0.04	-0.25	3.09
Late		2.03	2.70	-0.49	0.04	-0.23	2.97
Dry	T_E3	2.38	3.28	0.51	0.04	-0.30	3.6
Early		2.38	3.27	0.50	0.03	-0.24	3.1
Dry	T_E4	2.05	2.99	-0.01	0.03	-0.19	2.80
Late		2.06	3.08	0.19	0.03	-0.17	2.69
Tropical	T_E5	3.06	3.90	1.87	0.06	-0.21	3.47
Early		3.09	3.93	1.93	0.07	-0.30	4.84
Tropical	T_E6	2.66	3.48	0.21	0.05	-0.26	3.61
Late		2.65	3.47	0.13	0.06	-0.30	4.08
	Inverse	1.81	2.41	0.14	0.04	-0.44	7.39
	Variance	1.85	2.48	0.25	0.05	-0.45	8.50
		1.74	2.33	-0.15	0.03	-0.08	2.26
All	T_ALL	1.74	2.33	-0.12	0.03	-0.18	2.50
		1.73	2.33	0.12	0.03	-0.17	2.48

For tracking our experiments and logging all metrics, we leverage Weights & Biases (2)[4].

4 Results

We compare all predictions using six metrics: mean absolute error (MAE), root mean squared error (RMSE), bias error (BE), interval sharpness (IS, 13; 4; 11; 12), average calibration error (ACE 13; 4; 11; 12), and Gaussian negative log likelihood (NLL, 7). For experiments with MDN as the model, we show results in Table 1. With NGBoost and CatBoost as the models, we show results in Tables 2 and 3, respectively. A few observations become apparent:

1. The variance in metrics when trained on individual domains is quite high. In a real-life scenario of domain generalization, when it is difficult or even impossible to say which one of the training environments at hand might be the most similar to a given test environment, such high variance is undesirable.
2. While inverse variance ensembling – one of the most common methods of ensembling predictions – might improve deterministic metrics (MAE, RMSE, MAE), they invariably deteriorate the probabilistic ones (ACE and NLL).
3. In most cases, calibration improves predictions. In the handful of cases where it results in worse performance (T_E5 and T_E6 cells in Tables 1, 2 and 3, and 'inverse variance' cell in Table 1), this can be attributed to the domain shift between the source and target domains. T_E5 and T_E6 datasets are from the tropical environment and are 'far away', semantically, from T_TEST which is in snowy climes. Consequently, *post-hoc* calibrating predictions from datasets that are already poor representatives of a test set does more harm than good.

[4] https://wandb.ai/site

4. From Table 4, first rows of all cells, we see that the MDN's performance is mostly agnostic to the choice of the MoEx hyperparameter p (judging by say MAE and NLL), except for high values of p (see the $p = 1.00$ cell). However, calibratiion – even more so, robust calibration – removes, to a large extent, the variance in performance, a desirable quality in domain generalization (see second and third rows of all cells). This strengthens further our case that both calibration, and when in a multi-environment setting, robust calibration, should be an indispensable tool in a researcher's toolkit when making probabilistic predictions.
5. From Figure 1 we make two observations. First is that while (robustly calibrated) MAE with the MDN is lower than the (robustly calibrated) MAE from at all values of p, the opposite is true for NLL; we relegate further exploration of this to future work. Second, results based on the dataset under consideration suggest that $p = 1$ is a good estimate for the MoEx hyperparameter, provided it that predictions are followed by robust calibration.

5 Future Work

This article represents only intermediate results based on the training and development sets available at the time of writing. As the immediate next step we will make predictions on the recently open-sourced evaluation dataset, and compare our results with those of the winners of the competition. We will also experiment with other neural network and boosted tree architectures, such as deep evidential regression (19), `XGBoost` (5), and `LightGBM` (16). We will experiment with methods to ensemble predictions from different architectures; recent work has shown that such a hybrid ensemble can provide superior predictions than either machine learning- or deep learning-based models alone (14). For a fairer comparison, we will also extensively optimize the hyperparameters of all models under consideration. We will experiment with other domains of training besides supervised, such as domain adaptation (10), imbalanced risk minimization (IRM, 1), and feature calibration (20)

References

[1] ARJOVSKY, M., BOTTOU, L., GULRAJANI, I., AND LOPEZ-PAZ, D. Invariant Risk Minimization. *arXiv e-prints* (July 2019), arXiv:1907.02893.

[2] BIEWALD, L. Experiment tracking with weights and biases, 2020. Software available from wandb.com.

[3] BISHOP, C. M. Mixture density networks.

[4] BRACHER, J., RAY, E. L., GNEITING, T., AND REICH, N. G. Evaluating epidemic forecasts in an interval format. *PLOS Computational Biology 17*, 2 (2021), e1008618.

[5] CHEN, T., AND GUESTRIN, C. Xgboost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Aug 2016).

[6] CHOI, S., LEE, K., LIM, S., AND OH, S. Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling, 2017.

[7] CHUNG, Y., CHAR, I., GUO, H., SCHNEIDER, J., AND NEISWANGER, W. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254* (2021).

[8] DUAN, T., AVATI, A., DING, D. Y., THAI, K. K., BASU, S., NG, A. Y., AND SCHULER, A. Ngboost: Natural gradient boosting for probabilistic prediction, 2020.

[9] FIEDLER, J. Simple modifications to improve tabular neural networks, 2021.

[10] GANIN, Y., AND LEMPITSKY, V. Unsupervised Domain Adaptation by Backpropagation. *arXiv e-prints* (Sept. 2014), arXiv:1409.7495.

[11] GILDA, S., DRAPER, S. C., FABBRO, S., MAHONEY, W., PRUNET, S., WITHINGTON, K., WILSON, M., TING, Y.-S., AND SHEINIS, A. Uncertainty-Aware Learning for Improvements in Image Quality of the Canada-France-Hawaii Telescope. *Monthly Notices of the Royal Astronomical Society* (11 2021). stab3243.

[12] GILDA, S., TING, Y.-S., WITHINGTON, K., WILSON, M., PRUNET, S., MAHONEY, W., FABBRO, S., DRAPER, S. C., AND SHEINIS, A. Astronomical Image Quality Prediction based on Environmental and Telescope Operating Conditions. *arXiv e-prints* (Nov. 2020), arXiv:2011.03132.

[13] GNEITING, T., AND RAFTERY, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association 102*, 477 (2007), 359–378.

[14] GORISHNIY, Y., RUBACHEV, I., KHRULKOV, V., AND BABENKO, A. Revisiting deep learning models for tabular data, 2021.

[15] IZMAILOV, P., PODOPRIKHIN, D., GARIPOV, T., VETROV, D., AND WILSON, A. G. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407* (2018).

[16] KE, G., MENG, Q., FINLEY, T., WANG, T., CHEN, W., MA, W., YE, Q., AND LIU, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems* (2017), I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc.

[17] LI, B., WU, F., LIM, S.-N., BELONGIE, S., AND WEINBERGER, K. Q. On feature normalization and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 12383–12392.

[18] LI, B., WU, F., WEINBERGER, K. Q., AND BELONGIE, S. Positional normalization. In *Advances in Neural Information Processing Systems* (2019), pp. 1620–1632.

[19] MEINERT, N., AND LAVIN, A. Multivariate deep evidential regression. *arXiv preprint arXiv:2104.06135* (2021).

[20] PARK, D., SONG, H., KIM, M., AND LEE, J.-G. Task-agnostic undesirable feature deactivation using out-of-distribution data. *Advances in Neural Information Processing Systems 34* (2021).

[21] PROKHORENKOVA, L., GUSEV, G., VOROBEV, A., DOROGUSH, A. V., AND GULIN, A. Catboost: unbiased boosting with categorical features, 2019.

[22] SMITH, L. N. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2017), pp. 464–472.

[23] WALD, Y., FEDER, A., GREENFELD, D., AND SHALIT, U. On Calibration and Out-of-domain Generalization. *arXiv e-prints* (Feb. 2021), arXiv:2102.10395.

[24] YONG, H., HUANG, J., HUA, X., AND ZHANG, L. Gradient centralization: A new optimization technique for deep neural networks, 2020.

[25] YOU, Y., LI, J., REDDI, S., HSEU, J., KUMAR, S., BHOJANAPALLI, S., SONG, X., DEMMEL, J., KEUTZER, K., AND HSIEH, C.-J. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. *arXiv e-prints* (Apr. 2019), arXiv:1904.00962.

[26] ZELIKMAN, E., HEALY, C., ZHOU, S., AND AVATI, A. CRUDE: Calibrating Regression Uncertainty Distributions Empirically. *arXiv e-prints* (May 2020), arXiv:2005.12496.

A Tables

Table 2: Same as Table 1, but using `NGBoost` as the predictor.

Domain	Dataset	MAE (↓)	RMSE (↓)	BE (| ↓ |)	IS (| ↓ |)	ACE (| ↓ |)	NLL (↓)
M.T. Early	T_E1	2.09 2.01	2.77 2.68	-0.90 -0.66	0.03 0.03	-0.06 -0.00	2.35 2.31
M.T. Late	T_E2	1.92 1.88	2.57 2.52	-0.62 -0.46	0.03 0.03	-0.06 -0.02	2.29 2.25
Dry Early	T_E3	2.12 2.08	2.85 2.80	-0.80 -0.64	0.03 0.03	-0.03 0.02	2.41 2.38
Dry Late	T_E4	1.96 1.93	2.61 2.58	-0.38 -0.20	0.03 0.03	-0.03 0.01	2.31 2.29
Tropical Early	T_E5	4.31 4.51	5.56 5.82	3.57 3.84	0.11 0.11	-0.17 -0.16	2.89 2.91
Tropical Late	T_E6	4.44 4.60	5.65 5.85	3.72 3.92	0.11 0.11	-0.08 -0.02	2.89 2.91
	Inverse Variance	1.85 1.84	2.49 2.48	-0.20 -0.04	0.04 0.04	-0.35 -0.32	3.72 3.41
All	T_ALL	1.81 1.79 1.80	2.44 2.42 2.46	-0.28 -0.09 0.51	0.03 0.03 0.03	-0.02 0.01 0.00	2.23 2.22 2.24

Table 3: Same as Table 1, but using `CatBoost` as the predictor.

Domain	Dataset	MAE (↓)	RMSE (↓)	BE (| ↓ |)	IS (| ↓ |)	ACE (| ↓ |)	NLL (↓)
M.T. Early	T_E1	1.74 1.73	2.34 2.33	-0.19 -0.05	0.03 0.03	-0.02 -0.01	2.21 2.21
M.T. Late	T_E2	1.83 1.81	2.47 2.45	-0.30 -0.17	0.03 0.03	-0.04 -0.04	2.24 2.24
Dry Early	T_E3	1.80 1.79	2.40 2.39	-0.19 -0.12	0.03 0.03	-0.03 -0.01	2.27 2.25
Dry Late	T_E4	1.87 1.85	2.51 2.49	-0.48 -0.36	0.03 0.03	-0.05 -0.03	2.29 2.27
Tropical Early	T_E5	2.34 2.36	3.13 3.18	0.57 0.77	0.08 0.08	-0.24 -0.21	3.21 3.09
Tropical Late	T_E6	2.22 2.24	2.97 3.01	0.28 0.54	0.07 0.07	-0.22 -0.20	3.04 2.99
	Inverse Variance	1.83 1.82	2.45 2.44	0.01 0.16	0.05 0.05	-0.41 -0.40	5.59 5.30
All	T_ALL	1.88 1.86 1.85	2.56 2.53 2.52	-0.41 -0.27 -0.19	0.03 0.03 0.03	-0.02 -0.03 -0.02	2.24 2.24 2.23

Table 4: Performance of the MDN with MoEx augmentation at different values of the probability parameter p. We assume a $\lambda \sim 0.5$ throughout. Training set is `T_ALL`, validation set is `V_ALL`. In each cell, the first row contains metrics from raw predictions on `TEST`, second row from CRUDE-calibrated predictions using `V_ALL` as the calibration set, and the final row contains metrics when we leverage robust (per-domain) calibration as described in Section 3.

Probability (p)	MAE (↓)	RMSE (↓)	BE (| ↓ |)	IS (| ↓ |)	ACE (| ↓ |)	NLL (↓)
0.05	1.74	2.33	-0.15	0.03	-0.08	2.26
	1.74	2.33	-0.12	0.03	-0.18	2.50
	1.73	2.33	0.12	0.03	-0.17	2.48
0.20	1.72	2.29	-0.21	0.03	-0.10	2.27
	1.71	2.28	-0.11	0.03	-0.14	2.37
	1.70	2.29	0.16	0.03	-0.15	2.41
0.40	1.78	2.36	-0.32	0.03	-0.13	2.40
	1.76	2.35	-0.21	0.03	-0.17	2.51
	1.75	2.34	-0.12	0.03	-0.19	2.59
0.60	1.76	2.35	-0.07	0.03	-0.11	2.34
	1.76	2.35	-0.12	0.03	-0.16	2.45
	1.75	2.35	0.02	0.03	-0.17	2.52
0.80	1.73	2.33	-0.07	0.02	-0.10	2.30
	1.73	2.33	0.07	0.03	-0.15	2.41
	1.73	2.34	0.10	0.03	-0.15	2.43
1.00	1.79	2.38	-0.51	0.03	-0.06	2.27
	1.75	2.35	-0.31	0.03	-0.13	2.37
	1.73	2.33	-0.15	0.03	-0.12	2.36

B Figures

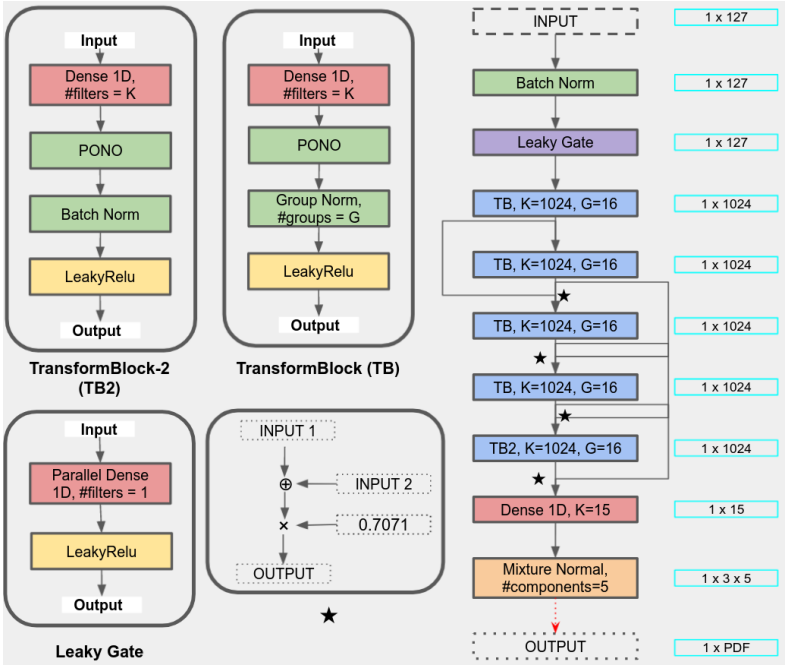

Figure 2: Architecture of the mixture density network (MDN, 3; 6). PONO is the positional normalization layer (18), and the use of LeakyGate is inspired by (9). We model the output variable `fact_temperature` conditioned on the input variables using a mixture of 5 β distributions.