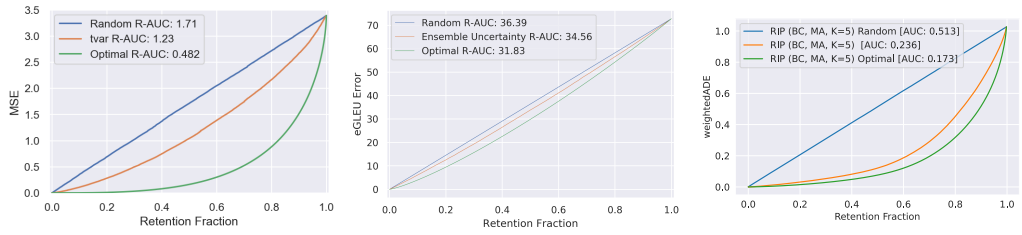
Appendices

A Assessment Metrics

As discussed in Section 2, in this work we consider robustness and uncertainty estimation to be two equally important factors in assessing the reliability of a model. We assume that as the degree of distributional shift increases, so should a model's errors; in other words, a model's uncertainty estimates should be correlated with the degree of its error. This informs our choice of assessment metrics, which must *jointly* assess robustness and uncertainty estimation.

One standard approach to jointly assess robustness and uncertainty are *error-retention curves* [12, 14], which plot a model's mean error over a dataset, as measured using a metric such as error-rate, MSE, eGLEU, cNLL, etc., with respect to the fraction of the dataset for which the model's predictions are used. These retention curves are traced by replacing a model's predictions with ground-truth labels obtained from an oracle in order of decreasing uncertainty, thereby decreasing error. Ideally, a model's uncertainty is correlated with its error, and therefore the most errorful predictions would be replaced first, which would yield the greatest reduction in mean error as more predictions are replaced. This represents a hybrid human-AI scenario, where a model can consult an oracle (human) for assistance in difficult situations and obtain from the oracle a perfect prediction on those examples.

The area under the retention curve (R-AUC) is a metric for jointly assessing robustness to distributional shift and the quality of the uncertainty estimates. R-AUC can be reduced either by improving the predictions of the model, such that it has lower overall error at any given retention rate, or by providing estimates of uncertainty which better correlate with error, such that the most incorrect predictions are rejected first. It is important that the dataset in question contains both a subset "matched" to the training data, and a distributionally shifted subset. Figure 4 provides example retention curves for the three tasks of the Shifts Dataset. In each figure, in addition to the uncertainty-based ranking, we included curves which represent "random" ranking, where uncertainty estimates are entirely non-informative, and "optimal" ranking, where uncertainty estimates perfectly correlate with error. These represent the lower and upper bounds on R-AUC performance as a function of uncertainty quality.



(a) Weather Prediction (b) Machine Translation (c) Vehicle Motion Prediction

Figure 4: Example error retention curves for the three tasks of the Shifts Dataset.

While clearly interpretable and intuitive, one concern that can be raised regarding error-retention curves is that they can be more sensitive to predictive performance than to the quality of uncertainty estimates, which can be seen in Figure 4b. This occurs on tasks where most errors have similar magnitude. Furthermore, for regression tasks, retention curves are dominated by noise in the targets (aleatoric uncertainty) at low retention fractions, when most systematic errors have already been detected. Therefore, in this work we propose another metric which jointly assesses robustness and uncertainty estimation.

First, we introduce the notion of an "acceptable prediction", which is a prediction whose error is acceptably small. This concept is natural for tasks with a non-binary notion of error, e.g., regression problems. For classification tasks, where predictions are already either correct or incorrect (acceptable/non-acceptable), this concept can be introduced by considering different levels of risk for different misclassifications. Formally, we say that a prediction is acceptable if an appropriate metric of error or risk $\mathcal{E}$ is below a *fixed* task-dependent error threshold T_e. For example, if temperature is predicted to within a degree of the ground truth, then it is acceptable. This allows us to mitigate

the issue of errors having similar magnitudes. This is expressed using via an indicator function as follows:

$$\mathcal{A}_{T_e}(\boldsymbol{x}) = \begin{cases} 1, \ \mathcal{E}(\boldsymbol{x}) \leq T_e \\ 0, \ \mathcal{E}(\boldsymbol{x}) > T_e \end{cases} \tag{1}$$

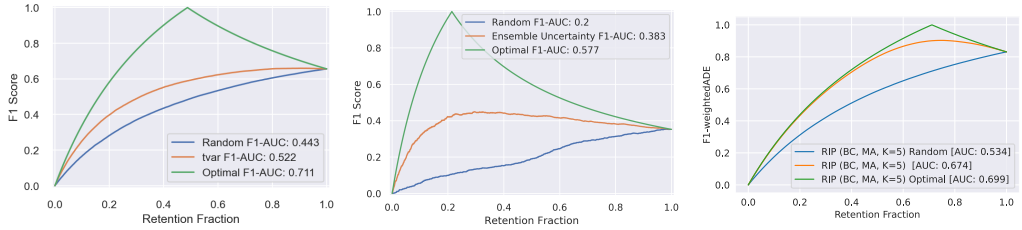
For a given dataset D and model, we first set an error threshold and determine which predictions are acceptable – this yields a set of "ground-truth" acceptability labels $\mathcal{A}_{i=1}^{N}$. We can now use these acceptability labels to assess whether the model's *estimates of uncertainty* $\mathcal{U}(\boldsymbol{x})$ can be used to indicate whether a prediction is acceptable. If the uncertainty score is greater than a threshold T_u, then we consider the prediction to be poor, if the uncertainty score is lower than this threshold, the prediction is considered to be acceptable.

$$\hat{\mathcal{A}}_{T_u}(\boldsymbol{x}) = \begin{cases} 1, \ \mathcal{U}(\boldsymbol{x}) \leq T_u \\ 0, \ \mathcal{U}(\boldsymbol{x}) > T_u \end{cases} \tag{2}$$

Next, given the true acceptability labels $\{\mathcal{A}_{T_e}(\boldsymbol{x}_i)\}_{i=1}^{N}$ and the threshold-conditional indicators $\{\hat{\mathcal{A}}_{T_u}(\boldsymbol{x})\}_{i=1}^{N}$ we sweep through all uncertainty scores in a dataset $\{\mathcal{U}(\boldsymbol{x}_i)\}_{i=1}^{N}$ in decreasing order and use them as thresholds to F1 for classifying whether a prediction is actually acceptable or not based on the uncertainty. Formally, this is done as follows:

$$P_i = \frac{\sum_{j=1}^{N} \mathcal{A}_{T_e}(\boldsymbol{x}_j) \cdot \hat{\mathcal{A}}_{\mathcal{U}_i}(\boldsymbol{x}_j)}{N - i}, \ R_i = \frac{\sum_{j=1}^{N} \mathcal{A}_{T_e}(\boldsymbol{x}_j) \cdot \hat{\mathcal{A}}_{\mathcal{U}_i}(\boldsymbol{x}_j)}{\sum_{j=1}^{N} \mathcal{A}_{T_e}(\boldsymbol{x}_j)}, \ \text{F1}_i = \frac{2 \cdot P_i \cdot R_i}{P_i + R_i} \tag{3}$$

where we use $N - i$ because we sort uncertainties from largest ($\mathcal{U}_1$) to smallest ($\mathcal{U}_N$). We then plot $\{\text{F1}_i\}_{i=1}^{N}$ against $1 - \frac{i}{N}$, i.e., the fraction of data we are classifying as acceptable, which we refer to as the retention fraction. This yields the following curves for the three Shifts tasks:



(a) Weather Prediction (b) Machine Translation (c) Motion Prediction

Figure 5: Examples of F1-Retention curves for the three tasks of the Shifts Dataset.

Here we plot the uncertainty-based F1-retention curves for all datasets. On each figure, we plot both the uncertainty-derived curves as well as the "random" and "optimal" baselines, where uncertainties are either completely uncorrelated or perfectly correlated with errors, respectively. Better models have a higher area under this F1-retention curve (F1-AUC). The predictive performance of the model defines the starting point at 100% retention – better models start higher. Thus, area under the F1 curve can be increased by having a model which yield better predictions or by improving the correlation between uncertainty and error. Note, in contrast to Figure 4b, the quality of the ranking affects area under the curve far more than for error-retention curves. Thus, this metric is especially useful when errors have similar magnitudes.

Finally, it is necessary to point out that area under the error-retention curve and F1-retention curve is a *summary statistic* which describes possible *operating points*. We can specify a particular operating point, such as 95% retention, and evaluate the error or F1 at that point for comparison. This is also an important figure, as all models work at a particular operating point which satisfies task-specific desiderata. In this work the desiderata for all tasks will be to not reject more than 5% of the input data.

B Shifts Dataset General Datasheet

Here we describe the motivation, uses, distribution as well as the maintenance and support plan for the Shifts Dataset as whole in the *datasheet for datasets* format [76]. The details of the composition, collection and pre-prossessing of each component dataset are provided in appendices C-E.

Motivation As discussed at length in the main body of the paper, the primary goal for the creation of the Shifts Dataset was the evaluation of uncertainty quantification models and robustness to distributional shift on a range of large-scale, industrial tasks spanning multiple modalities. To this end, Yandex Research, in collaboration with the Yandex.Translate, Yandex.Weather services and Yandex Self-Driving Group created the Shifts Dataset. As the dataset creation was done by Yandex teams, it was therefore funded by Yandex.

Uses The dataset is used as part of the Shifts Challenge which was organized as part of NeurIPS2021, which was organized around this dataset[8]. The Shifts Challenge consists of three tracks organized around each of the consituent datasets within Shifts. The dataset, baseline models and code to reproduce it all is provided in a GitHub repository[9]. Other than uncertainty and robustness research the dataset could be used for developing better models for each of the separate tasks - tabular data, translation and vehicle motion prediction.

Distribution The parts of the dataset which were produced by Yandex are distributed under an open-source CC BY NC SA 4.0 license. All the code is available under an open-source Apache 2.0 licence. It is our intention that the dataset be freely available for research purposes. The dataset is available as a tarball download from GitHub. Currently, as the Shifts Challenge is still underway, only the training and development sets are available. However, the full dataset, with full accompanying metadata, will be available once the challenge concludes on November 1st, 2021. Licence details for each constituent dataset in Shifts are described in appendices C-E.

Maintenance The dataset is being actively maintained by Yandex Research, with support from the weather, translation and self-driving teams, and the teams can be contacted by raising an issue on GitHub and by writing to the first author of this paper. The dataset is currently hosted on Yandex S3 storage and will be hosted there permanently for the foreseeable future. The dataset can be updated at the discretion of the dataset creators, though regular updates are not planned. Updates which expand the evaluation sets or add new ones will mean that the previous dev/eval sets are supported. Updates which fix errors in dev/eval sets mean that the prior ones are obsolete and unsupported. If any update is to occur, we will make an announcement via GitHub, twitter, and the Shifts challenge mailing list. Currently, as the data comes directly from Yandex, we do not allows other parties to update the Shifts Dataset. However, any issues found can be logged by raising an issue on GitHub or contacting the first author of this paper so that we can address them. Furthermore, as we are releasing the data under an open-source CC BY NC SA 4.0 license which allows modifications, we are happy for people to create derivative datasets using ours, provided the modifications are documents and the original dataset references.

Societal Consequences and Guidelines for Ethical Use Research on uncertainty estimation and robustness aims to make AI safer and more reliable, and therefore has limited negative societal consequences overall. Users of this dataset are encouraged to use it for the purpose of improving the reliability and safety of large-scale applications of machine learning. Furthermore, we encourage users of out dataset to develop compute and memory efficient methods for improving safety and reliability.

Responsibility The authors confirm that, to the best of our knowledge, the released dataset does not violate any prior licenses or rights. However, if such a violation were to exist, we are responsible for resolving this issue.

[8] research.yandex.com/shifts

[9] https://github.com/yandex-research/shifts

C Tabular Weather

The current appendix contains a description of the composition, collection, pre-processing and partitioning of the Shifts Tabular Weather Prediction dataset. Additionally, it contains a description of the metrics used for assessment and an expanded set of experimental results.

C.1 Dataset Description

Composition The data consists of pairs of meteorological features and target values at a particular latitude/longitude and time. The target value is air temperature measurements at 2 metres above the ground for regression and precipitation and cloudiness class from weather station measurements for classification. The feature vectors include both weather-related features such as sun evaluation at the current location, climate values of temperature, pressure and topography, and meteorological parameters on different pressure and surface levels from *weather forecast model predictions*. *Weather forecast model predictions* are values produced by the following weather forecast models: Global Forecast System (GFS),[10] Global Deterministic Forecast System from the Canadian Meteorological Center (CMC),[11] and the Weather Research and Forecasting (WRF) Model.[12] Each model returns the following predicted values: wind, humidity, pressure, clouds, precipitation, dew point, snow depth, air and soil temperature characteristics. Where applicable, the predictions are given at different isobaric levels from 50 hPa ($\approx$ 20 km above ground) to the ground level. The GFS and WRF models run 4 times a day (0, 6, 12 and 18 GMT), and the CMC model runs twice a day (0 and 12 GMT). Model spatial grid resolution is $0.25^\circ \times 0.25^\circ$ for GFS and $0.24^\circ \times 0.24^\circ$ for CMC. The WRF model is calculated for over 60 domains all over the globe, spatial resolution for each domain is 6×6 km. Altogether, there are 123 features in total. It is important to note that the features are highly heterogeneous, i.e., they are of different types and scales. The target air temperature values at different locations are taken from about 8K weather stations located across the globe, each of which periodically ($\approx$ each 3 hours) reports a set of measurements. In total, the dataset has 129 columns: 123 features, 4 meta-data attributes including time, latitude, longitude, and 2 targets - temperature (target for regression task), precipitation class (target for classification task) and climate type. The full feature list is provided in Section C.2.

Collection Process The data for features from GFS and CMC weather forecast models was downloaded in the GRIB file format from the web resources `https://www.ncdc.noaa.gov` and `https://weather.gc.ca/`, respectively. The GRIB files were decoded and collected by the production system of the Yandex Weather forecast service. MD5 hashes for files were checked after downloading the data. The parameters from WRF model were obtained from WRF model v3.6.1 computation on Yandex Weather servers. The data was checked for mistakes and outliers. Some parameters were converted to different units (for example degrees from K to C). We selected a subset of 123 weather parameters from the full dataset based on expertise and research of feature importance for weather forecasts of temperature and precipitation for the Yandex Weather production system. The data for weather station observations was downloaded from `https://www.ncdc.noaa.gov` and was decoded from SYNOP code. We filtered missed values and outliers by comparing with previous observations on the same weather station, and by comparing observation with nearby weather stations. Scripts and program codes for data collection and processing were prepared by in-house Yandex Weather software engineers. The period of data collection is from September 2018 to September 2019.

Preprocessing, Cleaning and Labelling The data was logged during applying trained CatBoost models for weather forecast prediction of the Yandex Weather service and was validated on Yandex Weather users by providing actual weather forecasts and accessing its mistakes on users and station measurements. We labeled data to match the timestamp of features and targets from these logs. Also we selected features only for latitudes and longitudes of weather observation stations to match with the measurements. Targets for air temperature were converted to degrees Celsius. Targets for precipitation class were constructed from cloudiness and precipitation measurements to create 9

[10] `https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs`

[11] `https://weather.gc.ca/grib/grib2_glb_25km_e.html`

[12] `https://www.mmm.ucar.edu/weather-research-and-forecasting-model`

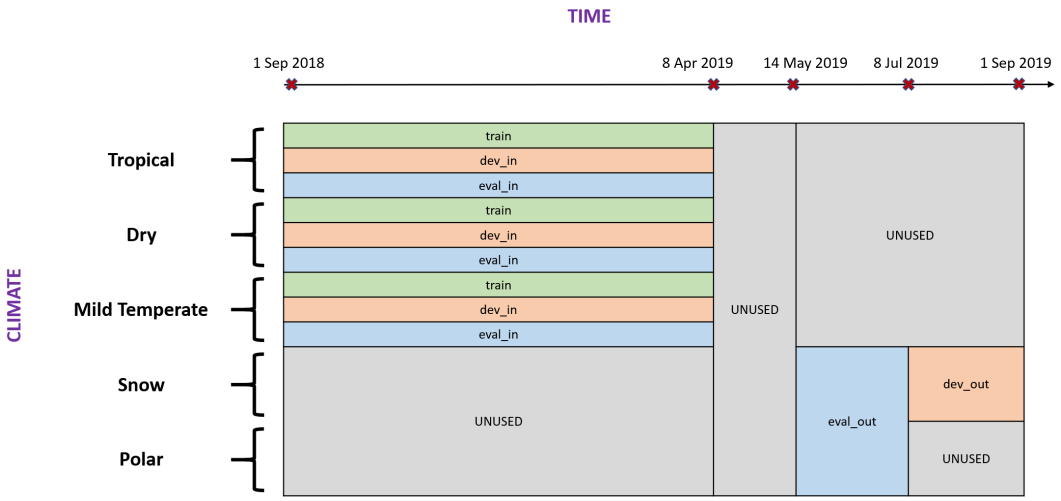


Figure 6: Canonical Partitioning of Weather Prediction dataset.

classes and labeled as follows: 0 — no precipitation, no clouds, 1 — no precipitation, partly cloudy, 2 — rain, partly cloudy, 3 — sleet, partly cloudy, 4 — snow, partly cloudy, 5 — no precipitation, cloudy, 6 — rain, cloudy, 7 — sleet, cloudy, 8 — snow, cloudy. The "raw" data was not saved, because it requires large amount of disk space. It was deleted after processing the data.

Partitioning into train, development, and evaluation sets To analyze the robustness of learned models to *climate shifts*, we use the Koppen climate classification [77] that provides publicly available data[13] that maps latitudes and longitudes at a 0.5° resolution to one of five main climate types: *Tropical*, *Dry*, *Mild Temperate*, *Snow* and *Polar*. This information is available over the years 1901 to 2010. The Weather Prediction dataset is augmented such that each sample has an associated climate type. The climate type is determined by minimizing the 1-norm between the longitudes/latitudes in the weather data and the Koppen climate classification for the most recent year available, 2010. The climate type is not used as a training feature.

There are 10M records in the full dataset distributed uniformly between September 1^{st}, 2018, and September 1^{st}, 2019, with samples across all five climate types. To test the robustness of the models, we evaluate how well they perform on time-shifted and climate-shifted data. Model performance is expected to decrease with time and climate shifts. However, a robust model is expected to be stable with these shifts.

In order to provide a standard benchmark which contains data which is both matched and shifted relative to the training set, we split the full dataset into 'canonically partitioned'[14] training, development, and evaluation datasets as follows (see Figure 6):

- The training data consists of measurements made from September 2018 till April 8^{th}, 2019 for climate types *Tropical*, *Dry*, and *Mild Temperate*. The training data includes two dummy rows in order to ensure there is at least one example of each of the precipitation classes (the targets for the classification task). The values for each of the features of the dummy examples are computed by averaging across the whole training dataset.
- The development data is composed of in-domain (`dev_in`) and out-of-domain (`dev_out`) data. The in-domain data corresponds to the same time range and climate types as the training data. The out-of-domain development data consists of measurements made from 8^{th} July till 1^{st} September 2019 for the climate type *Snow*. 50K data points are subsampled for the climate type *Snow* within this time range to construct `dev_out`.

[13] Available to download from `http://hanschen.org/koppen`

[14] Alternative partitioning can be made from the full data, but we will use the canonical partition throughout this work.

Table 8: Number of samples in the canonical partitioning of Weather Prediction dataset.

	Data	# of samples					
		Total	Tropical	Dry	Mild Temperate	Snow	Polar
Training	`train`	**3,129,592**	416,310	690,284	2,022,998	0	0
Development	`dev_in`	**50,000**	6,641	10,961	32,398	0	0
	`dev_out`	**50,000**	0	0	0	50,000	0
	`dev`	**100,000**	6,641	10,961	32,398	50,000	0
Evaluation	`eval_in`	**561,105**	74,406	123,487	363,212	0	0
	`eval_out`	**576,626**	0	0	0	525,967	50,659
	`eval`	**1,137,731**	74,406	123,487	363,212	525,967	50,659

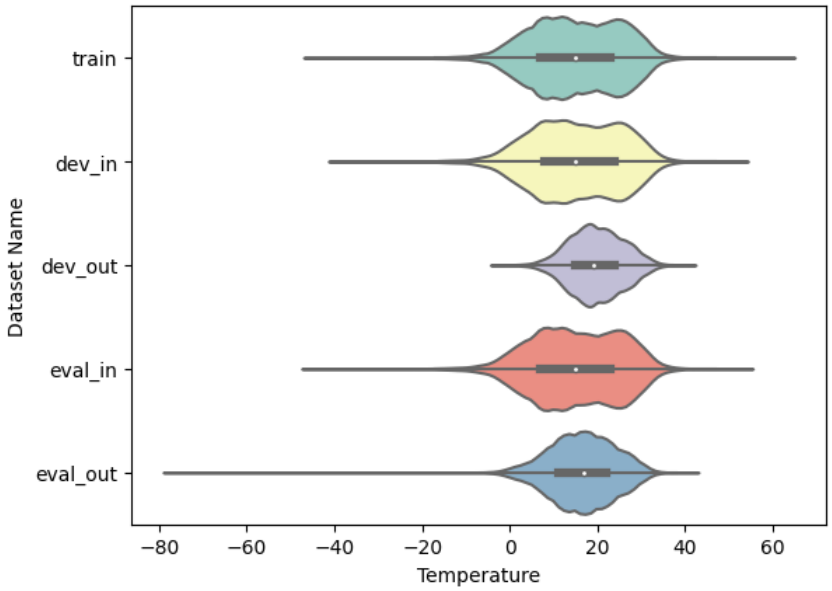
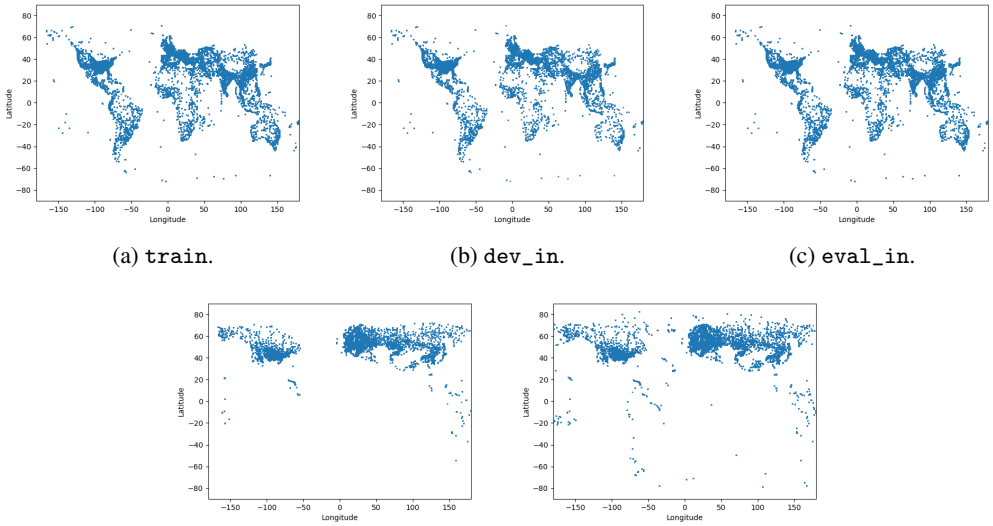


Figure 7: Temperature distributions on canonical partitions of Weather Prediction dataset.

- The evaluation data is also composed of in-domain (`eval_in`) and out-of-domain (`eval_out`) data. As before, the in-domain data corresponds to the same time range and climate types as the training data. The out-of-domain evaluation data is further shifted than the out-of-domain development data; measurements are taken from 14^{th} May till 8^{th} July 2019, which is more distant in terms of the time of the year from the in-domain data compared to the out-of-domain development data. The climate types are restricted to *Snow* and *Polar*.

Table 8 details the number of samples in the selected partition of the data. It also details the number of samples for each climate type for each part of the dataset. The in-domain data is split in approximately 83.7-1.3-15% ratio between training, development, and evaluation. Figure 7 depicts the shift in the target temperatures between the training, development, and evaluation datasets. It is clear that the temperature distribution is different for `dev_out` and `eval_out` compared to the in-domain sets. The higher average temperature in the out of domain sets is perhaps due to the out of domain data being sourced from the Summer regions (for the northern hemisphere) while the in-domain data is largely sourced from the Winter time period. Figure 8 further shows the shift in the samples' locations (latitudes/longitudes) between training, development, and evaluation datasets. The location shift is a natural result of the climate shifts present in the datasets where the training data tends to correspond to warmer parts of the world, whereas the development and evaluation datasets include colder climates too.

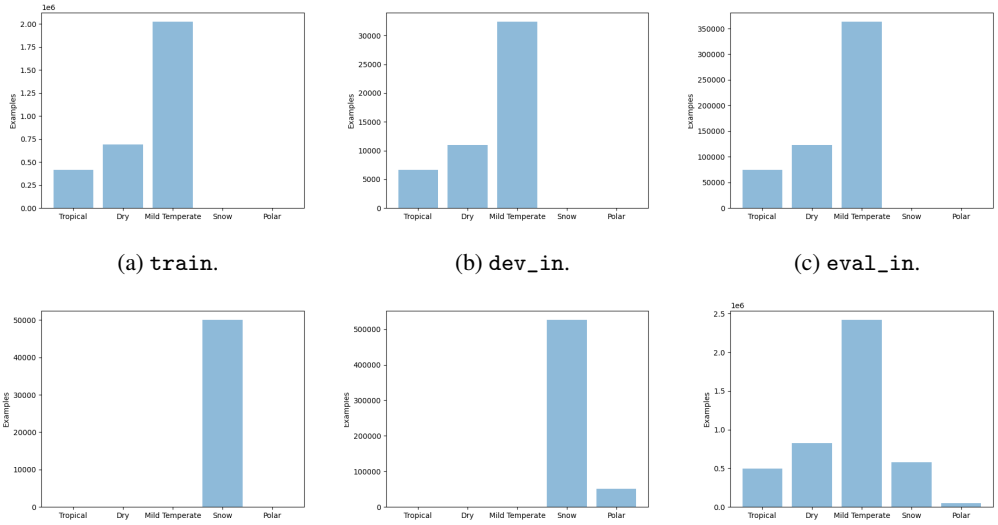
Format This dataset is provided in CSV format.



(a) `train`. (b) `dev_in`. (c) `eval_in`.

(d) `dev_out`. (e) `eval_out`.

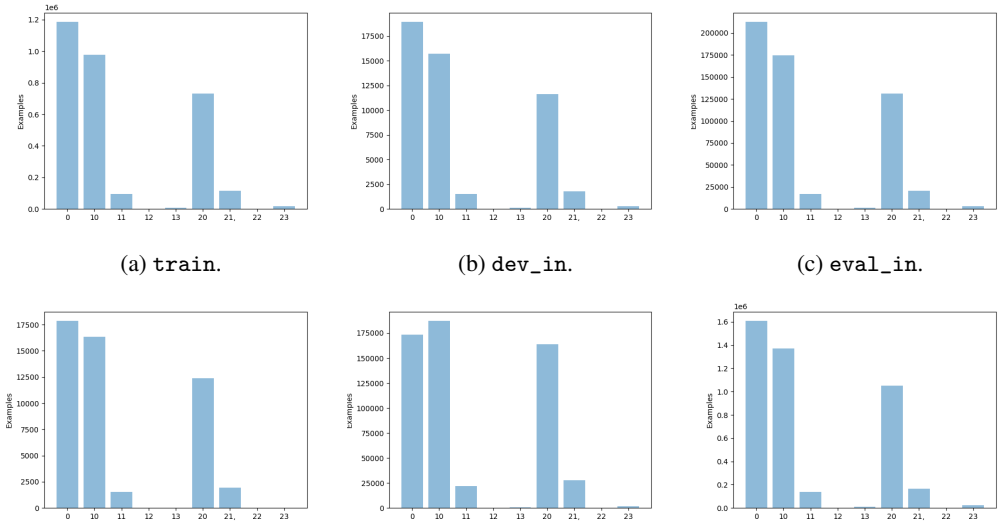
Figure 8: Location of samples from canonical partitioning of Weather Prediction dataset.



(a) `train`. (b) `dev_in`. (c) `eval_in`.

(d) `dev_out`. (e) `eval_out`. (f) `all`.

Figure 9: Distribution of climate types from canonical partitioning of Weather Prediction dataset.



(a) `train`. (b) `dev_in`. (c) `eval_in`.

(d) `dev_out`. (e) `eval_out`. (f) `all`.

Figure 10: Distribution of precipitation classes from canonical partitioning of Weather Prediction dataset.

Licence This dataset is provided under the CC BY NC SA 4.0 license.

C.2 Detailed description of features and targets

Meta-Data Features

1. fact_time — timestamp
2. fact_latitude — geographical latitude, degrees
3. fact_longitude — geographical longitude, degrees
4. climate — major climate type

Targets

1. fact_temperature — air temperature 2m above the ground, C
2. fact_cwsm_class - precipitation class

Features

1. climate_pressure — climate pressure, mmHg
2. climate_temperature — climate temperature, C
3. cmc_0_0_0_1000 — temperature at 1000 hPa isobaric level, K
4. cmc_0_0_0_2 — temperature at 2m, K
5. cmc_0_0_0_2_grad — difference between temperatures on adjacent horizons at 2m, K
6. cmc_0_0_0_2_interpolated — temperature at 2m interpolated between horizons, K
7. cmc_0_0_0_2_next — temperature at 2m for next horizon, K
8. cmc_0_0_0_500 — temperature at 500 hPa isobaric level, K
9. cmc_0_0_0_700 — temperature at 700 hPa isobaric level, K
10. cmc_0_0_0_850 — temperature at 850 hPa isobaric level, K
11. cmc_0_0_0_925 — temperature at 925 hPa isobaric level, K

12. cmc_0_0_6_2 — dew point temp at 2m, K
13. cmc_0_0_7_1000 — dew point depression at 1000 hPa isobaric level, K
14. cmc_0_0_7_2 — dew point depression at 2m, K
15. cmc_0_0_7_500 — dew point depression at 500 hPa isobaric level, K
16. cmc_0_0_7_700 — dew point depression at 700 hPa isobaric level, K
17. cmc_0_0_7_850 — dew point depression at 850 hPa isobaric level, K
18. cmc_0_0_7_925 — dew point depression at 925 hPa isobaric level, K
19. cmc_0_1_0_0 — absolute humidity from 0 to 1
20. cmc_0_1_11_0 — snow depth, m
21. cmc_0_1_65_0 — rain accumulated from cmc gentime, mm
22. cmc_0_1_65_0_grad — rain accumulated from cmc gentime difference between adjacent horizons, mm
23. cmc_0_1_65_0_next — rain accumulated from cmc gentime for next horizon, mm
24. cmc_0_1_66_0 — snow accumulated from cmc gentime, mm
25. cmc_0_1_66_0_grad — snow accumulated from cmc gentime difference between adjacent horizons, mm
26. cmc_0_1_66_0_next — snow accumulated from cmc gentime for next horizon, mm
27. cmc_0_1_67_0 — ice rain accumulated from cmc gentime, mm
28. cmc_0_1_67_0_grad — ice rain accumulated from cmc gentime difference between adjacent horizons, mm
29. cmc_0_1_67_0_next — ice rain accumulated from cmc gentime for next horizon, mm
30. cmc_0_1_68_0 — iced graupel accumulated from cmc gentime, mm
31. cmc_0_1_68_0_grad — iced graupel accumulated from cmc gentime difference between adjacent horizons, mm
32. cmc_0_1_68_0_next — iced graupel accumulated from cmc gentime for next horizon, mm
33. cmc_0_1_7_0 — instant precipitation intensity, mm/h
34. cmc_0_2_2_10 — wind U component at 10m, m/s
35. cmc_0_2_2_1000 — wind U component at 1000 hPa isobaric level, m/s
36. cmc_0_2_2_500 — wind U component at 500 hPa isobaric level, m/s
37. cmc_0_2_2_700 — wind U component at 700 hPa isobaric level, m/s
38. cmc_0_2_2_850 — wind U component at 850 hPa isobaric level, m/s
39. cmc_0_2_2_925 — wind U component at 925 hPa isobaric level, m/s
40. cmc_0_2_3_10 — wind V component at 10m, m/s
41. cmc_0_2_3_1000 — wind V component at 1000 hPa isobaric level, m/s
42. cmc_0_2_3_500 — wind V component at 500 hPa isobaric level, m/s
43. cmc_0_2_3_700 — wind V component at 700 hPa isobaric level, m/s
44. cmc_0_2_3_850 — wind V component at 850 hPa isobaric level, m/s
45. cmc_0_2_3_925 — wind V component at 925 hPa isobaric level, m/s
46. cmc_0_3_0_0 — surface pressure, Pa
47. cmc_0_3_0_0_next — next horizon surface pressure, Pa
48. cmc_0_3_1_0 — sea level pressure, Pa
49. cmc_0_3_5_1000 — geopotential height at 1000 hPa isobaric level, gpm (geopotential meter)
50. cmc_0_3_5_500 — geopotential height at 500 hPa isobaric level, gpm
51. cmc_0_3_5_700 — geopotential height at 700 hPa isobaric level, gpm

52. cmc_0_3_5_850 — geopotential height at 850 hPa isobaric level, gpm
53. cmc_0_3_5_925 — geopotential height at 925 hPa isobaric level, gpm
54. cmc_0_6_1_0 — cloudiness, % from 0 to 100
55. cmc_available — is there any data from cmc
56. cmc_horizon_h — cmc horizon, h
57. cmc_precipitations — avg precipitations rate between adjacent horizons, mm/h
58. cmc_timedelta_s — difference between cmc and forecast time, s
59. gfs_2m_dewpoint — dew point temperature at 2m, C
60. gfs_2m_dewpoint_grad — dew point temperature at 2m difference between horizons, C
61. gfs_2m_dewpoint_next — dew point temperature on next horizon, C
62. gfs_a_vorticity — absolute vorticity at height 1000 hPa, s-1
63. gfs_available — is there any data from gfs
64. gfs_cloudness — sum of 3 level cloudiness, from 0 to 3
65. gfs_clouds_sea — Cloud mixing ratio at level 1000 hPa, kg/kg 0.0
66. gfs_horizon_h — gfs horizon, h
67. gfs_humidity — relative humidity at 2m, %
68. gfs_precipitable_water — total precipitable water, kg m^-2
69. gfs_precipitations — avg precipitations rate between adjacent horizons, mm/h
70. gfs_pressure — surface pressure, mmHg
71. gfs_r_velocity — vertical Velocity at 1000 hPa, Pa/s
72. gfs_soil_temperature — soil temperature at 0.0-0.1 m, C
73. gfs_soil_temperature_available — is there gfs soil temp data
74. gfs_temperature_10000 — temperature at vertical level at 100 hPa, C
75. gfs_temperature_15000 — temperature at vertical level at 150 hPa, C
76. gfs_temperature_20000 — temperature at vertical level at 200 hPa, C
77. gfs_temperature_25000 — temperature at vertical level at 250 hPa, C
78. gfs_temperature_30000 — temperature at vertical level at 300 hPa, C
79. gfs_temperature_35000 — temperature at vertical level at 350 hPa, C
80. gfs_temperature_40000 — temperature at vertical level at 400 hPa, C
81. gfs_temperature_45000 — temperature at vertical level at 450 hPa, C
82. gfs_temperature_5000 — temperature at vertical level at 50 hPa, C
83. gfs_temperature_50000 — temperature at vertical level at 500 hPa, C
84. gfs_temperature_55000 — temperature at vertical level at 550 hPa, C
85. gfs_temperature_60000 — temperature at vertical level at 600 hPa, C
86. gfs_temperature_65000 — temperature at vertical level at 650 hPa, C
87. gfs_temperature_7000 — temperature at vertical level at 70 hPa, C
88. gfs_temperature_70000 — temperature at vertical level at 700 hPa, C
89. gfs_temperature_75000 — temperature at vertical level at 750 hPa, C
90. gfs_temperature_80000 — temperature at vertical level at 800 hPa, C
91. gfs_temperature_85000 — temperature at vertical level at 850 hPa, C
92. gfs_temperature_90000 — temperature at vertical level at 900 hPa, C
93. gfs_temperature_92500 — temperature at vertical level at 925 hPa, C
94. gfs_temperature_95000 — temperature at vertical level at 950 hPa, C
95. gfs_temperature_97500 — temperature at vertical level at 975 hPa, C

96. gfs_temperature_sea — temperature at 2m, C
97. gfs_temperature_sea_grad — temperature difference adjacent horizons at 2m
98. gfs_temperature_sea_interpolated — gfs_temperature_sea_interpolated between horizons, C
99. gfs_temperature_sea_next — next horizon temperature at 2m, C
100. gfs_timedelta_s — difference between gfs and forecast time, s
101. gfs_total_clouds_cover_high — cloud coverage (between horizons, divisible by 6) at high level, %
102. gfs_total_clouds_cover_low — cloud coverage (between horizons, divisible by 6) at low level, %
103. gfs_total_clouds_cover_low_grad — difference between low level cloud coverage on adjacent horizons, %
104. gfs_total_clouds_cover_low_next — next horizon cloud coverage (between horizons, divisible by 6) at low level, %
105. gfs_total_clouds_cover_middle — cloud coverage (between horizons, divisible by 6) at middle level, %
106. gfs_u_wind — 10 meter U wind component, m/s
107. gfs_v_wind — 10 meter V wind component, m/s
108. gfs_wind_speed — wind velocity, sqrt(gfs_u_wind^2 + gfs_v_wind^2), m/s
109. sun_elevation — sun height proxy above horizon (without corrections for precision and diffraction)
110. topography_bathymetry — height above or below sea level, m
111. wrf_available — is there any data from wrf
112. wrf_graupel — avg graupel rate between two horizons, mm/h
113. wrf_hail — hail velocity on two horizons, mm/h
114. wrf_psfc — pressure, Pa
115. wrf_rain — avg rain rate between two horizons, mm/h
116. wrf_rh2 — relative humidity at 2m, from 0 to 1
117. wrf_snow — avg snow rate between two horizons, mm/h
118. wrf_t2 — temperature at 2m, K
119. wrf_t2_grad — difference between temperatures at 2m on adjacent horizons, K
120. wrf_t2_interpolated — wrf_t2_interpolated between horizons, K
121. wrf_t2_next — next horizon temperature at 2m, K
122. wrf_wind_u — wind U component, m/s
123. wrf_wind_v — wind V component, m/s

C.3 Metrics

We aim at comparing different models in terms of uncertainty estimation and robustness to distributional shifts. Several performance metrics are considered.

Predictive Performance For temperature prediction, predictive performance and robustness to distributional shifts are evaluated by measuring RMSE and MAE between predictions and targets: lower the RMSE/MAE score on the test sets, greater the robustness of the models to the distributional shift. For classification, we use accuracy and macro-averaged F1. More robust models are expected to have higher values of these metrics.

Joint assessment of Uncertainty and Robustness We jointly assess robustness and uncertainty estimation via error-retention and F1-retention curves, described in Section 2 and detailed in Appendix A. For regression, we use MSE as the error metric instead of RMSE as it is linear with respect to

the error for each datapoint. For the F1-retention curve an acceptable prediction is defined as one where MSE < 1.0. This corresponds to an error of 1 degree or less, which most people cannot feel. Typically people are sensitive to differences in surrounding temperature of over a degree. These two performance metrics are respectively denoted as R-AUC and F1-AUC. For classification, we use the error rate to compute R-AUC. For both classification and regression, a good uncertainty measure is expected to achieve low R-AUC and high F1-AUC. Additionally, the F1 score at a retention rate of 95% of the most certain samples is also quoted and is denoted as F1@95%, which is a single point summary jointly of the uncertainty and robustness. Finally, ROC-AUC is used as a summary statistic for evaluating uncertainty-based out-of-distribution data detection.

C.4 Training details

The regression models are optimized with the loss function `RMSEWithUncertainty` [52] that predicts mean and variance of the normal distribution by optimizing the negative log-likelihood. Each model is constructed with a depth of 8 and then is trained for 20,000 iterations at a learning rate of 0.3. The classification models are optimized with the loss function `MultiClass` that predicts a discrete probability distribution over all classes. Each model is constructed with a depth of 6 and then is trained for 10,000 iterations at a learning rate of 0.4. Hyperparameter tuning is performed on the `dev_in` data for both tasks. All models were trained within under 8 hours using a normal laptop.

C.5 Additional experiments

In addition to considering ensembles of GBDT models implemented in CatBoost, we additionally consider ensembles of neural models. Specifically, we consider the FT-Transformer model [78]. We use FT-Transformers as the basis for Monte-Carlo Dropout Ensembles (MCDP) [13] as well as Deep Ensembles [14]. Additionally, we consider combining ensembles of CatBoost models with a Deep Ensemble of FT-Transformer models. Predictive performance figures are presented in table 9. Here, we can see that ensembles of CatBoost models and Deep ensembles of FT-Transformer models have very similar performance, with the latter marginally outperforming the former. However, their combination yields the most competitive figures. These results are consistent for both the classification and regression tasks.

Table 9: Predictive performance for Weather prediction. Mean is quoted for the single models.

Dataset	Model	Regression						Classification					
		RMSE ↓			MAE ↓			Accuracy (%) ↑			Macro F1 (%) ↑		
		In	Shifted	Full	In	Shifted	Full	In	Shifted	Full	In	Shifted	Full
dev	CatBoost, Single	1.59	2.30	1.98	1.18	1.75	1.47	67.0	47.5	57.2	48.8	27.9	43.3
	CatBoost, Ensemble	1.51	2.12	1.84	1.11	1.61	1.36	68.5	50.3	59.4	51.0	29.0	45.6
	FT-Transformer, Single	1.61	2.13	1.89	1.18	1.61	1.39	67.2	49.4	58.3	46.7	33.9	42.3
	FT-Transformer, MCDP	1.59	2.09	1.84	1.16	1.58	1.37	67.2	50.0	58.6	46.6	34.4	42.3
	FT-Transformer, Ensemble	1.50	**2.01**	1.77	1.10	**1.52**	1.31	68.8	**51.5**	60.2	49.2	**35.5**	44.7
	CatBoost ⊕ FT-Transformer	**1.47**	**2.01**	**1.76**	**1.08**	1.53	**1.30**	**69.3**	**51.5**	**60.4**	**51.4**	34.5	**46.4**
eval	CatBoost, Single	1.60	2.60	2.16	1.19	1.91	1.56	66.7	44.5	55.5	47.5	27.4	39.3
	CatBoost, Ensemble	1.52	2.37	2.00	1.11	1.75	1.44	68.2	46.7	57.3	49.8	28.8	41.4
	FT-Transformer, Single	1.62	2.40	2.05	1.18	1.77	1.48	67.0	45.9	56.3	44.0	29.1	37.4
	FT-Transformer, MCDP	1.59	2.34	2.01	1.17	1.73	1.45	67.0	46.4	56.6	44.0	29.4	37.7
	FT-Transformer, Ensemble	1.51	**2.24**	1.92	1.10	**1.66**	**1.38**	68.6	**48.0**	58.1	46.3	30.5	39.6
	CatBoost ⊕ FT-Transformer	**1.48**	2.25	**1.91**	**1.08**	**1.66**	**1.38**	**69.0**	**48.0**	**58.4**	**50.1**	**30.4**	**42.1**

We jointly assess robustness and uncertainty quality for the additional baselines in the table below. Again, the result show that combining all models yields the best results. Curiously, the results also show that Monte-Carlo dropout ensembles are now competitive for CatBoost ensembles. This suggests that the uncertainty quality of MCDP is better than for CatBoost ensembles, even if CatBoost has the better raw predictive quality.

Finally, we examine the quality of different uncertainty measures which are derivable from all of the baseline models. The results are provided in Table 11. The results show an interesting trend, where the model which has the best joint uncertainty and robustness performance is a combination of CatBoost and FT-Transformer ensembles, and the best measure of uncertainty is total variance and confidence for regression and classification, respectively. Both are measures of *total uncertainty*.

Table 10: Retention performance for Weather prediction. Mean is quoted for the single models.

Dataset	Model	Regression			Classification		
		R-AUC ↓	F1-AUC (%) ↑	F1@95% ↑	R-AUC ↓	F1-AUC (%) ↑	F1@95% ↑
dev	CatBoost, Single	1.894	44.35	62.72	0.1666	57.72	73.04
	CatBoost, Ensemble	1.227	52.20	65.83	0.1522	59.07	74.86
	FT-Transformer, Single	1.245	51.69	65.08	0.1592	58.51	73.80
	FT-Transformer, MCDP	1.197	52.08	65.62	0.1565	58.80	74.16
	FT-Transformer, Ensemble	1.051	53.66	**67.56**	0.1472	59.54	75.38
	CatBoost ⊕ FT-Transformer	**1.035**	**54.04**	67.47	**0.1453**	**59.71**	**75.58**
eval	CatBoost, Single	2.320	43.41	61.89	0.1799	56.25	71.56
	CatBoost, Ensemble	1.335	52.36	64.72	0.1640	58.22	73.17
	FT-Transformer, Single	1.386	51.86	63.96	0.1705	57.72	72.17
	FT-Transformer, MCDP	1.321	52.29	64.57	0.1676	58.04	72.55
	FT-Transformer, Ensemble	1.168	53.77	**66.40**	0.1576	58.95	73.84
	CatBoost ⊕ FT-Transformer	**1.151**	**54.09**	66.28	**0.1561**	**59.07**	**74.02**

At the same time, the best model for anomaly detection is a catboost ensemble using measures of *knowledge uncertainty*. This highlights how the best model and uncertainty measure to use greatly depends on the task.

Table 11: Comparing ensembled F1-AUC and ROC-AUC for various uncertainty measures on the tests sets from the canonical partitioning of Weather Prediction dataset for regression and classification.

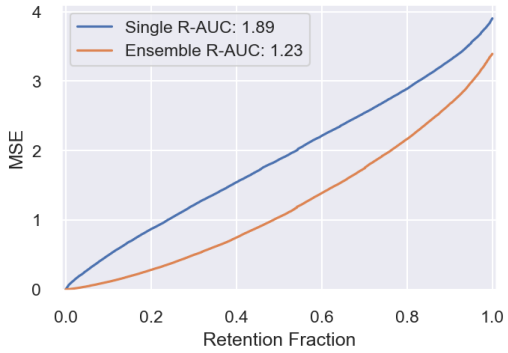
Data	Metric	Model	Regression			Classification				
			Total Unc.	Knowledge Unc.		Total Unc.		Knowledge Unc.		
			tvar	varm	EPKL	Conf	Entropy	MI	EPKL	RMI
dev	F1-AUC (%) ↑	CatBoost	52.20	50.12	50.51	59.07	58.86	57.72	57.69	57.66
		FT-Transformer	53.66	51.86	53.53	59.54	59.13	56.36	56.28	56.20
		CatBoost ⊕ FT-Transformer	**54.04**	52.22	51.49	**59.71**	59.26	57.65	57.49	57.36
	ROC-AUC (%) ↑	CatBoost	62.96	82.31	**85.29**	63.98	65.00	83.75	83.96	**84.12**
		FT-Transformer	58.10	65.89	61.63	35.46	65.48	71.89	71.85	71.79
		CatBoost ⊕ FT-Transformer	62.73	76.63	83.29	34.63	66.10	80.46	80.10	79.78
eval	F1-AUC (%) ↑	CatBoost	52.36	49.81	50.40	58.22	57.89	56.99	56.96	56.93
		FT-Transformer	53.77	51.83	53.58	58.95	58.55	55.68	55.59	55.51
		CatBoost ⊕ FT-Transformer	**54.09**	52.12	51.44	**59.07**	58.62	56.92	56.75	56.59
	ROC-AUC (%) ↑	CatBoost	65.99	78.32	**79.90**	66.20	66.76	83.44	83.59	**83.68**
		FT-Transformer	65.03	68.78	67.67	30.68	70.37	76.46	76.43	76.36
		CatBoost ⊕ FT-Transformer	67.78	75.43	79.29	30.86	69.92	82.49	82.16	81.85

C.5.1 Further experiments

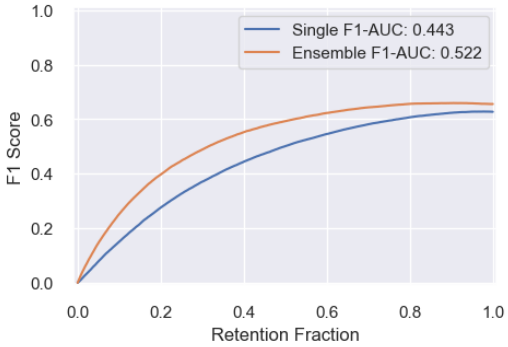
Figure 13 depicts additional splits beyond the canonical partition of the tabular weather data. Table 12 summarises the experiments to be performed with a brief description of what each experiment involves. All experiments are to be performed using CatBoost for both the regression and classification tasks. These experiments aim to better understand whether time or climate shift in the data leads to a greater performance drop from in-domain to shifted datasets. Hence, the focus here is on robustness only. The corresponding results for each experiment are given in Table 13.

Table 12: Description of additional experiments.

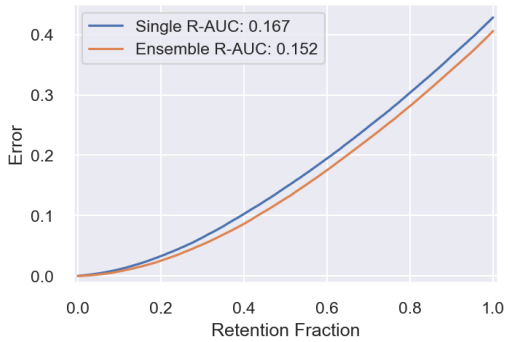
Exp	Training set	Development set	Description
A	`train`	`dev_in`	Time & climate shifts
B	`train` ⊕ `train_xclim`	`dev_in` ⊕ `dev_xclim`	Time shift
C	`train` ⊕ `train_xtime`	`dev_in` ⊕ `dev_xtime`	Climate shift
D	`train` ⊕ `train_xclim` ⊕ `train_xtime`	`dev_in` ⊕ `dev_xclim` ⊕ `dev_xtime`	No shift



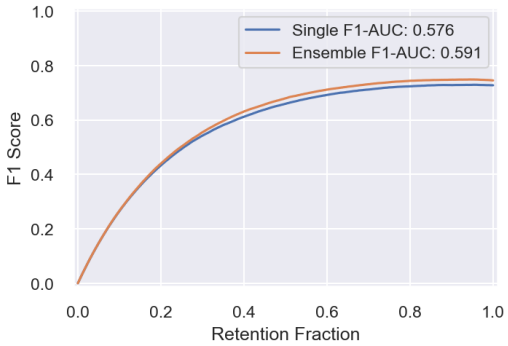
(a) CatBoost, Regression, MSE.



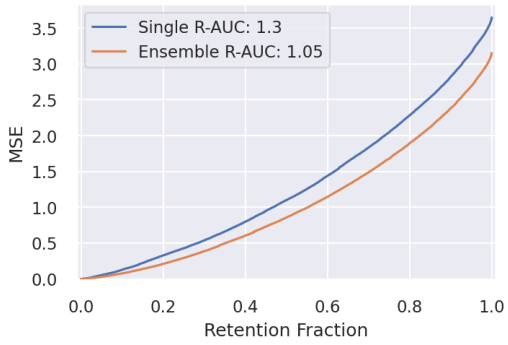
(b) CatBoost, Regression, F1.



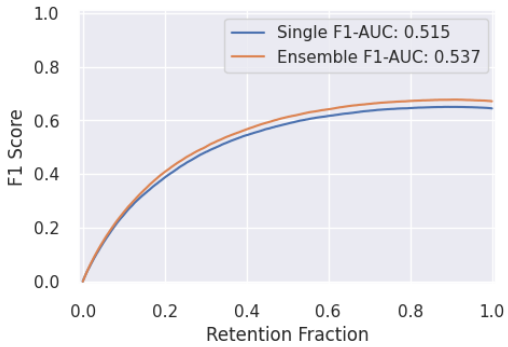
(c) CatBoost, Classification, error rate.



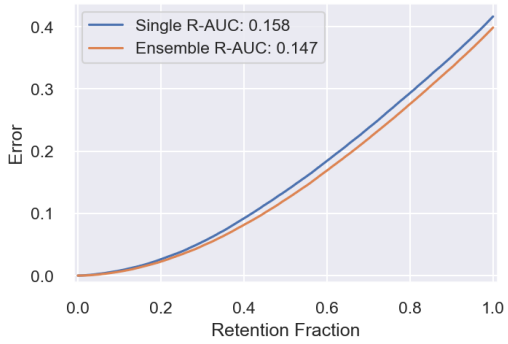
(d) CatBoost, Classification, F1.



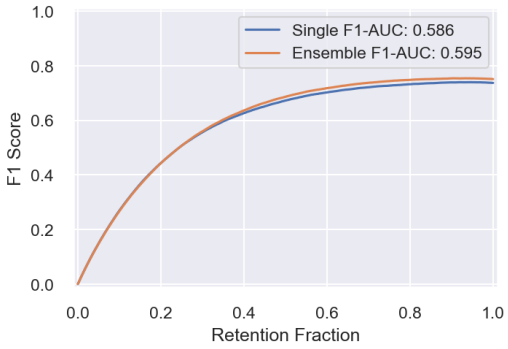
(e) FT-Trans, Regression, MSE.



(f) FT-Trans, Regression, F1.

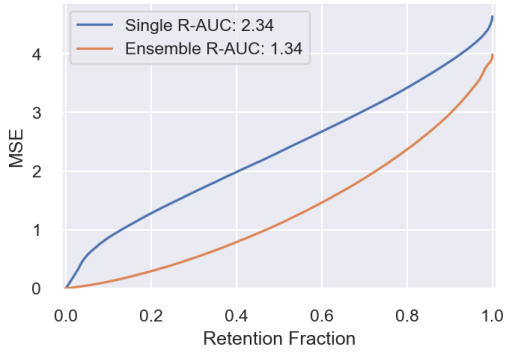


(g) FT-Trans, Classification, error rate.

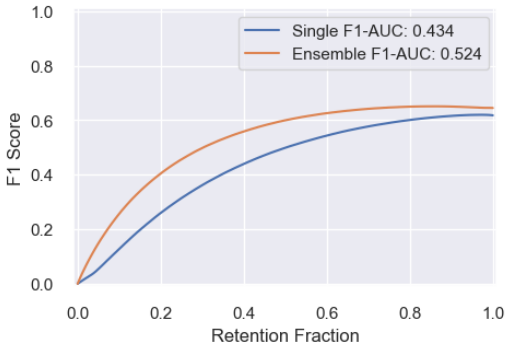


(h) FT-Trans, Classification, F1.

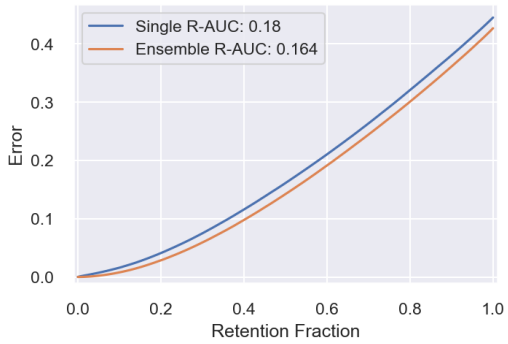
Figure 11: Retention curves for CatBoost and FT-Transformer on `dev` for the canonical Weather prediction dataset.



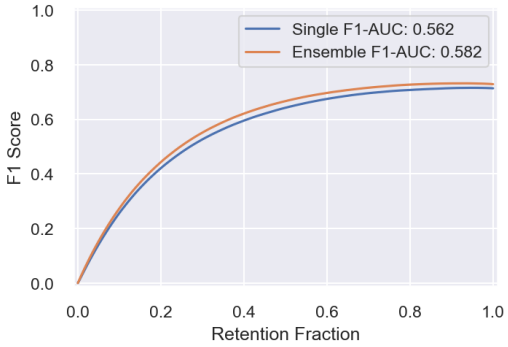
(a) CatBoost, Regression, MSE.



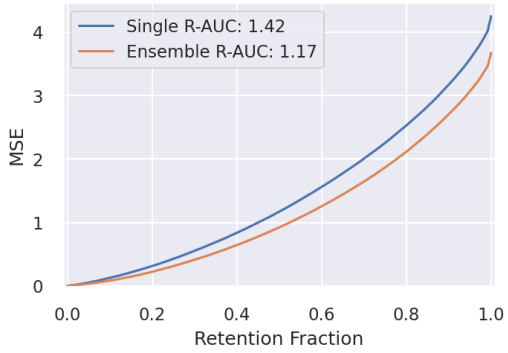
(b) CatBoost, Regression, F1.



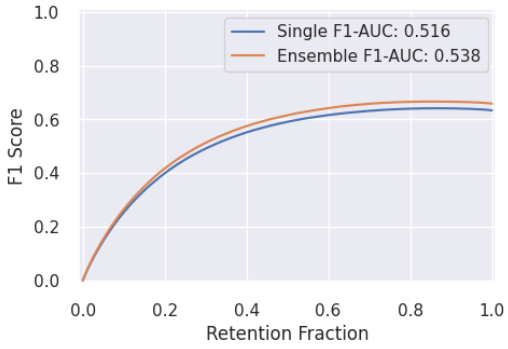
(c) CatBoost, Classification, error rate.



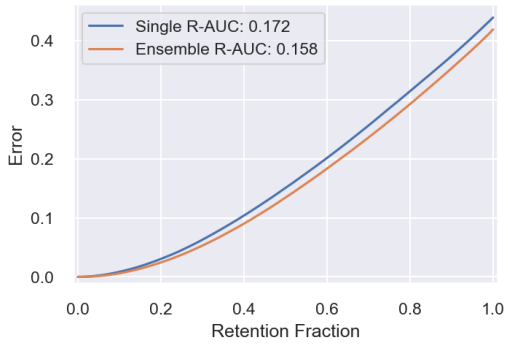
(d) CatBoost, Classification, F1.



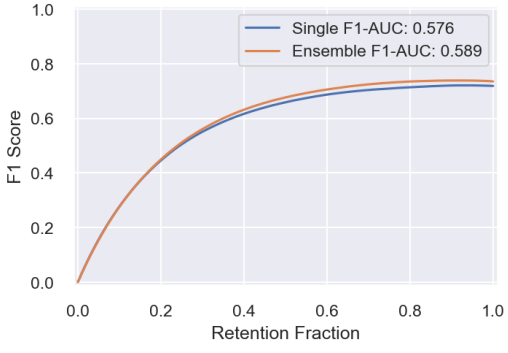
(e) FT-Trans, Regression, MSE.



(f) FT-Trans, Regression, F1.



(g) FT-Trans, Classification, error rate.



(h) FT-Trans, Classification, F1.

Figure 12: Retention curves with CatBoost and FT-Transformer on `eval` for the canonical Weather prediction dataset.

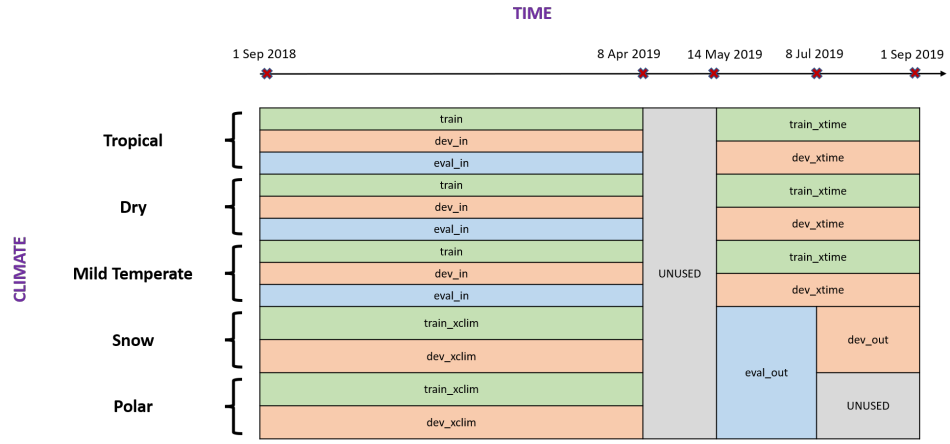


Figure 13: Extended splits of tabular weather data.

Table 13: Predictive performance for Weather prediction using different training sets. Mean is quoted for the single models.

Dataset	Model	Regression								Classification							
		RMSE ↓				MAE ↓				Accuracy (%) ↑				Macro F1 (%) ↑			
		A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
dev_in	CatBoost, Single	1.59	1.62	1.61	1.63	1.18	1.21	1.20	1.21	67.0	66.1	66.6	65.8	48.4	46.3	48.4	45.9
	CatBoost, Ensemble	1.51	1.52	1.51	1.54	1.11	1.12	1.11	1.14	68.5	67.2	67.7	66.8	51.0	48.5	51.0	48.4
dev_out	CatBoost, Single	2.30	2.30	2.04	1.95	1.75	1.75	1.54	1.48	47.5	50.9	51.8	54.0	27.9	31.7	32.0	34.1
	CatBoost, Ensemble	2.12	2.05	1.93	1.85	1.61	1.55	1.45	1.40	50.3	53.4	53.0	55.4	29.0	32.4	33.5	35.2
eval_in	CatBoost, Single	1.60	1.63	1.62	1.64	1.19	1.21	1.20	1.22	66.7	65.9	66.3	65.7	47.5	45.6	47.5	45.5
	CatBoost, Ensemble	1.52	1.53	1.52	1.55	1.11	1.12	1.12	1.14	68.2	67.1	67.6	66.7	49.8	48.1	49.3	48.0
eval_out	CatBoost, Single	2.60	2.62	2.28	2.15	1.91	1.93	1.69	1.62	44.5	48.3	48.6	51.5	27.4	30.1	29.2	31.9
	CatBoost, Ensemble	2.37	2.26	2.16	2.04	1.75	1.69	1.60	1.53	46.7	50.4	50.2	53.0	28.8	32.1	30.6	33.9

D Machine Translation

The current appendix contains a description of the composition, collection, pre-processing and partitioning of the Shifts Machine Translation dataset. Additionally, it contains a description of the metrics used for assessment and an expanded set of experimental results.

D.1 Dataset Description

Composition The Shifts Machine Translation datasets consists of a training, development (dev) and evaluation (eval) set. Each set consists of pairs of source and target sentences in English and Russian, respectively. As most production NMT systems are built using a variety of general purpose corpora, we do not provide a new training corpus, rather, we will use the freely available WMT'20 English-Russian corpus. This data covers a variety of domains, but primarily focuses on parliamentary and news data. For the most part, this data is grammatically and orthographically correct and language use is formal. This is representative of the type of data used, for example, to build the Yandex.Translate NMT system. The composition of the WMT'20 En-Ru corpus is detailed on the workshop for machine translation website here: `http://www.statmt.org/wmt20/translation-task.html`. For simplicity of access and archiving purposes we downloaded the WMT'20 En-Ru training data set and also made it available on the Shifts Dataset and Challenge GitHub here: `https://github.com/yandex-research/shifts`.

The dev and eval datasets consist of an "in-domain" partition matched to the training data, and an "out-of-distribution", or shifted partition, which contains examples of atypical language usage. We select the English-Russian Newstest'19 as the in-domain *development set* and will use a new corpus of news data collected from GlobalVoices News service [54] and manually annotated using expert human translators as the in-domain *evaluation set*. For the shifted development and evaluation data we use the Reddit corpus prepared for the WMT'19 robustness challenge [43]. This data contains

examples of slang, acronyms, lack of punctuation, poor orthography, concatenations, profanity, and poor grammar, among other forms of atypical language usage. This data is representative of the types of inputs that machine translation services find challenging. As Russian target annotations are not available, we pass the data through a two-stage process, where orthographic, grammatical and punctuation mistakes are corrected, and the source-side English sentences are translated into Russian by expert in-house Yandex translators. The development set is constructed from the same 1400-sentence test-set used for the WMT'19 robustness challenge. For the heldout evaluation set we use the open-source MTNT crawler which connects to the Reddit API to collect a further set of 3,000 English sentences from Reddit, which is similarly corrected and translated. Note that the Reddit data has comments made by users, but no personal identification data (login, name, etc...) or other user identification data was recorded or stored - the dataset only only contains the raw comments made on a public discussion platform. In terms of size, these development and evaluation sets are comparable or larger to the ones used in the WMT challenges and for evaluating productions systems.

Table 14: NMT Data Description - All Data is English-Russian

Data Set	N. Sentences	Avg. Sentence Length En	Avg. Sentence Length Ru	Type
WMT'20	62M	23,9	20.9	Train
NWT'19	1997	24.5	24.7	In-domain Dev
GlobalVoices	3,000	25.1	24.1	In-domain Eval
WMT'19 MTNT Reddit	1,362	17.2	16.5	Shifted Dev
Shifts Reddit	3,063	16.1	16.4	Shifted Eval

Both the development and evaluation Reddit data was manually annotated by members of the Yandex.Translate team with the following 7 non-exclusive anomaly flags:

- **Punctuation anomalies**: Some punctuation marks are missed or used incorrectly or some formatting (like Wiki markup) is used in the sentence.
- **Spelling anomalies**: The sentence contains spelling errors, including incorrect concatenation of two words as well as incorrect use of hyphens.
- **Capitalization anomalies**: Words that should be capitalized according to the language rules are written in lower case or vice versa.
- **Fluency anomalies**: The sentence is non-fluent due to wrong or missing prepositions, pronouns or ungrammatical form choice.
- **Slang anomalies**: In the sentence there are slang words of abbreviations like "idk" for "I don't know" or "cuz" for "because".
- **Emoji anomalies**: The sentence contains emojis either at the end of it, or instead of some words.
- **Tags anomalies**: The sentence contains markup for usernames or code like "r/username".

An analysis of the occurrence and co-occurrence of these anomalies is provided in figure 14.

Collection Process GlobalVoices[54] data was crawled for parallel news articles in English and Russian using internal Yandex tools. The raw articles were manually split into sentence-pairs by in-house Yandex assessors. A full set of 30000 sentence pairs was produced, from which a subset of 3000 sentences was uniformly randomly sampled. Reddit data was crawled using the open-source MTNT [43] crawler from `https://github.com/pmichel31415/mtnt`. This crawler links in with the Reddit API to allow mining and crawling Reddit for data. The crawler collected a set of 100K user comments which were then split into sentences using the NLTK toolkit. Then a set of 3500 sentences was randomly uniformly selected. After pre-processing and cleaning a set of 3065 sentences was produced.

Preprocessing and Cleaning For the GlobalVoices data parallel sentences markup was done manually by in-house Yandex assessors; non-parallel sentences were removed from dataset. For Reddit data 1-word phrases and sentences consisting only of non-alphabetical symbols were removed.

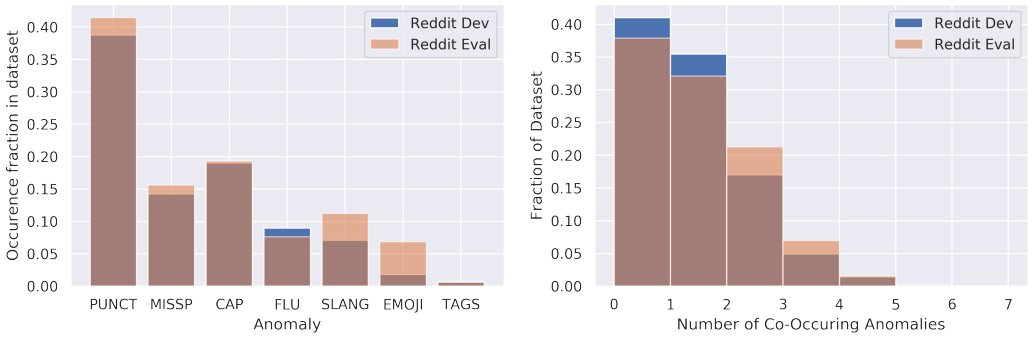


Figure 14: Analysis of anomaly occurrence and co-occurrence in Reddit (shifted) development and evaluation data

Professional editors were used to manually correct grammatical and orthographic mistakes prior to translating into Russian, but were explicitly told to maintain the non-formal style as much as possible. This error correction was used only for obtaining target-side Russian translation.

Guidelines on ethical use Users are discouraged from attempting to discover to which Reddit users the comments belong by manually or automatically crawling through Reddit to find the comments.

Format This dataset is provided in raw text format and a TSV with metadata for the dev and eval reddit data.

License The Shifts Machine Translation dataset is released under a mixed licence. GlobalVoices evaluation data is released under CC BY NC SA 4.0 . The source-side text for the Reddit development and evaluation datasets exist under terms of the Reddit API. The target side Russian sentences were obtained by Yandex via in-house professional translators and are released under CC BY NC SA 4.0. We highlight that the development set source sentences are the same ones as used in the MTNT dataset.

D.2 Metrics

To evaluate the performance of our models we will consider the following two metrics : corpus-level BLEU [55] and sentence-level GLEU [56, 57, 58]. GLEU is an analogue of BLEU which is stable when computed at the level of individual sentences. Thus, it is far more useful at evaluating system performance on a per-sample basis, rather than at the level of an entire corpus. Note that GLEU correlates strongly with BLEU at the corpus level.

Machine translation is inherently a multi-modal task, as a sentence can be translated in multiple equally valid ways. Furthermore, translation systems often yield multiple translation hypothesis. To account for this we will consider two GLEU-based metrics for evaluating translation quality. First is the *expected GLEU* or **eGLEU** across all translation hypotheses returned by a translation models. Each hypothesis is assumed to be assigned a *confidence score*, and confidences across each hypotheses by sum to one. This is our primary assessment metric:

$$\text{eGLEU} = \frac{1}{N}\sum_{i=1}^{N}\sum_{h=1}^{H}\text{GLEU}_{i,h}\cdot w_{i,h}, \quad w_{i,h} > 0, \sum_{h=1}^{H} w_h = 1 \tag{4}$$

Additionally, we will consider the *maximum GLEU* or **maxGLEU** across all hypothesis, which represents an upper bound on performance, given a model can appropriately rank it's hypotheses:

$$\text{maxGLEU} = \frac{1}{N}\sum_{i=1}^{N}\max_h\left[\text{GLEU}_{i,h}\right] \tag{5}$$

Finally, in order to calculate area under the error retention curve we need to introduce an *error metric*, where lower error is better. This is trivially done by introducing *eGLEU error*, which defined as:

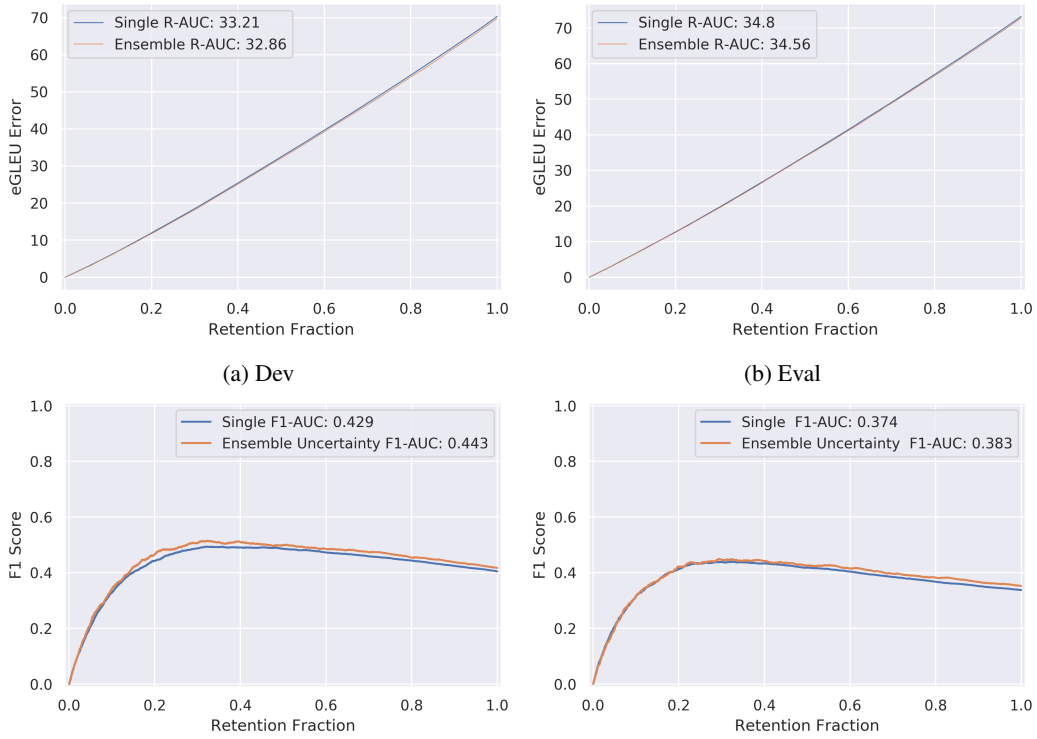
$$\text{eGLEU Error} = 100 - \text{eGLEU} \tag{6}$$

Thus, in section 4, area under the error retention curve (R-AUC), as well as the F1 metric for detecting 'valid predictions' will be calculated using eGLEU Error.

D.3 Training details

Training data was standard used the standard perl-based script provided in Fairseq [59] examples. Duplicate sentence pairs as well as sentence pairs where source and target text matched were removed. Models were trained using Fairseq version 0.8. A full description and for from preprocessing and training is provided here. All models were trained used 8xV100 GPUs over roughly 48 hours.

D.4 Additional Results



(a) Dev (b) Eval

(c) Dev (d) Eval

Figure 15: Location of samples from canonical partitioning of Weather Prediction dataset.

E Vehicle Motion Prediction

The current appendix contains a description of the composition, collection, pre-processing and partitioning of the Shifts Vehicle Motion Prediction dataset. Additionally, it contains a description of the metrics used for assessment and an expanded set of experimental results.

E.1 Dataset Description

Table 15: A comparison of various motion prediction datasets. The Shifts Vehicle Motion Prediction dataset is the largest by number of scenes and total size in hours.

Dataset	Scene Length (s)	# Scenes			Total Size (h)	Avg. # Actors
		Train	Dev	Eval		
Argoverse	5	205,942	39,472	78,143	320	50
Lyft	25	134,000	11,000	16,000	1,118	79
Waymo	20	72,347	15,503	15,503	574	-
Shifts	10	500,000	50,000	50,000	1,667	29

Composition The dataset for the Vehicle Motion Prediction task was collected by the Yandex Self-Driving Group (SDG) fleet. This is the largest vehicle motion prediction dataset released to date, containing 600,000 scenes (see Table 15 for a comparison to other public datasets). The dataset consists of scenes spanning six locations, three seasons, three times of day, and four weather conditions (cf. Table 16 and 17). Each of these conditions is available in the form of tags associated with every scene. Each scene is 10 seconds long and is divided into 5 seconds of context features and 5 seconds of ground truth targets for prediction, separated by the time $T = 0$. The goal of the task is to predict the movement trajectory of vehicles at time $T \in (0, 5]$ based on the information available for time $T \in [-5, 0]$.

Table 16: The number of scenes in the Vehicle Motion Prediction dataset by location and season.

Location	Train	Dev	Eval
Moscow	450,504	30,505	30,534
Skolkovo	6,283	2,218	2,956
Innopolis	15,086	5,164	5,016
Ann Arbor	19,349	8,290	6,617
Modiin	3,502	2,262	1,555
Tel Aviv	5,276	1,561	3,322

Season	Train	Dev	Eval
Summer	85,698	10,634	10,481
Autumn	126,845	15,290	15,840
Winter	287,457	24,076	23,679
Spring	0	0	0

Each scene includes information about the state of dynamic objects (i.e., vehicles, pedestrians) and an HD map. Each vehicle is described by its position, velocity, linear acceleration, and orientation (yaw, known up to $\pm\pi$). A pedestrian state consists of a position vector and a velocity vector. All state components are represented in a common coordinate frame and sampled at 5Hz frequency by the perception stack running on the Yandex SDG fleet. The HD map includes lane information (e.g., traffic direction, lane priority, speed limit, traffic light association), road boundaries, crosswalks, and traffic light states, which are also sampled at 5Hz. To facilitate easy use of this dataset, we provide utilities to render scene information as a feature map, which can be used as an input to a standard vision model (e.g., a ResNet [79]). Our utilities represent each scene as a birds-eye-view image with each channel corresponding to a particular feature (e.g., a vehicle occupancy map) at a particular timestep. We also provide pre-rendered feature maps for every prediction request (cf. Appendix E.2) in the dataset, which are used to train the baseline models. The maps are 128×128 pixels in size with each pixel covering 1 square meter, have 17 channels describing both HD map information and dynamic object states at time $T = 0$, and are centered with respect to the agent for which a prediction is being made. Researchers working with the dataset are free to use these feature maps, use the provided utilities to render another set of feature maps at different (earlier) timesteps, or construct their own scene representations from the raw data.

The ground truth part of a scene contains future states of dynamic objects sampled at 5Hz for a total of 25 state samples. Some objects might not have all 25 states available due to occlusions or imperfections of the on-board perception system.

Table 17: The number of scenes in the Vehicle Motion Prediction dataset by precipitation and time of day.

Precipitation Type	Train	Dev	Eval
No	432,598	44,799	44,274
Rain	15,618	1,857	1,751
Sleet	15,210	1,082	990
Snow	36,574	2,262	2,985

Sun Phase	Train	Dev	Eval
Astronomical Night	171,867	13,164	13,113
Daylight	299,065	33,879	33,979
Twilight	29,068	2,957	2,908

A number of vehicles in the scene are labeled as *prediction requests*. These are the vehicles that are visible at the most recent time $T = 0$ in the context features part of a scene, and therefore would call for a prediction in a deployed system. For such vehicles we provide not only their future trajectories, but also a number of non–mutually exclusive tags (detailed in Table 18) describing the associated maneuver in more detail – whether the vehicle is turning, accelerating, slowing down, etc. – for a total of 10 maneuver types. Note that some prediction requests may not have all 25 state samples available. We call prediction requests with fully-observed state *valid* prediction requests and propose to evaluate predictions only on those.

Table 18: Number of actor maneuvers of the respective type.

Maneuver Type	Train	Dev	Eval
Move Left	254,843	25,049	25,820
Move Right	322,231	30,074	30,633
Move Forward	5,032,724	395,467	413,920
Move Back	54,677	4,811	4,891
Acceleration	2,473,750	206,977	215,009
Deceleration	2,050,186	168,550	174,477
Uniform Movement	6,369,920	566,083	573,033
Stopping	441,619	38,411	39,336
Starting	739,143	64,986	65,759
Stationary	4,620,678	433,161	433,576

In order to study the effects of distributional shift, as well as assess the robustness and uncertainty estimation of baseline models, we divide the Vehicle Motion Prediction dataset such that there are *in-domain* partitions which match the location and precipitation type of the training set, and *out-of-domain* or *shifted* partitions which do not match the training data along one or more of those axes. Furthermore, we provide a *development* set which acts as a validation set, and an *evaluation* set which acts as the test set. For standardized benchmarking we define a *canonical partitioning* of the full dataset (cf. Figure 16, Table 19) as the following. The training, in-domain development, and in-domain evaluation data are taken from Moscow. Distributionally shifted development data is taken from Skolkovo, Modiin, and Innopolis. Distributionally shifted evaluation data is taken from Tel Aviv and Ann Arbor. In addition, we remove all cases of precipitation from the in-domain training, development, and evaluation sets, while distributionally shifted datasets include precipitation. The canonical partitioning is fully described in Figure 16. This partitioning is also the one used in the Shifts Challenge.

Collection Process The Vehicle Motion Prediction data was collected by the perception system running onboard a number of self-driving vehicles equipped with LiDAR sensors, radars, and cameras. This perception system consists of a number of neural network–based detectors followed by an object tracker that fuses detections across sensor modalities and time. The provided HD map for each location has been constructed and validated by cartographers employed by Yandex SDG. The provided dataset was sampled from a much larger dataset collected over a course of 8 months. The sampling procedure was biased towards sampling scenes on which the motion prediction system currently used

	Moscow	Skolkovo	Modiin	Innopolis	Ann-Arbor	Tel Aviv
No precipitation	train development in evaluation in	development out	development out	development out	evaluation out	evaluation out
Rain	UNUSED	development out	development out	development out	evaluation out	evaluation out
Sleet	UNUSED	UNUSED	UNUSED	UNUSED	evaluation out	evaluation out
Snow	UNUSED	development out	development out	development out	evaluation out	evaluation out

Figure 16: The canonical partitioning of the Vehicle Motion Prediction dataset.

Table 19: The number of scenes in the canonical dataset partitioning.

Dataset Partition	In-Distribution	Distributionally Shifted
Train	388,406	-
Development	27,036	9,569
Evaluation	26,865	9,939

by the SDC fleet makes mistakes, as well as sampling more scenes from locations where the fleet drives less frequently.

Preprocessing and Cleaning The collected dataset has been cleaned from scenes in which:

- any kind of onboard system failure was detected, as the perception system output can potentially be unreliable in such scenes;
- the perception system has produced outputs that clearly violate physical constraints, such as actors having unrealistic acceleration or colliding with one other.

Format This dataset is provided in protobuf format.

License We release this dataset under the CC BY NC SA 4.0 license.

E.2 Task Setup

Vehicle Motion Prediction is a complex task and therefore must be described in detail. We provide a training dataset $\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$ of time-profiled ground truth trajectories (i.e., plans) $\boldsymbol{y}$ paired with high-dimensional observations (features) $\boldsymbol{x}$ of the corresponding scenes. Each $\boldsymbol{y} = (s_1, \ldots, s_T)$ corresponds to the trajectory of a given vehicle observed through the SDG perception stack. Each state s_t corresponds to the x- and y-displacement of the vehicle at timestep t, s.t. $\boldsymbol{y} \in \mathbb{R}^{T\times 2}$. We consider the performance of models on development and evaluation datasets $\mathcal{D}^j_{\text{dev}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{M_j}$. and $\mathcal{D}^j_{\text{eval}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{M_j}$. See Figure 17 for a depiction of the task.

Prediction Requests. There are N (M_j) prediction requests in the training dataset (evaluation datasets), with many requests for each scene corresponding to the many different vehicle trajectories observed. For example, in the canonical partition of the data, there are 388,406 scenes in the training dataset (Moscow, no precipitation), and 5,649,675 valid prediction requests.

Models can be trained to make use of ground truth trajectories that contain occlusions (i.e., prediction requests that are not valid) during training, such as through linear interpolation of missing steps. However, for the baseline methods considered in this work, both training and evaluation are done using only the fully observed ground truth trajectories.

Next, we describe the two levels of uncertainty quantification that we consider for each prediction request in the proposed task: per-trajectory and per–prediction request uncertainty scores.

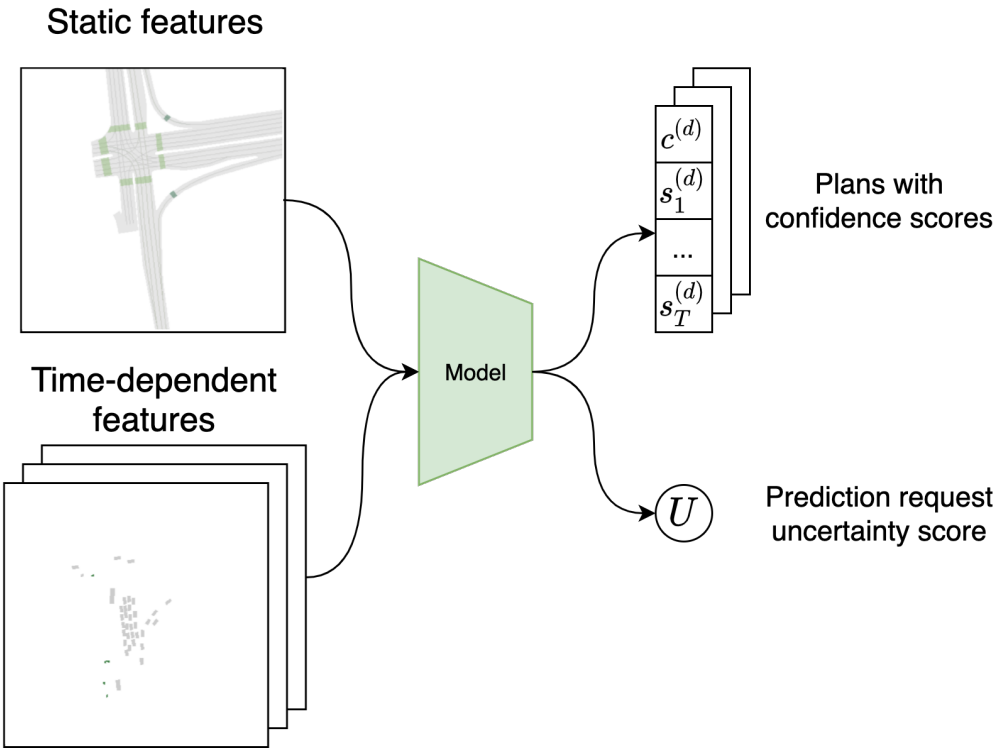


Figure 17: Diagram of the Vehicle Motion Prediction task. Models take as input a single scene context $\boldsymbol{x}$ composed of static (HD map) and time-dependent input features, and predict trajectories $\{\boldsymbol{y}^{(d)} \mid d \in 1, \ldots, D\}$ with corresponding per-trajectory confidence scores $\{\boldsymbol{c}^{(d)} \mid d \in 1, \ldots, D\}$, as well as a single per–prediction request uncertainty score U.

Per-Trajectory Confidence Scores. Like machine translation, motion prediction is an inherently multimodal task. A motion prediction model can produce a different number of sampled trajectories (plans) D_i for each input $\boldsymbol{x}_i$; in other words, for two inputs $\boldsymbol{x}_i, \boldsymbol{x}_j$ with $i \neq j$, D_i and D_j can differ. As a justification, consider that in a certain context, multiple trajectories may be desirable to capture multimodality (e.g., the vehicle of interest is at a T-junction), and in others a single or fewer trajectories would be sufficient (e.g., the vehicle is clearly proceeding straight). In our task, we expect a stochastic model to accompany its D_i predicted trajectories on a given input $\boldsymbol{x}_i$ with scalar per-trajectory confidence scores $c_i^{(d)}, d \in \{1, \ldots D_i\}$. These provide an ordering of the plausibility of the various trajectories predicted for a given input. The scores must be non-negative and sum to 1 (i.e., form a valid probability distribution).

Per–Prediction Request Uncertainty Score. We also expect models to produce scalar uncertainty estimates corresponding to each prediction request input $\boldsymbol{x}_i$. For example, on evaluation dataset $\mathcal{D}_{\text{eval}}^j$, we have M_j per–prediction request uncertainty scores $\{U_i \mid i \in 1, \ldots, M_j\}$. These correspond to the model's uncertainty in making any trajectory prediction for the agent of interest. In a real-world deployment setting, a self-driving vehicle would associate a high per–prediction request uncertainty score with a scene context that is particularly unfamiliar or high-risk.

Next, we will describe standard motion prediction performance metrics, followed by confidence-aware metrics which reward models with well-calibrated uncertainty.

E.3 Performance Metrics

Standard Performance Metrics. We assess the performance of a motion prediction system using several standard metrics.

The average displacement error (ADE) measures the quality of a predicted trajectory $\boldsymbol{y}$ with respect to the ground truth trajectory $\boldsymbol{y}^*$ as

$$\text{ADE}(\boldsymbol{y}) \coloneqq \frac{1}{T}\sum_{t=1}^{T} \|s_t - s_t^*\|_2 \,, \tag{7}$$

where $\boldsymbol{y} = (s_1, \ldots, s_T)$. Analogously, the final displacement error

$$\text{FDE}(\boldsymbol{y}) \coloneqq \|s_T - s_T^*\|_2 \,, \tag{8}$$

measures the quality at the last timestep.

Stochastic models define a predictive distribution $q(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\theta})$, and can therefore be evaluated over the D trajectories sampled for a given input $\boldsymbol{x}$. For example, we can measure an aggregated ADE over D samples with

$$\text{aggADE}_D(q) \coloneqq \bigoplus_{\{\boldsymbol{y}\}_{d=1}^{D} \sim q(\boldsymbol{y}|\boldsymbol{x})} \text{ADE}(\boldsymbol{y}^d), \tag{9}$$

where $\oplus$ is an aggregation operator, e.g., $\oplus = \min$ recovers the minimum ADE (minADE_D) commonly used in evaluation of stochastic motion prediction models [72, 66]. We consider minimum and mean aggregation of the average displacement error (minADE, avgADE), as well as of the final displacement error (minFDE, avgFDE).

Per-Trajectory Confidence-Aware Metrics. A stochastic model used in practice for motion prediction must ultimately *decide* on a particular predicted trajectory for a given prediction request. We may make this decision by selecting for evaluation the predicted trajectory with the highest per-trajectory confidence score. In other words, given per-trajectory confidence scores $\{c^{(d)} \mid d \in 1, \ldots, D\}$ we select the top trajectory $y^{(d^*)}, d^* = \arg\max_d c^{(d)}$, and measure the decision quality using *top1* ADE and FDE metrics, e.g.,

$$\text{top1ADE}_D(q) \coloneqq \text{ADE}(\boldsymbol{y}^{(d^*)}). \tag{10}$$

We may also wish to assess the quality of the relative weighting of the D trajectories with their corresponding per-trajectory confidence scores $c^{(d)}$. For this the following weighted metric can be considered:

$$\text{weightedADE}_D(q) \coloneqq \sum_{d \in D} c^{(d)} \cdot \text{ADE}(\boldsymbol{y}^{(d)}). \tag{11}$$

The top1FDE and weightedFDE metrics follow analogously to the above. Unfortunately, these metrics, while highly intuitive, have a conceptual limitation. Consider the following loss:

$$\mathcal{L}\left(\text{p}(\boldsymbol{y}|\boldsymbol{x}), \{\hat{c}_i^{(1:D)}, \hat{\boldsymbol{y}}^{(1:D)}\}\right) = \mathbb{E}_{\text{p}(\boldsymbol{y}|\boldsymbol{x})}\left[\sum_{d=1}^{D} c^d \text{ADE}(\hat{\boldsymbol{y}}^d, \boldsymbol{y})\right], \; \{\hat{c}_i^{(1:D)}, \hat{\boldsymbol{y}}^{(1:D)}\} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{\theta}) \tag{12}$$

which is the expected weightedADE given a set of trajectories and weights from a model. If we wish to minimize this loss with respect to the predicted trajectories and weights, then:

$$\begin{aligned}
\mathcal{L}_{\text{min}} &= \min_{\{\hat{c}_i^{(1:D)}, \hat{\boldsymbol{y}}^{(1:D)}\}} \left\{ \mathbb{E}_{\text{p}(\boldsymbol{y}|\boldsymbol{x})} \left[\sum_{d=1}^{D} c^d \text{ADE}(\hat{\boldsymbol{y}}^d, \boldsymbol{y}) \right] \right\} \\
&= \min_{\{\hat{c}_i^{(1:D)}\}} \left\{ \sum_{d=1}^{D} \hat{c}^d \left(\min_{\{\hat{y}^{(d)}\}} \left\{ \mathbb{E}_{\text{p}(\boldsymbol{y}|\boldsymbol{x})} \left[\text{ADE}(\hat{\boldsymbol{y}}^d, \boldsymbol{y}) \right] \right\} \right) \right\} \\
&= \min_{\{\hat{c}_i^{(1:D)}\}} \left\{ \mathbb{E}_{\text{p}(\boldsymbol{y}|\boldsymbol{x})} \left[\text{ADE}(\hat{\boldsymbol{y}}^*, \boldsymbol{y}) \right] \sum_{d=1}^{D} \hat{c}^d \right\} \\
&= \mathbb{E}_{\text{p}(\boldsymbol{y}|\boldsymbol{x})} \left[\text{ADE}(\hat{\boldsymbol{y}}^*, \boldsymbol{y}) \right]
\end{aligned} \tag{13}$$

where $\hat{\boldsymbol{y}}^*$ is the *weighted geometric median*

$$\hat{\boldsymbol{y}}^* = \arg_{\{\hat{\boldsymbol{y}}\}} \min \left\{ \mathbb{E}_{\text{p}(\boldsymbol{y}|\boldsymbol{x})} \left[\text{ADE}(\hat{\boldsymbol{y}}, \boldsymbol{y}) \right] \right\} \tag{14}$$

Thus, the optimal model would suffer from *mode-collapse* and always yields the weighted geometric median of the modes of the true distribution of trajectories. To put this concretely, at a T-junction, where trajectories can go either left or right, the optimal model will yield a trajectory going straight, which is clearly a fundamentally undesirable behaviour. Mathematically, the problem lies in the additive nature of the metric – each mode can be optimized independently of the others. This can be avoided by instead considering a likelihood based metric, such as the following one:

$$\text{cNLL}(\mathcal{D}) \coloneqq \frac{1}{N}\sum_{n=1}^{N}\left\{-\ln\left[\sum_{d=1}^{D} c^{(d)} \prod_{t=1}^{T} \mathcal{N}(\boldsymbol{y}_{t,i}^{*};\ \boldsymbol{s}_t^{(d)}(\boldsymbol{x}_i;\boldsymbol{\theta}), \boldsymbol{\Sigma}=\mathbf{1})\right]\right\} - T\ln 2\pi \tag{15}$$

Under the following metric, which assumes that each mode is modelled using a Normal distribution of fixed variance, an optimal model would place a Normal over each mode and weight them appropriately. This can be clearly demonstrated using the following numerical example:

$$y \sim \mathrm{p}(y) = 0.5 \cdot \mathcal{N}(x, 10, 1) + 0.5 \cdot \mathcal{N}(x, -10, 1) \tag{16}$$

$$\begin{aligned}&\mathbb{E}_{\mathrm{p}}(y)[\text{wADE}(y, \boldsymbol{s}^{(1:2)} = [10, -10], \boldsymbol{c} = [0.5, 0.5])] = 201.5\\ &\mathbb{E}_{\mathrm{p}}(y)[\text{wADE}(y, \boldsymbol{s}^{(1:2)} = [0, 0], \boldsymbol{c} = [0.5, 0.5])] = 101.50\end{aligned} \tag{17}$$

$$\mathbb{E}_{\mathrm{p}}(y)[\text{cNLL}(y, \boldsymbol{s}^{(1:2)} = [10, -10], \boldsymbol{c} = [0.5, 0.5])] = 1.09 \tag{18}$$

$$\mathbb{E}_{\mathrm{p}}(y)[\text{cNLL}(y, \boldsymbol{s}^{(1:2)} = [0, 0], \boldsymbol{c} = [0.5, 0.5])] = 50.75 \tag{19}$$

Where we have a bimodal Gaussian mixture distribution with modes at -10, 10. We assume we have a model which predicts the means of two trajectories with equal weight. We have two situations: either the model yields two distinct modes at -10, 10 or a collapsed mode at 0 (the median). We can see that predicting the median will yield a lower weightedADE and correctly predicting two distinct modes will yield the lower cNLL. It is important to highlight that this argument holds *in expectation* and is relevant to situations which contain inherent ambiguity and multi-modality. Note that the offset $T \ln 2\pi$ is used to make assure that the minimal value of this metric is 0, so that it can be used for error-retention and F1-retention plots.

Per-Prediction Request Confidence-Aware Metrics. In addition to making a decision amongst many possible trajectories in a particular situation, a motion planning agent should know when, in general, any trajectories it predicts will be inaccurate (e.g., due to unfamiliarity of the setting, or inherent ambiguity in the path of the vehicle for which a prediction is requested). We evaluate the quality of uncertainty quantification jointly with robustness to distributional shift using the retention-based metrics described in Section 2, with the per–prediction request uncertainty scores determining retention order. Note that each retention curve is plotted with respect to a particular error metric above (e.g., we consider AUC for retention with respect to the cNLL metric introduced above, written as R-AUC). Additionally, we also assess whether the per–prediction uncertainty scores can be used to discriminate between in-domain and shifted scenes. In this case, quality is assessed via area under a ROC curve (ROC-AUC).

E.4 Experimental Setup

Robust Imitative Planning. In detail, we use the following approach for trajectory and confidence score generation.

1) **Trajectory Generation.** Given a scene input $\boldsymbol{x}$, K ensemble members generate G trajectories.[15]
2) **Trajectory Scoring.** We score each of the G trajectories by computing a log probability under each of the K trained likelihood models.
3) **Per-Trajectory Confidence Scores.** We aggregate the $G \cdot K$ resulting log probabilities to G scores using a per-trajectory aggregation operator $\oplus_{\text{trajectory}}$.[16] By aggregating over the log-likelihood estimates sampled from the model posterior (i.e., contributed by each ensemble member), we obtain a robust score for each of the G trajectories [72].

[15] In practice, each ensemble member generates the same number of trajectories Q, s.t. $G = K \cdot Q$.

[16] For example, applying a `min` aggregation is informed by robust control literature [80] in which we aim to optimize for the worst-case scenario, as measured by the log-likelihood of the "most pessimistic" model for a given trajectory.

4) **Trajectory Selection.** Among the G trajectories, the RIP ensemble produces the top D trajectories as determined by their corresponding G per-trajectory confidence scores, where D is a hyperparameter.

5) **Per–Prediction Request Uncertainty Score.** We aggregate the D top per-trajectory confidence scores to a single uncertainty score U using the aggregator $\oplus_{\text{pred-req}}$.[17] This value conveys the ensemble's estimated uncertainty for a given scene context and a particular prediction request.

6) **Confidence Reporting.** We obtain scores $c^{(d)}$ by applying a `softmax` to the D top per-trajectory confidence scores. We report these $c^{(d)}$ and U (computed in step **5**) as our final per-trajectory confidence scores and per–prediction request uncertainty score, respectively.

To summarize, our implementation of RIP for motion prediction produces D trajectories and corresponding normalized per-trajectory scores $\{c^{(d)} \mid d \in 1, \ldots, D\}$, as well as an aggregated uncertainty score U for the overall prediction request.

Backbone Likelihood Model. We consider two different model classes as ensemble members: a simple behavioral cloning agent with a Gated Recurrent Unit decoder (BC) [81, 73] and a Deep Imitative Model (DIM) [74] with an autoregressive flow decoder [82], following [72]. In both cases, we model the likelihood of a trajectory $\boldsymbol{y}$ in context $\boldsymbol{x}$ to come from an expert (i.e., from the distribution of ground truth trajectories), with learnable parameters $\boldsymbol{\theta}$, as

$$q(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\theta}) = \prod_{t=1}^{T} p(s_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}; \boldsymbol{\theta}) \; = \prod_{t=1}^{T} \mathcal{N}(s_t; \mu(\boldsymbol{y}_{<t}, \boldsymbol{x}; \boldsymbol{\theta}), \Sigma(\boldsymbol{y}_{<t}, \boldsymbol{x}; \boldsymbol{\theta})), \tag{20}$$

where $\mu(\cdot; \boldsymbol{\theta})$ and $\Sigma(\cdot; \boldsymbol{\theta})$ are two heads of a recurrent neural network with shared torso. Hence we assume that the conditional densities are normally distributed, and learn those parameters through maximum likelihood estimation. Notably, for the BC model, we found that conditioning on samples $\hat{\boldsymbol{y}}_{<t}$ instead of ground truth values $\boldsymbol{y}_{<t}$ (where usage of ground truth is often referred to as teacher forcing in RNN literature) significantly improved performance across all datasets and metrics.

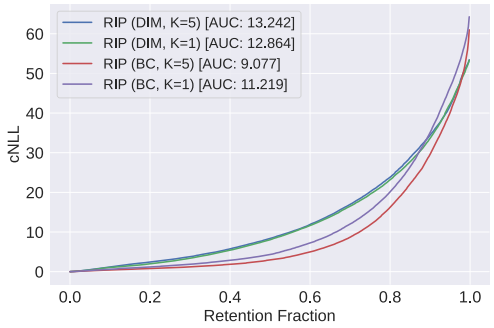
Uncertainty Estimation Methods. The above ensembling is done using multiple stochastic models trained with different random seeds, as introduced in Deep Ensembles [14]. For each ensemble member, we generate Q trajectories. We can also use a Monte Carlo Dropout [13] approach for each ensemble member, in which we sample new dropout masks *at test time* during each of the Q forward passes (and corresponding trajectory generations). Following [75] we refer to the combination of this uncertainty estimation method with ensembling as Dropout Ensembles. Previous work has investigated the benefits of Deep Ensembles from a loss landscape perspective [83], and found that Deep Ensembles tend to explore diverse modes in function space, whereas approximate variational methods such as Monte Carlo Dropout explore around a particular mode. Dropout Ensembles are hence motivated as ensembles of variational methods which aim to consider a diverse set of modes, with local exploration around each mode.

Setup. We report performance of RIP across the two backbone models – Behavioral Cloning (BC) [73] and Deep Imitative Model (DIM) [74] – as well as the two uncertainty estimation methods – Deep Ensembles [14] and Dropout Ensembles [13, 75]. We evaluate RIP on development (`dev`) and evaluation (`eval`) datasets in in-distribution (In), distributionally shifted (Shifted), and combined in-distribution and shifted (Full) settings. With both backbone model classes we vary the number of ensemble members $K \in \{1, 3, 5\}$, train with learning rate 1e-4, use a cosine annealing LR schedule with 1 epoch warmup, and use gradient clipping at 1. We sample $Q = 10$ trajectories from each of the ensemble members. We consider two types of aggregation: "Lower Quartile" in which we compute the mean minus the standard deviation $\mu - \sigma$ of the input scores, and "Model Averaging" (MA) in which we compute the mean μ of the input scores. LQ reflects the intuition to assign a high score to a trajectory when the ensemble members assign it a high score on average, and tend to be certain (have a low standard deviation) in their scoring; MA reflects only the prior intuition. This aggregation strategy (LQ or MA) is used as both the per-trajectory aggregation operator $\oplus_{\text{trajectory}}$ and the per–prediction request aggregation operator $\oplus_{\text{pred-req}}$ (where the latter is followed by negation to obtain an uncertainty, as opposed to a confidence). We fix the RIP ensemble at all K to produce the top $D = 5$ trajectories as ranked by their per-trajectory confidence score.

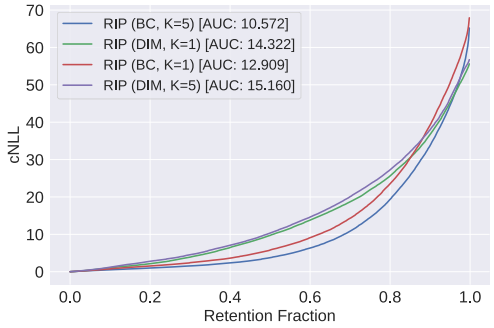
[17] In practice, this is done by applying the aggregation (e.g., $\oplus_{\text{pred-req}} = \texttt{mean}$) to the confidences $c^{(d)}$, and then *negating* to obtain the uncertainty score U.

E.5 Additional Results

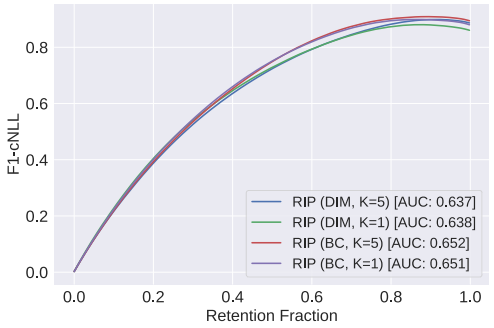
Below, we report predictive performance using standard-metrics, robustness and uncertainty quantification metrics, and retention plots across the RIP variants.



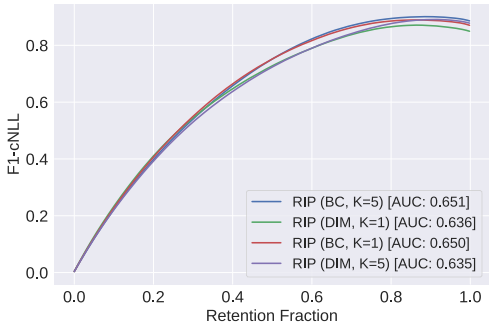
(a) Full `dev` cNLL retention.



(b) Full `eval` cNLL retention.



(c) Full `dev` F1-cNLL retention.



(d) Full `eval` F1-cNLL retention.

Figure 18: cNLL and F1-cNLL retention curves on the Full (i.e., containing both the in-distribution and distributionally shifted datapoints) `dev` (left column) and `eval` (right column) partitions of the Vehicle Motion Prediction dataset. Top row: retention on cNLL (lower ↓ AUC is better). Bottom row: retention on F1-cNLL (higher ↑ AUC is better). We vary the backbone model and number of ensemble members, fix the Model Averaging (MA) aggregation strategy for the per-trajectory aggregation operator $\oplus_{\text{trajectory}}$ and the per–prediction request aggregation operator $\oplus_{\text{pred-req}}$ (based on results from Table 7), and otherwise use the standard RIP settings enumerated in Appendix E.4.

Table 20: *Predictive performance* of RIP, across model backbones (behavioral cloning (BC) [73] and Deep Imitative Model (DIM) [74]) and uncertainty estimation methods (Deep Ensembles [14] and Dropout Ensembles [75]). Each section contains losses computed over the in-distribution (In), distributionally shifted (Shifted), and combined (Full) development and evaluation datasets. Altogether, we vary the backbone model, uncertainty estimation method, aggregation strategy (applied for both the per-trajectory aggregation operator $\oplus_{\text{trajectory}}$ and the per–prediction request aggregation operator $\oplus_{\text{pred-req}}$), and the number of ensemble members K. See Appendix E.4 for setup details.

Dataset	Method	Model	minADE ↓			weightedADE ↓			minFDE ↓			weightedFDE ↓			cNLL ↓		
			In	Shifted	Full	In	Shifted	Full	In	Shifted	Full	In	Shifted	Full	In	Shifted	Full
Dev	Deep Ensemble	BC, LQ, K=1	0.818	0.960	0.835	1.088	1.245	1.107	1.718	2.113	1.765	2.368	2.777	2.417	59.64	98.54	64.29
		BC, LQ, K=3	0.780	0.909	0.795	1.040	1.170	1.056	1.638	2.018	1.683	2.254	2.609	2.297	54.78	87.81	58.73
		BC, LQ, K=5	0.766	0.888	0.780	1.017	1.138	1.031	1.618	1.980	1.661	2.214	2.552	2.254	56.45	90.25	60.49
		BC, MA, K=1	0.818	0.960	0.835	1.088	1.245	1.107	1.718	2.113	1.765	2.368	2.777	2.417	59.64	98.54	64.29
		BC, MA, K=3	0.780	0.908	0.795	1.034	1.166	1.050	1.641	2.018	1.686	2.249	2.611	2.292	55.00	88.45	59.00
		BC, MA, K=5	0.765	0.887	0.779	1.012	1.133	1.026	1.617	1.976	1.660	2.210	2.551	2.251	56.86	91.54	61.01
		DIM, LQ, K=1	0.750	0.818	0.758	1.523	1.583	1.530	1.497	1.720	1.524	3.472	3.639	3.492	50.66	73.00	53.34
		DIM, LQ, K=3	0.717	0.787	0.725	1.407	1.470	1.415	1.467	1.687	1.493	3.219	3.397	3.240	48.88	**70.93**	51.52
		DIM, LQ, K=5	0.720	0.787	0.728	1.399	1.470	1.407	1.487	1.704	1.513	3.202	3.397	3.225	51.12	72.87	53.72
		DIM, MA, K=1	0.750	0.818	0.758	1.523	1.583	1.530	1.497	1.720	1.524	3.472	3.639	3.492	50.66	73.00	53.34
		DIM, MA, K=3	0.717	**0.785**	0.725	1.410	1.475	1.418	1.466	**1.685**	1.492	3.226	3.409	3.248	**48.74**	71.30	**51.44**
		DIM, MA, K=5	0.719	0.786	0.727	1.399	1.469	1.408	1.482	1.698	1.508	3.202	3.393	3.225	50.85	72.45	53.43
	Dropout Ensemble	BC, LQ, K=1	0.803	0.908	0.815	1.116	1.236	1.130	1.649	1.952	1.685	2.409	2.718	2.446	55.98	82.49	59.15
		BC, LQ, K=3	0.741	0.853	0.754	1.013	1.132	1.028	1.542	1.873	1.581	2.209	2.545	2.249	53.01	83.93	56.71
		BC, LQ, K=5	0.759	0.878	0.773	**1.008**	1.127	**1.023**	1.605	1.960	1.648	**2.204**	**2.538**	**2.244**	55.58	88.78	59.55
		BC, MA, K=1	0.803	0.908	0.815	1.116	1.236	1.130	1.649	1.952	1.685	2.409	2.718	2.446	55.98	82.49	59.15
		BC, MA, K=3	0.739	0.850	0.752	1.020	1.135	1.033	1.534	1.864	1.574	2.223	2.553	2.263	53.09	83.81	56.76
		BC, MA, K=5	0.757	0.877	0.771	1.010	**1.126**	1.024	1.597	1.952	1.640	2.209	2.539	2.248	55.82	89.57	59.86
		DIM, LQ, K=1	0.750	0.831	0.759	1.498	1.587	1.509	1.510	1.757	1.539	3.432	3.662	3.459	52.57	76.54	55.44
		DIM, LQ, K=3	**0.716**	0.786	0.725	1.412	1.473	1.419	1.466	1.687	1.493	3.234	3.408	3.254	49.69	72.58	52.43
		DIM, LQ, K=5	0.723	0.793	0.731	1.409	1.475	1.417	1.494	1.717	1.521	3.224	3.408	3.246	51.25	73.47	53.91
		DIM, MA, K=1	0.750	0.831	0.759	1.498	1.587	1.509	1.510	1.757	1.539	3.432	3.662	3.459	52.57	76.54	55.44
		DIM, MA, K=3	**0.716**	0.786	**0.724**	1.414	1.479	1.422	**1.465**	**1.685**	**1.491**	3.238	3.420	3.260	49.38	71.86	52.07
		DIM, MA, K=5	0.721	0.793	0.729	1.409	1.474	1.417	1.489	1.717	1.516	3.224	3.405	3.246	50.99	73.64	53.70
Eval	Deep Ensemble	BC, LQ, K=1	0.829	1.084	0.880	1.104	1.407	1.164	1.733	2.420	1.870	2.394	3.197	2.555	60.20	98.82	67.93
		BC, LQ, K=3	0.792	1.026	0.839	1.056	1.326	1.110	1.658	2.297	1.786	2.284	3.005	2.429	55.97	90.54	62.89
		BC, LQ, K=5	0.777	1.015	0.825	1.032	1.303	1.086	1.636	2.283	1.765	2.242	2.964	2.386	57.26	93.92	64.60
		BC, MA, K=1	0.829	1.084	0.880	1.104	1.407	1.164	1.733	2.420	1.870	2.394	3.197	2.555	60.20	98.82	67.93
		BC, MA, K=3	0.792	1.025	0.838	1.050	1.319	1.104	1.661	2.294	1.788	2.278	2.997	2.422	55.94	90.53	62.87
		BC, MA, K=5	0.777	1.014	0.824	1.028	1.299	1.082	1.636	2.278	1.765	2.238	2.957	2.382	57.75	95.00	65.20
		DIM, LQ, K=1	0.759	0.942	0.796	1.551	1.883	1.618	1.511	1.983	1.605	3.536	4.376	3.704	50.50	76.00	55.60
		DIM, LQ, K=3	0.726	0.914	0.764	1.433	1.756	1.498	1.481	1.972	1.579	3.277	4.094	3.440	49.45	76.66	54.89
		DIM, LQ, K=5	0.729	0.921	0.768	1.422	1.757	1.489	1.498	2.007	1.600	3.253	4.098	3.422	51.61	79.71	57.24
		DIM, MA, K=1	0.759	0.942	0.796	1.551	1.883	1.618	1.511	1.983	1.605	3.536	4.376	3.704	50.50	76.00	55.60
		DIM, MA, K=3	0.726	0.912	0.763	1.437	1.759	1.502	1.478	1.967	1.576	3.286	4.101	3.449	49.09	76.07	54.49
		DIM, MA, K=5	0.728	0.918	0.766	1.424	1.754	1.490	1.493	2.000	1.595	3.256	4.093	3.424	51.19	78.85	56.73
	Dropout Ensemble	BC, LQ, K=1	0.812	1.038	0.857	1.128	1.410	1.184	1.664	2.267	1.784	2.430	3.170	2.578	56.57	86.28	62.52
		BC, LQ, K=3	0.751	0.972	0.795	1.029	**1.297**	1.082	1.558	2.154	1.677	2.238	**2.948**	2.380	53.94	86.68	60.49
		BC, LQ, K=5	0.770	1.008	0.817	**1.024**	**1.297**	**1.079**	1.623	2.268	1.752	**2.233**	2.957	**2.378**	56.49	92.77	63.75
		BC, MA, K=1	0.812	1.038	0.857	1.128	1.410	1.184	1.664	2.267	1.784	2.430	3.170	2.578	56.57	86.28	62.52
		BC, MA, K=3	0.749	0.970	0.794	1.036	1.305	1.090	1.551	2.147	1.670	2.253	2.963	2.395	54.07	86.94	60.65
		BC, MA, K=5	0.768	1.004	0.815	1.027	1.299	1.081	1.615	2.253	1.743	2.239	2.958	2.383	56.90	93.27	64.18
		DIM, LQ, K=1	0.739	0.924	0.776	1.478	1.815	1.546	1.474	**1.949**	1.569	3.380	4.239	3.552	49.90	75.31	54.98
		DIM, LQ, K=3	0.722	0.910	0.760	1.431	1.763	1.497	1.470	1.967	1.569	3.266	4.112	3.435	49.30	75.24	54.49
		DIM, LQ, K=5	0.729	0.929	0.769	1.430	1.769	1.497	1.497	2.027	1.603	3.268	4.126	3.440	50.77	80.02	56.63
		DIM, MA, K=1	0.739	0.924	0.776	1.478	1.815	1.546	1.474	**1.949**	1.569	3.380	4.239	3.552	49.90	75.31	54.98
		DIM, MA, K=3	**0.720**	**0.907**	**0.758**	1.432	1.760	1.497	**1.465**	1.960	**1.564**	3.267	4.107	3.435	**48.74**	**74.70**	**53.93**
		DIM, MA, K=5	0.728	0.925	0.767	1.431	1.766	1.498	1.494	2.017	1.599	3.269	4.120	3.439	50.51	79.30	56.28

Table 21: *Uncertainty and robustness performance* of RIP across the two backbone models (BC and DIM) and uncertainty estimation methods (Deep Ensemble and Dropout Ensemble). The error metric for computing the area under the rejection curve (R-AUC) and area under the F1 curve (F1-AUC) is **cNLL**. We use a threshold of 25 for the F1 metrics, which approximately corresponds to a 1 meter deviation on all trajectories. See Appendix E.4 for setup details.

Dataset	Method	Model	R-AUC ↓			F1-AUC (%) ↑			F1@95% ↑			ROC-AUC (%) ↑
			In	Shifted	Full	In	Shifted	Full	In	Shifted	Full	
Dev	Deep Ensemble	BC, LQ, K=1	11.06	13.91	11.22	64.9	66.7	65.1	89.1	90.2	89.3	51.0
		BC, LQ, K=3	11.26	11.69	11.18	63.4	66.0	63.8	88.5	90.3	88.8	46.7
		BC, LQ, K=5	9.68	10.38	9.62	64.3	66.4	64.6	89.7	91.0	90.0	47.3
		BC, MA, K=1	11.06	13.91	11.22	64.9	66.7	65.1	89.1	90.2	89.3	51.0
		BC, MA, K=3	9.31	10.73	9.31	64.8	66.5	65.0	90.3	91.3	90.6	48.6
		BC, MA, K=5	9.07	10.47	9.08	64.9	66.5	65.2	90.4	91.3	90.6	49.2
		DIM, LQ, K=1	12.54	15.28	12.86	63.6	64.8	63.8	87.2	88.8	87.4	**51.8**
		DIM, LQ, K=3	12.30	14.51	12.57	63.7	64.9	63.8	89.3	89.9	89.3	51.4
		DIM, LQ, K=5	12.87	15.01	13.14	63.5	64.8	63.7	89.7	90.2	89.7	51.4
		DIM, MA, K=1	12.57	15.10	12.86	63.7	64.9	63.8	87.2	88.8	87.4	**51.8**
		DIM, MA, K=3	12.38	14.46	12.64	63.7	64.9	63.8	89.2	89.9	89.3	51.4
		DIM, MA, K=5	12.97	15.10	13.24	63.5	64.8	63.7	89.6	90.2	89.7	51.4
	Dropout Ensemble	BC, LQ, K=1	8.87	10.00	8.87	**65.3**	**67.1**	**65.6**	89.7	90.4	89.9	51.2
		BC, LQ, K=3	**8.11**	**9.53**	**8.14**	64.9	66.5	65.1	90.6	91.3	**90.8**	50.9
		BC, LQ, K=5	8.28	9.60	8.28	65.0	66.6	65.2	90.5	91.3	90.7	50.7
		BC, MA, K=1	8.87	9.99	8.87	**65.3**	**67.1**	**65.6**	89.7	90.4	89.9	51.2
		BC, MA, K=3	8.53	9.79	8.54	64.9	66.5	65.1	**90.7**	**91.4**	**90.8**	50.3
		BC, MA, K=5	8.89	10.23	8.90	64.9	66.5	65.2	90.5	**91.4**	90.7	50.2
		DIM, LQ, K=1	12.57	16.41	13.03	63.8	64.7	63.9	87.6	89.1	87.8	51.5
		DIM, LQ, K=3	12.37	14.91	12.69	63.7	64.8	63.8	89.2	90.0	89.3	51.3
		DIM, LQ, K=5	12.94	15.18	13.22	63.6	64.8	63.7	89.6	90.2	89.7	51.4
		DIM, MA, K=1	12.61	16.30	13.06	63.8	64.8	63.9	87.6	89.1	87.7	51.6
		DIM, MA, K=3	12.49	14.80	12.79	63.6	64.8	63.8	89.2	90.0	89.3	51.4
		DIM, MA, K=5	13.05	15.20	13.33	63.5	64.8	63.7	89.5	90.2	89.6	51.4
Eval	Deep Ensemble	BC, LQ, K=1	11.16	20.84	12.91	64.9	65.5	65.0	88.9	85.6	88.4	52.8
		BC, LQ, K=3	11.31	17.09	12.38	63.4	64.8	63.7	88.4	86.4	88.0	50.9
		BC, LQ, K=5	9.77	15.95	10.88	64.3	65.4	64.5	89.5	87.1	89.1	51.4
		BC, MA, K=1	11.17	20.84	12.91	64.9	65.5	65.0	88.9	85.6	88.4	52.8
		BC, MA, K=3	9.40	16.76	10.73	64.8	65.6	65.0	90.2	87.5	89.7	51.3
		BC, MA, K=5	9.20	16.85	10.57	65.0	65.6	65.1	90.2	87.5	89.7	52.1
		DIM, LQ, K=1	12.78	20.78	14.28	63.5	63.7	63.6	86.9	83.9	86.3	52.0
		DIM, LQ, K=3	12.66	21.40	14.32	63.6	63.9	63.7	89.1	86.0	88.5	51.4
		DIM, LQ, K=5	13.26	22.59	15.05	63.5	63.8	63.6	89.5	86.5	88.9	51.2
		DIM, MA, K=1	12.81	20.83	14.32	63.6	63.8	63.6	86.9	83.9	86.3	51.8
		DIM, MA, K=3	12.74	21.51	14.42	63.6	63.9	63.7	89.1	86.0	88.5	51.1
		DIM, MA, K=5	13.37	22.68	15.16	63.5	63.7	63.5	89.5	86.5	88.9	50.9
	Dropout Ensemble	BC, LQ, K=1	9.06	15.49	10.22	**65.3**	**66.1**	**65.5**	89.5	86.4	89.0	53.7
		BC, LQ, K=3	**8.22**	**14.83**	**9.39**	64.9	65.6	65.1	**90.5**	87.5	**90.0**	53.9
		BC, LQ, K=5	8.39	15.16	9.57	65.0	65.7	65.2	90.4	87.6	89.9	**54.5**
		BC, MA, K=1	9.07	15.50	10.22	**65.3**	**66.1**	**65.5**	89.5	86.4	89.0	53.7
		BC, MA, K=3	8.69	15.90	9.99	64.9	65.6	65.1	**90.5**	**87.7**	**90.0**	53.0
		BC, MA, K=5	9.05	16.69	10.41	65.0	65.6	65.1	90.4	87.6	89.9	53.2
		DIM, LQ, K=1	12.45	20.27	13.92	63.6	63.7	63.6	87.7	84.7	87.1	51.8
		DIM, LQ, K=3	12.63	21.32	14.29	63.7	63.9	63.7	89.1	86.1	88.6	51.3
		DIM, LQ, K=5	13.22	22.78	15.04	63.5	63.8	63.6	89.4	86.3	88.8	51.2
		DIM, MA, K=1	12.51	20.33	14.00	63.6	63.8	63.7	87.7	84.7	87.1	51.5
		DIM, MA, K=3	12.73	21.43	14.40	63.6	63.9	63.7	89.1	86.0	88.5	51.1
		DIM, MA, K=5	13.36	22.85	15.19	63.5	63.8	63.6	89.4	86.3	88.8	50.9