UNIQUE: A Framework for Uncertainty Quantification Benchmarking

Jessica Lanini, Minh Tam Davide Huynh, Gaetano Scebba, Nadine Schneider, and Raquel Rodríguez-Pérez*

Novartis Biomedical Research, Novartis Campus, 4002 Basel, Switzerland

*Corresponding author

R.R.P. Phone: 41-795-42-2309, E-mail: raquel.rodriguez_perez@novartis.com

Abstract

Machine learning (ML) models have become key in decision-making for many disciplines, including drug discovery and medicinal chemistry. ML models are generally evaluated prior to their usage for high-stake decisions, such as compound synthesis or experimental testing. However, no ML model is robust and predictive in all real-world scenarios. Therefore, uncertainty quantification (UQ) in ML predictions has gained importance in recent years. Many investigations have focused on developing methodologies that provide accurate uncertainty estimates for ML-based predictions. Unfortunately, there is no UQ strategy that consistently provides robust estimates about model's applicability on new samples. Depending on the dataset, prediction task, and algorithm, accurate uncertainty estimations might be unfeasible to obtain. Moreover, the optimum UQ metric also varies across applications, and previous investigations have shown a lack of consistency across benchmarks. Herein, the UNIQUE (UNcertaInty QUantification bEnchmarking) framework is introduced to facilitate the comparison of UQ strategies in ML-based predictions. This Python library unifies the benchmarking of multiple UQ metrics, including the calculation of non-standard UQ metrics (combining information from the dataset and model), and providing a comprehensive evaluation. In such framework, UQ metrics are evaluated for different application scenarios, e.g. eliminate the predictions with the lowest confidence or obtain a reliable uncertainty estimate for an acquisition function. Taken together, this library will help to standardize UQ investigations and evaluate new methodologies.

Keywords: Uncertainty quantification, uncertainty estimation, applicability domain, machine learning, benchmarking, model evaluations, decision-making.

1. Introduction

Machine learning (ML) models have been increasingly applied to various fields, including life sciences and drug discovery[1–4]. Even when high performance is observed in model evaluations, some predictions might be unreliable in real-world applications. Assessing model's robustness, stability, and applicability to new samples is challenging and uncertainty quantification (UQ) has gained great importance for ML-based decision-making[5–7]. Providing uncertainty estimates alongside model's predictions equips end-users with crucial information on when or to what degree to trust ML predictions for their decisions.

UQ strategies aim at estimating the likelihood of outcomes associated with a prediction, which can be reported as probabilities for classification or prediction intervals for regression tasks[8–11]. Uncertainty in predictions is strongly related to the data used for training the ML model. The 'applicability domain' concept has been used for many years to assess uncertainty in terms of the feature space, also known as covariate shift, generally using a distance to the training set[12–14]. This type of uncertainty metrics related to the data (and generally based on distances) can be termed *data-based* UQ metrics. With the rise of ML, UQ metrics based on models' output have become more standard. Such *model-based* UQ metrics can be obtained through multiple strategies depending on the ML algorithm, and often consist of a variance. Some examples include the variance from the posterior distribution of a Bayesian model, from an ensemble of models (generated by bootstrapping, Monte Carlo dropout or random initializations) or from a mean-variance estimation that uses a negative log likelihood (NLL) loss[15–21]. Such variance values are generally used to obtain a prediction interval.

The relevance of UQ in ML applications has given birth to tools spanning different fields[22–26]. For instance, Fortuna and MAIPE libraries calibrate uncertainty estimations and provide

conformal sets and prediction intervals[22,26], and the uncertainty toolbox by Chung et al. also includes functions to calculate evaluation scores and compare UQ metrics[25]. However, limitations have been reported in previous investigations both for *data*- and *model*-based UQ methods. Ovadia et al. showed the lack of robustness of uncertainty estimates since UQ's quality consistently degraded with increasing data shift[9]. Moreover, there is generalized lack of correlation between UQ methods and model errors[27]. It has been reported that the most suitable UQ method highly depends on the modeling task and, even more interestingly, the type of evaluation[28–30]. Hence, there is not a reliable and consistently superior method to quantify prediction uncertainty. This highlights the need for a consistent and standardized benchmarking framework that considers the application scenario when evaluating UQ methodologies[31].

Herein, the UNIQUE (UNcertaInty Quantification bEnchmarking) framework is presented to facilitate the benchmarking of different strategies and metrics to quantify uncertainty in ML-based predictions. To the best of the authors' knowledge, no tool unifies the benchmark of multiple UQ methodologies, calculation of additional UQ metrics, and a comprehensive evaluation. In this work, the design and implementation of the UNIQUE Python library is detailed and illustrated with a practical example on compound property prediction.

2. Design and implementation

The UNIQUE library allows calculating, combining, and benchmarking multiple UQ metrics for a given dataset and model. Comprehensive evaluation statistics and intuitive visualizations are generated to assess the quality of the uncertainty estimates (UQ metrics). Current UNIQUE implementation supports UQ for regression tasks. UNIQUE is a model-agnostic tool, meaning it is not dependent on any specific upstream ML model or its building platform and training functionalities. To achieve that, UNIQUE requires the user to provide information about the dataset and model's outputs. As shown in **Figure 1**, the framework is composed of four main components: (i) input types, (ii) uncertainty metrics (or methods), (iii) error models, and (iv) evaluation metrics (or scores). The *input types* module defines the data features or ML model outputs that are provided by the user. The *uncertainty metrics* component pre-processes or calculates UQ metrics based on the input types. The *error models* component allows building ML models to predict the error of the original model's predictions. Error models' predictions are considered as a special case of UQ metric. Finally, the *evaluation metrics* module includes different functionalities to estimate and compare the quality of all calculated UQ metrics.

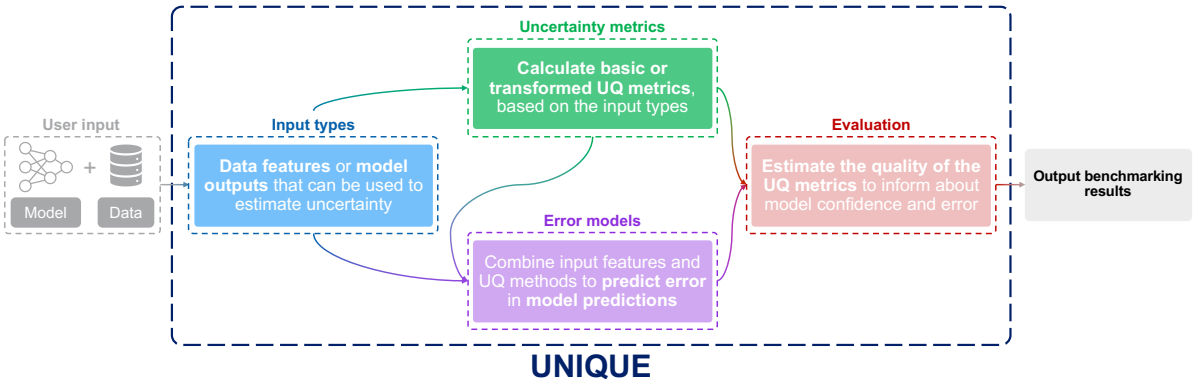

Figure 1. General scheme of the UNIQUE library. The main modules of the UNIQUE library are shown with a brief description. The UNIQUE framework allows obtaining a comprehensive benchmark of UQ metrics from provided original ML model outputs and input

data. UNIQUE also includes the capability of calculating additional UQ metrics, error models, and perform different evaluation types to estimate the quality of UQ metrics.

2.1. User input

The user needs to create an input file and a configuration file to run UNIQUE's pipeline. The input file should have a table-like format with the following columns (or keys):

- **IDs**: unique identifiers (IDs) (if no available, the index of the table is used).
- **Labels**: target labels used to train the ML model.
- **Predictions**: ML model predictions (a single value per ID is expected).
- **Data split**: subset each datapoint belongs to. UNIQUE can be run with training and test subsets only, but some functionalities require a calibration set.

Depending on the UQ metrics to evaluate, additional columns can be added accordingly. Two additional types of columns are accepted:

- **Data features column(s)**: featurization of each sample. Features generally correspond to the ones used in the original ML model, but other representations can also be used.
- **Model outputs column(s)**: output(s) related to the ML model. For example, a column containing the prediction variance per each sample.

The configuration file (YAML) contains all the specifications needed to retrieve and run the UNIQUE pipeline, such as information for different inputs and which UQ metric to evaluate. Once both files are prepared, the UNIQUE benchmarking workflow can be run end-to-end through the *unique.Pipeline* object.

2.2. Input types

This module defines the type of input for each column included in the data table provided by the user. Following the distinction between *data*-based and *model*-based UQ metrics, UNIQUE identifies two classes of inputs: *data*- (or *feature*-based) and *model*-based input types. *Feature input types* correspond to columns containing featurization of the samples, such as real-valued, counts or binary vectors. Since UNIQUE currently focuses on regression problems, *model input types* correspond to model variances, which might be obtained through multiple methods (e.g. ensemble model or Bayesian model).

The user could directly provide a UQ metric (e.g. variance) or information to generate a UQ metric within UNIQUE (e.g. a feature vector to calculate the distance to the training set). Each input type is associated with a set of UQ metrics that could be calculated (supported UQ metrics), which are detailed below.

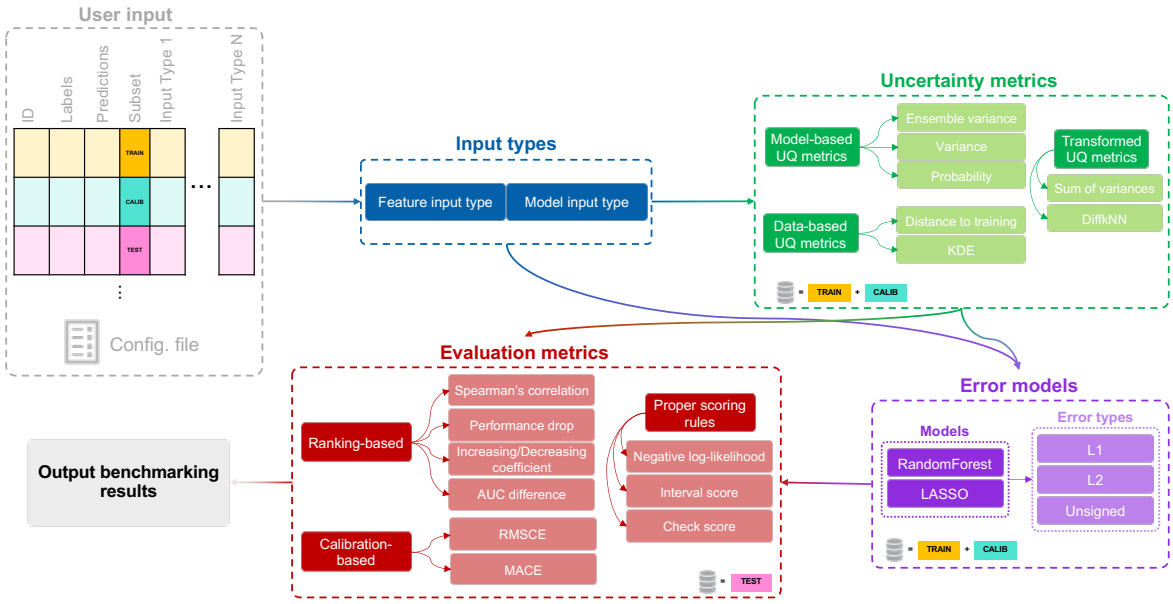

Figure 2. General workflow and components of the UNIQUE framework. The input from the user is a tabular dataset containing identifiers, labels, predictions, subset labels, and other feature- or model-related inputs. Columns of this input table are mapped to UNIQUE's *input*

types, which are defined as *UQ metrics* or used to compute *UQ metrics. Error models* are considered as a special category of UQ metric and consist of ML models that predict the error of the original model. In the *evaluation* module, all UQ metrics are evaluated with different scores (ranking-based, calibration-based or proper scoring rules). The final output of UNIQUE is a benchmark of several UQ metrics specific to the input dataset and ML model.

2.3. Uncertainty metrics

This UNIQUE component focuses on calculating and defining UQ metrics, which can be either derived from *data*- or *model*-based inputs (*base* UQ metrics), or a combination of both (*transformed* UQ metrics). Such transformed UQ metrics have been introduced to enable the benchmark of more complex methods that potentially combine the strengths of data- and model-based UQ metrics.

Data-based UQ metrics are either distances to the *k*-nearest neighbors (k-NN) in the training set or kernel density estimations (KDEs) from the training set. k-NN is available with **Manhattan, Euclidean,** and **Tanimoto distances**, whereas KDE is implemented with Gaussian or Exponential kernels, and Euclidean or Manhattan distance in the following combinations: **Gaussian-Euclidean, Gaussian-Manhattan,** and **Exponential-Manhattan KDE**. *Model*-based UQ metrics correspond to the estimated **Variance**. All mentioned data- and model-based UQ metrics can be referred as *base* UQ metrics.

Transformed UQ metrics combine data- and model-based uncertainty estimates to generate more complex uncertainty estimates. The **sum of variances** is calculated to include information from multiple sources of variance estimations, primarily a distance metric (data-based UQ) and model predicted variance (model-based UQ). For that, distances are first converted to variances assuming a linear relationship and using the calibrated NLL, as described in Hirschfeld et al.[29]:

$$\hat{\sigma}^2(x) := aU(x) + b$$

where coefficients a and b are computed from the minimization of the NLL of errors in the validation set:

$$a_*, b_* = \underset{a,b}{argmin} \frac{1}{2} \sum_{x,y \in D_{val}} ln(2\pi) + ln(\hat{\sigma}^2(x)) + \frac{(y - \mu(x))^2}{2\hat{\sigma}^2(x)}$$

DiffkNN is another transformed UQ metric that was adapted from Sheridan et al.[32] and further generalized for UQ estimations. In UNIQUE, DiffkNN is defined as the absolute difference between predicted values of a test sample and its k-NN in the training set. Here, the implementation is also available with UQ metrics instead of predicted values. For instance, absolute difference between the variance of each sample and its closest neighbors from the training set (using a given distance metric and features).

2.4. Error models

UNIQUE supports the generation of ML models to predict the error of the original model. These so-called *error models* have been proposed in previous works both to predict the absolute and squared differences between predicted and observed values[33,34]. As illustrated in **Figure 3**, UNIQUE includes predictions of L1, L2, and unsigned errors with two algorithms: least absolute shrinkage and selection operator (Lasso)[35] and random forest (RF)[36]. Three subset of input features are considered to build the error model: (i) original model prediction, UQ metrics, and input features provided by the user, (ii) original model prediction and UQ metrics, (iii) original model prediction and transformed UQ metrics. Error predictions are a special case of UQ metric, which can also be evaluated accordingly in UNIQUE's benchmark. In this case, a calibration subset is required to have a wider distribution of error values[33].

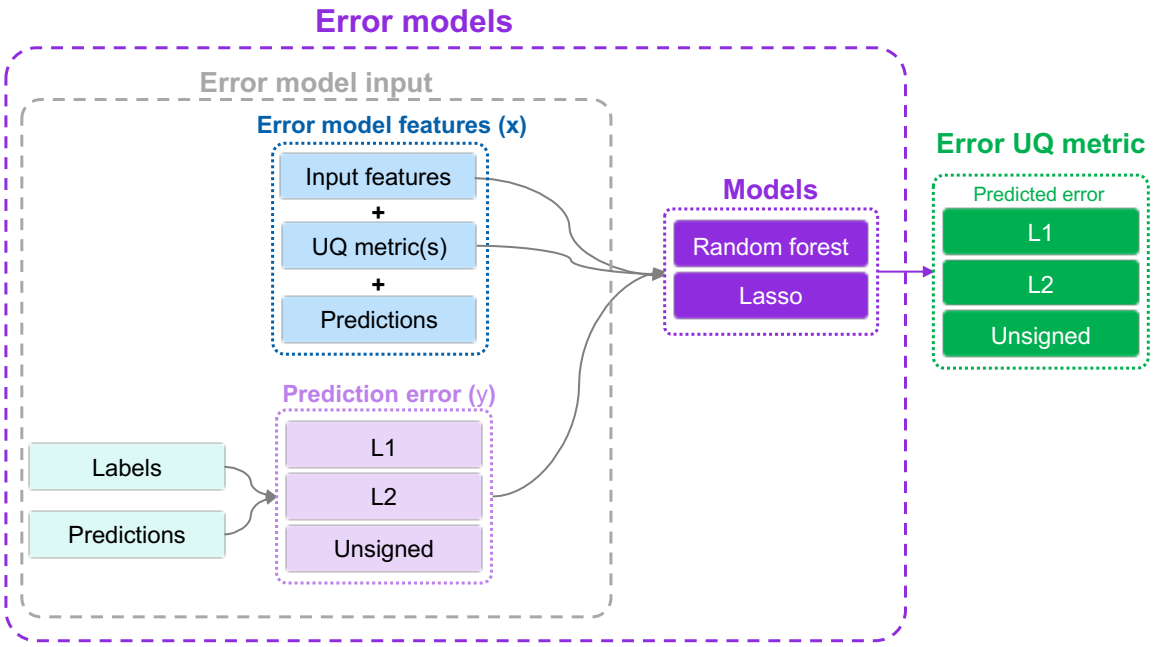

Figure 3. Workflow for error models generation. Error models are built with three different feature subsets: (i) original model prediction, UQ metrics, and input features provided by the user, (ii) original model prediction and UQ metrics, (iii) original model prediction and transformed UQ metrics. Prediction errors can be defined as L1, L2, or unsigned. Lasso or RF models are generated based on the training and calibration data.

2.5. Evaluation

The evaluation module indicates the quality of the uncertainty estimates and provides a recommendation for future usage. There are three evaluation types to cover different use cases:

- **Ranking-based evaluation**. It evaluates whether a UQ metric is a good indicator of the model prediction errors. The ranking of samples based on their actual prediction error is compared to the ranking based on the UQ metric. Example: Spearman's correlation coefficient.
- **Calibration-based evaluation**. It assesses whether the prediction intervals are well-calibrated, i.e. consistent with the underlying target distribution. Example: mean absolute calibration error (MACE)[37].

- **Proper scoring rules.** It focuses on the distributional prediction quality by assigning a scalar summary measure, where the maximum score is reached when the predicted distribution exactly matches the target one. Example: NLL[38,39].

Table S1 reports the complete list of implemented scores from each evaluation type and a brief description.

2.6. Output benchmarking results

UNIQUE's output consists of a collection of summary tables and plots that benchmark all UQ metrics. Each figure reports the results for a type of evaluation and highlights the best performing UQ metric. For the selection of the most promising UQ metric, bootstrapping is carried out on the test set and a distribution of evaluation scores is obtained. Wilcoxon ranked sum tests are conducted to assess whether the differences in evaluation scores are statistically significant across UQ metrics (pairwise comparisons) and Bonferroni correction is considered due to multiple testing[29]. Bootstrapping is applied with three evaluation metrics: Spearman's correlation coefficient (ranking-based evaluation), mean absolute calibration error (calibration-based evaluation), and NLL (proper scoring rules). The best UQ metric is defined as the one with the highest number of occurrences with significantly better distribution of evaluation scores than its counterparts. Hence, the best UQ metric could be different across evaluation types and scores. Additional visualizations can be obtained for additional insights into the most promising UQ metrics.

3. Application: LogD7.4 prediction

To highlight the application of UNIQUE, a practical example is presented in the context of molecular property predictions. A model was built to predict the lipophilicity of a molecule, which is an important parameter in drug discovery. A publicly available dataset from ChEMBL (ID: CHEMBL3301361)[40,41] was used, which consists of 4,200 compounds with measured LogD7.4 values (distribution coefficients between n-octanol and buffer at pH = 7.4). Different uncertainty estimates were calculated and benchmarked with the UNIQUE library.

3.1. Model building

Molecules were represented numerically using Morgan fingerprints (RDKit version 2023.09.1[42]), which encoded atom environments up to radius 3 and were mapped onto a vector of dimensionality 2048[42,43]. Compounds were randomly split into training (50%), calibration (30%), and test (20%) subsets. A random forest (RF)[36] regressor was generated using the training set (number of trees: 200; minimum of samples per split: 2, and minimum of samples in a leaf node: 2; *scikit-learn version v. 1.3*[44]), and predictions were obtained for the calibration and test sets. Model performance was estimated with the mean absolute error (MAE), root mean squared error (RMSE), and coefficient of determination (R^2). Calibration (and test) set performance was MAE = 0.66 (0.65), RMSE = 0.87 (0.87), and R^2 = 0.47 (0.45) respectively. The predictions from 200 trees were collected, and their prediction variances associated with each sample used as one of the UQ metrics.

3.2. Running UNIQUE

Table S2 shows an exemplary input file, which contains the (i) ChEMBL identifiers, (ii) LogD values as labels, (iii) RF predictions, (iv) the subset memberships that were used during RF model building, (v) Morgan fingerprints per each compound (feature input type), and (vi)

prediction variance across individual decision trees (model input type). **Table S3** reports the configuration file for UNIQUE's pipeline definition and **Table S4** reports the code snippet to run the pipeline. The outputs of *Pipeline* are the computed UQ metrics and their evaluation scores.

3.3. Calculation protocol

With the provided inputs, UNIQUE calculated three data-based UQ metrics using the input features (Morgan fingerprints): the distance to the k-NN in the training set using Manhattan and Euclidean distances, as well as the KDE with Gaussian kernel and Euclidean distance. Moreover, prediction variance was considered as model-based UQ metric. *Transformed* UQ metrics were also computed, namely sum of variance and distances and DiffNN. Specifically, Manhattan and Euclidean distances, and KDE from the training set were converted to variances using the calibrated NLL formalism (*vida supra*) and summed together with the RF's prediction variance. DiffNN was calculated using Morgan fingerprints and the two selected distance metrics to identify the k-NNs in the training set. Then, the absolute mean differences in variances and predicted values between the test compounds and its k-NNs were calculated to obtain the DiffNN scores (i.e. DiffNN variances and DiffNN predictions). Finally, three RF error models were built to predict L1-error with different subsets of features: (i) original RF model LogD7.4 prediction, UQ metrics, and Morgan fingerprints, (ii) original RF model LogD7.4 prediction and UQ metrics, (iii) original RF model LogD7.4 prediction and transformed UQ metrics. UQ metrics were evaluated as detailed above.

(a) Ranking-based evaluation

	UQ Method	Subset	AUC Difference: UQ vs. True Error	Spearman Correlation	Decreasing Coefficient	Increasing Coefficient	Performance Drop: High UQ vs. Low UQ (3-Bins)	Performance Drop: All vs. Low UQ (3-Bins)	Performance Drop: All vs. Low UQ (10-Bins)	Performance Drop: High UQ vs. Low UQ (10-Bins)
0	TanimotoDistance[fingerprints]	TEST	0.204	0.322	**4.452**	0.153	2.001	1.443	1.597	2.694
1	EnsembleVariance[variance]	TEST	0.194	0.345	4.229	0.086	2.040	1.540	1.824	2.605
2	Diff5NN[TanimotoDistance[fingerprints], EnsembleVariance[variance]]	TEST	0.256	0.217	4.240	0.960	1.526	1.185	1.171	1.712
3	Diff5NN[TanimotoDistance[fingerprints], predictions]	TEST	0.324	0.017	2.643	**1.966**	1.096	1.030	0.991	0.972
4	Dist2Var[TanimotoDistance[fingerprints]]	TEST	0.204	0.322	**4.452**	0.153	2.001	1.443	1.597	2.694
5	SumOfVariances[Dist2Var[TanimotoDistance[fingerprints]]]	TEST	0.186	0.360	4.406	0.036	2.130	1.533	1.822	2.919
6	UniqueRandomForestRegressor[fingerprints+UQmetrics+predictions](I1)	TEST	**0.185**	**0.375**	4.350	0.147	**2.214**	**1.616**	1.775	2.607
7	UniqueRandomForestRegressor[UQmetrics+predictions](I1)	TEST	0.189	0.356	4.421	0.083	2.126	1.547	**1.840**	2.952
8	UniqueRandomForestRegressor[transformedUQmetrics+predictions](I1)	TEST	0.187	0.365	4.411	0.162	2.199	1.559	1.831	**3.094**

(b) Calibration-based evaluation

	UQ Method	Subset	MACE	RMSCE
0	EnsembleVariance[variance]	TEST	0.145	0.180
1	Dist2Var[TanimotoDistance[fingerprints]]	TEST	0.138	0.173
2	SumOfVariances[Dist2Var[TanimotoDistance[fingerprints]]]	TEST	**0.072**	**0.088**
3	UniqueRandomForestRegressor[fingerprints+UQmetrics+predictions](I1)	TEST	0.164	0.202
4	UniqueRandomForestRegressor[UQmetrics+predictions](I1)	TEST	0.166	0.204
5	UniqueRandomForestRegressor[transformedUQmetrics+predictions](I1)	TEST	0.163	0.201

(c) Proper scoring rules evaluation

	UQ Method	Subset	NLL	CheckScore	CRPS	IntervalScore
0	EnsembleVariance[variance]	TEST	1.294	0.237	0.469	2.359
1	Dist2Var[TanimotoDistance[fingerprints]]	TEST	1.228	0.238	0.472	2.367
2	SumOfVariances[Dist2Var[TanimotoDistance[fingerprints]]]	TEST	1.308	0.247	0.488	2.520
3	UniqueRandomForestRegressor[fingerprints+UQmetrics+predictions](I1)	TEST	1.226	0.237	0.469	2.364
4	UniqueRandomForestRegressor[UQmetrics+predictions](I1)	TEST	1.231	0.237	0.469	2.357
5	UniqueRandomForestRegressor[transformedUQmetrics+predictions](I1)	TEST	**1.213**	**0.236**	**0.468**	**2.345**

Figure 4. Evaluation summary for the LogD7.4 dataset. Reported are the (a) ranking-based, (b) calibration-based, and (c) proper scoring rules evaluation summary tables. Multiple evaluation metrics or scores are reported for each UQ metric (UQ Method column). For each evaluation score, the UQ method with the best value (highest/lowest) is highlighted in bold. Finally, the best UQ metric is obtained with the Wilcoxon rank sum test on bootstrap samples and is highlighted in green. The best performing UQ metric is obtained for each evaluation type, using Spearman's correlation coefficient (ranking-based), MACE (calibration-based), and NLL (proper scoring rules).

3.4. UNIQUE's results

Figure 4a and **Figure 4c** show that one of the error models was highlighted as the best UQ method in both the ranking-based and proper scoring rules evaluations. Such error model was a RF trained on the input features (Morgan fingerprints), all the *base* UQ metrics, and the original RF model predictions. Specifically, the RF error model predicted the L1-error of the original RF model. This UQ metric achieved the highest Spearman's correlation coefficient (ρ=0.38) and the lowest NLL (NLL=1.21) in the bootstrap estimations. **Figure 4b** shows that the sum of the ensemble variance and Tanimoto distance to the training set was the best UQ method according to the calibration-based evaluation. In that case, model-based and data-based UQ metrics were summed, and converted to a variance using the calibrated NLL. Such strategy showed the lowest MACE (MACE=0.07) in the bootstrap estimations.

Figure S1a and **Figure S1b** show that the RF error model provided lower uncertainty estimates for compounds associated to lower errors, and vice versa. Test set predictions were binned into three categories (low, medium, high) according to the estimated uncertainty. Compounds with predicted L1-error lower than 0.39 resulted in an average MAE of 0.41, whereas compounds with L1-error predictions higher than 0.71 and lower than 1.63 showed an average MAE of 0.86. **Figure S1c** reports the MAE estimations at varying number of test compounds, which are added according of increasing or decreasing uncertainty estimate's values. The figure shows how test samples with higher uncertainty estimates are associated with larger errors, and vice versa. These visualizations focus on the ranking-based evaluation. **Figure S2** reports visualizations related to the calibration-based evaluation, namely the calibration curve and ordered prediction intervals[25], which can also be generated within the framework.

Overall, these results indicate that depending on the application, a different UQ metric could be preferred. A RF error model would be more successful at identifying compounds with high or low prediction confidence, and thus defining an applicability domain for the model. On the other hand, the sum of data-based and model-based UQ metrics would more accurately estimate prediction intervals (error bars), which might be more useful to build an acquisition function for active learning.

4. Conclusions

Finding accurate and robust uncertainty estimations in prediction models constitutes a principal focus in ML research. Different strategies to quantify uncertainty are available, but there is no method that can be generally applied and produces consistently accurate estimations of prediction uncertainty. Moreover, for many datasets, finding an appropriate UQ strategy constitutes a challenging task. Attempting to alleviate this issue, the UNIQUE library has been introduced as a framework for UQ benchmarking. This library extends some methods reported in previous libraries with new state-of-the-art UQ metrics, namely a combination of data- and model-based uncertainty estimates and error models (ML models to predict the original model error). Since there is no gold standard UQ method that is consistently superior and successful across modeling tasks and datasets, UNIQUE also facilitates the comparison of UQ strategies and scrutinizes their quality with several evaluation metrics that capture different aspects. This comprehensive benchmark allows for *ad-hoc* evaluation of the UQ metrics by ensuring application-specific prioritization of the score to optimize for. For instance, ranking-based evaluations might be more relevant for the removal of the *least confident* predictions to avoid decision-making based on those. On the other hand, well-calibrated uncertainty estimates might be required for acquisition functions in active learning efforts (assessed by calibration-based evaluation). Taken together, the UNIQUE framework enables an extensive comparison between different uncertainty estimates, standardizing the benchmark of new developments in the field of UQ for AI/ML.

Data and software availability

The presented framework is available in GitHub: https://github.com/Novartis/UNIQUE. The dataset used is publicly available in ChEMBL.

References

1. M Cartwright Hugh. *Machine Learning in Chemistry: The Impact of Artificial Intelligence*. (Royal Society of Chemistry, 2020).
2. Cios, K. J., Kurgan, L. A. & Reformat, M. Machine learning in the life sciences. *IEEE Engineering in Medicine and Biology Magazine* **26**, 14–16 (2007).
3. Dara, S., Dhamercherla, S., Jadav, S. S., Babu, C. M. & Ahsan, M. J. Machine Learning in Drug Discovery: A Review. *Artificial Intelligence Review 2021 55:3* **55**, 1947–1999 (2021).
4. Zhang, Y. Q. & Rajapakse, J. C. Machine Learning in Bioinformatics. *Machine Learning in Bioinformatics* 1–456 (2008) doi:10.1002/9780470397428.
5. Abdar, M. *et al.* A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion* **76**, 243–297 (2021).
6. Amodei, D. *et al.* Concrete Problems in AI Safety. *ArXiv* (2016).
7. Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence 2019 1:5* **1**, 206–215 (2019).
8. Guo, C., Pleiss, G., Sun, Y. & Weinberger, K. Q. On Calibration of Modern Neural Networks. in *Proceedings of the 34th International Conference on Machine Learning* 1321–1330 (PMLR, 2017).
9. Ovadia, Y. *et al.* Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *Adv Neural Inf Process Syst* **32**, (2019).
10. Krishnan, R. & Tickoo, O. Improving model calibration with accuracy versus uncertainty optimization. *Adv Neural Inf Process Syst* **33**, 18237–18248 (2020).
11. Nixon, J. *et al.* Measuring Calibration in Deep Learning. *ArXiv* 38–41 (2019).

12. Mathea, M., Klingspohn, W. & Baumann, K. Chemoinformatic Classification Methods and their Applicability Domain. *Mol Inform* **35**, 160–180 (2016).

13. Varnek, A. & Baskin, I. I. Chemoinformatics as a Theoretical Chemistry Discipline. *Mol Inform* **30**, 20–32 (2011).

14. Sutton, C. *et al.* Identifying domains of applicability of machine learning models for materials science. *Nature Communications 2020 11:1* **11**, 1–9 (2020).

15. MacKay, D. J. C. Bayesian methods for adaptive models. (California Institute of Technology , Pasadena, CA, United States, 1992). doi:10.7907/H3A1-WM07.

16. Khan, M. E. *et al.* Fast and Scalable Bayesian Deep Learning by Weight-Perturbation in Adam. 2611–2620 Preprint at https://proceedings.mlr.press/v80/khan18a.html (2018).

17. Osawa, K. *et al.* Practical Deep Learning with Bayesian Principles. *Adv Neural Inf Process Syst* **32**, (2019).

18. Wenzel, F. *et al.* How Good is the Bayes Posterior in Deep Neural Networks Really? 10248–10259 Preprint at https://proceedings.mlr.press/v119/wenzel20a.html (2020).

19. Bernardo, J. M. & Smith, A. F. M. Bayesian Theory. *Bayesian Theory* 1–595 (2008) doi:10.1002/9780470316870.

20. Shahriari, B., Swersky, K., Wang, Z., Adams, R. P. & De Freitas, N. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE* **104**, 148–175 (2016).

21. Neal, R. M. *Bayesian Learning for Neural Networks*. vol. 118 (Springer New York, New York, NY, 1996).

22. Detommaso, G. *et al.* Fortuna: A Library for Uncertainty Quantification in Deep Learning. *ArXiv* (2023).

23. Nado, Z. *et al.* Uncertainty Baselines: Benchmarks for Uncertainty & Robustness in Deep Learning. *ArXiv* (2021).

24. Ghosh, S. *et al.* Uncertainty Quantification 360: A Holistic Toolkit for Quantifying and Communicating the Uncertainty of AI. *ArXiv* (2021).

25. Chung, Y., Char, I., Guo, H., Schneider, J. & Neiswanger, W. Uncertainty Toolbox: an Open-Source Library for Assessing, Visualizing, and Improving Uncertainty Quantification. *ArXiv* (2021).

26. Taquet, V., Blot, V., Morzadec, T., Lacombe, L. & Brunel, N. MAPIE: an open-source library for distribution-free uncertainty quantification. (2022).

27. Cortés-Ciriano, I. & Bender, A. Concepts and Applications of Conformal Prediction in Computational Drug Discovery. in *Artificial Intelligence in Drug Discovery* 63–101 (2020). doi:10.1039/9781788016841-00063.

28. Tran, K. *et al.* Methods for comparing uncertainty quantifications for material property predictions. *Mach Learn Sci Technol* **1**, 025006 (2020).

29. Hirschfeld, L., Swanson, K., Yang, K., Barzilay, R. & Coley, C. W. Uncertainty Quantification Using Neural Networks for Molecular Property Prediction. *J Chem Inf Model* **60**, 3770–3780 (2020).

30. Tran, D. *et al.* Plex: Towards Reliability using Pretrained Large Model Extensions. *ArXiv* (2022).

31. Rasmussen, M. H., Duan, C., Kulik, H. J. & Jensen, J. H. Uncertain of uncertainties? A comparison of uncertainty quantification metrics for chemical data sets. *J Cheminform* **15**, 1–17 (2023).

32. Sheridan, R. P., Culberson, J. C., Joshi, E., Tudor, M. & Karnachi, P. Prediction Accuracy of Production ADMET Models as a Function of Version: Activity Cliffs Rule. *J Chem Inf Model* **62**, 3275–3280 (2022).

33. Lahlou, S. *et al.* DEUP: Direct Epistemic Uncertainty Prediction. *ArXiv* (2021).

34. Tavazza, F., Decost, B. & Choudhary, K. Uncertainty prediction for machine learning models of material properties. *ACS Omega* **6**, 32431–32440 (2021).

35. Tibshirani, R. Regression Shrinkage and Selection via the Lasso. *Source: Journal of the Royal Statistical Society. Series B (Methodological)* **58**, 267–288 (1996).

36. Breiman, L. Random forests. *Mach Learn* **45**, 5–32 (2001).

37. Kuleshov, V., Fenner, N. & Ermon, S. Accurate Uncertainties for Deep Learning Using Calibrated Regression. *35th International Conference on Machine Learning, ICML 2018* **6**, 4369–4377 (2018).

38. Gneiting, T. & Raftery, A. E. Strictly Proper Scoring Rules, Prediction, and Estimation. *J Am Stat Assoc* **102**, 359–378 (2007).

39. Ehm, W. & Gneiting, T. Local Proper Scoring Rules of Order Two. *The Annals of Statistics* **40**, 609 (2012).

40. Mark Wenlock and Nicholas Tomkinson. *Experimental in Vitro DMPK and Physicochemical Data on a Set of Publicly Disclosed Compounds - CHEMBL3301361*. (2015).

41. Gaulton, A. *et al.* ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Res* **40**, (2012).

42. RDKit: Open-source cheminformatics. https://www.rdkit.org/.

43. Rogers, D. & Hahn, M. Extended-connectivity fingerprints. *J Chem Inf Model* **50**, 742–754 (2010).

44. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).

Acknowledgements

The authors thank Grégori Gerebtzoff, Nikolas Fechner, and Nikolaus Stiefl for helpful scientific discussions. M.T.D.H. and G.S. thank the Translational Medicine Data Science Academy program for their fellowship at Novartis.

Author contributions

J.L. and R.R.P. conceived and designed the framework; J.L., R.R.P. M.T.D.H. and G.S. implemented the framework and wrote the manuscript; J.L., R.R.P., and N.S. supervised the study; all authors discussed the results and revised the manuscript.

Competing interests

The authors are/were employees and/or shareholders of Novartis AG.

Table of Contents