Supplemental information 1

Uncertainty-aware machine learning to predict non-cancer human toxicity for the global chemical market

Kerstin von Borries[a], Katie V Beckwith[b], Jonathan M Goodman[b], Weihsueh A Chiu[c], Olivier Jolliet[a] and Peter Fantke[d,e*]*

[a]Quantitative Sustainability Assessment, Department of Environmental and Resource Engineering, Technical University of Denmark, Bygningstorvet 115, 2800 Kgs. Lyngby, Denmark

[b]Centre for Molecular Informatics, Department of Chemistry, University of Cambridge, Lensfield Road, Cambridge CB2 1EW, United Kingdom

[c]Department of Veterinary Physiology and Pharmacology, Texas A&M University, 4466 TAMU, College Station, TX 77843-4466, United States of America

[d]substitute ApS, Graaspurvevej 55, 2400 Copenhagen, Denmark

[e]Department for Evolutionary Ecology and Environmental Toxicology, Goethe University, 60438 Frankfurt am Main, Germany

Contents

S1.1 Data preprocessing 2
S1.1a Data and SMILES curation 2
S1.1b Features 3
S1.2 Conventional ML model training without uncertainty quantification 4
S1.2a Conventional ML algorithms 4
S1.2b Cross-validation procedure 5
S1.2c Consensus model building 6
S1.2d Conventional ML performance results 7
S1.3 Uncertainty-aware ML model training 10
S1.3a Uncertainty frameworks 10
S1.3b Cross-validation procedure 11
S1.3c Comparison with conventional ML models 13
S1.3d UAM performance and calibration results 14
S1.4 Model application 24
References 28

S1.1 Data preprocessing

S1.1a Data and SMILES curation

We trained our models using the large dataset of probabilistic PODs derived by Aurisano et al. (2023) through a statistical approach based on curated in vivo effect data from the US EPA Toxicity Value Database (ToxValDB). The dataset contained PODs for reproductive/developmental and general non-cancer effects via chronic oral exposure for more than 10,000 chemicals that act as surrogates for PODs derived through authoritative assessment. The separate endpoints allow considering the differing severity between lifelong reproductive/developmental impairments and general non-cancer effects that typically manifest later in life, when used for life cycle impact assessment[1]. We complemented the dataset with 26 general non-cancer PODs for dioxin-like chemicals that were not yet part of the dataset by leveraging toxic equivalency factors (TEFs) reported by DeVito et al. (2024)[2], thereby improving the representation of chemicals with low general non-cancer PODs (signifying high toxicity). For every chemical in the dataset, chemical structure information in the form of SMILES was collected from the US EPA Comptox Chemicals Dashboard[3] and complemented with results from the open chemistry database PubChem[4] using the python package PubChemPy[5] and the PubChem Identifier exchange service.[6] Structures were identified primarily via their CAS registration numbers and secondarily via their chemical names or abbreviations. All SMILES were canonicalized. Following harmonization and the related evident duplicate removal, CAS numbers with multiple valid SMILES (n=133) were inspected to select the most appropriate structure based on information provided in the data set. In the case of ambiguous or missing information, the simplest / non-stereometric form was chosen. Inversely, multiple CAS with the same associated SMILES were inspected selecting the most representative record for the given structure. This affected 150 structures with reported POD_{rd} and 175 structures with reported POD_{nc}. This approach yielded unique CAS-SMILES combinations covering 4832 chemicals in the reproductive/developmental dataset and 5370 chemicals in the general non-cancer dataset. Subsequently, both datasets were filtered to keep PODs that were obtained from a minimum of 4 underlying measured data points to reduce data-related uncertainty as Aurisano et al. derived surrogate PODs using the 25^{th} percentile. The resulting data sets consisted of 3402 chemicals with POD_{rd} and 2717 chemicals with POD_{nc} ("expanded dataset"). After standardizing chemical structures using the protocol by Mansouri et al. (2018) – which includes the removal of inorganic and organometallic compounds, large molecules (>1000 g/mol), and the stripping of salts and solvents – 2357 unique chemicals with POD_{rd} and 1945 with POD_{nc} were available for model training and validation ("standardized dataset"). In addition, the "expanded data" set was used to assess the potential for achieving robust uncertainty quantification across diverse chemicals beyond the typically applicability domain of QSAR models. All PODs were converted from mg/kg-d to log10-transformed units of mol/kg-d prior to training representing the rational link between molecules and their biological activity,[7] though we note that such conversion have been shown to have little influence in other modeling studies.[8] Prediction results were presented in mg/kg-d for easier interpretation.

Figure S1-1 shows a histogram of the expanded dataset of reproductive/developmental and general non-cancer PODs, highlighting the three most toxic compounds.

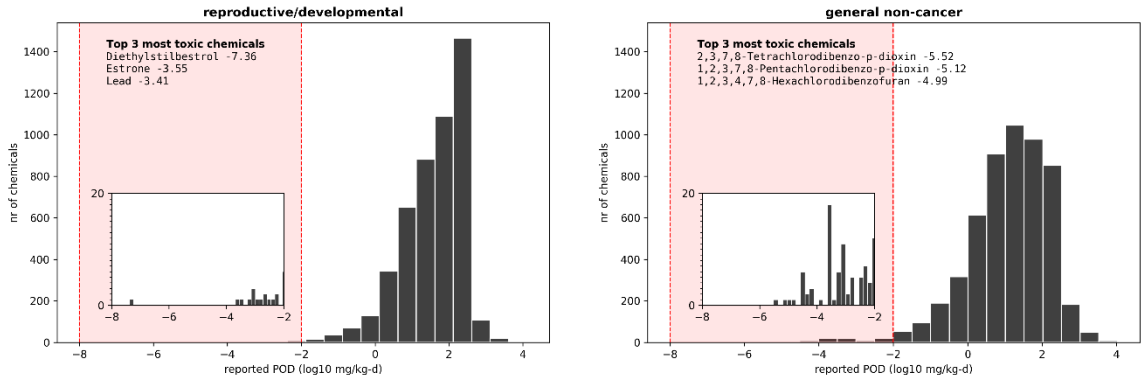

Figure S1-1. Histogram of reported reproductive/developmental (left) and general non-cancer (right) PODs highlighting toxic outliers with POD < 0.01 mg/kg-d representing 0.4% and 1.5% of training chemicals with reported POD_{rd} and POD_{nc}, respectively.

S1.1b Features

We considered four types of molecular descriptors that numerically describe chemical structures to be used as training features for the ML models. We chose two types of fingerprints (MACCS, MORGAN) and two types of continuous descriptors (RDKIT, CDDD) (see Table S1-1 for more details). The RDKIT descriptors as well as the MACCS and MORGAN fingerprints were calculated with *RDKit* v2022.9.5.[9] MORGAN fingerprints were calculated with a length of 512 bits and a radius of 2, following initial tests that showed that increasing fingerprint length (up to 4096) and radius (up to 4) did not help improve modelling performance. The CDDD embedding was calculated with the *CDDD* python package provided by Winter et al. (2019).[10]

While MACCS and RDKIT descriptors are knowledge-based descriptors that encode structural fragments or properties of chemical structures, MORGAN fingerprints and CDDD embedding provide an unweighted representation of the full chemical structure. We removed highly correlated features (Pearson correlation coefficient > 0.9) from the knowledge-based descriptors but not from the structural representations as these provide a valid representation only when used together. In addition, we applied a Yeo-Johnson transformation to the numeric RDKIT descriptors to achieve Gaussian-like distributions and scale all RDKIT descriptors to zero mean and unit variance. To avoid data leakage in cross-validation, all feature processing steps are learned from training sets and applied accordingly to the respective test sets within each cross-validation fold (see sections S1.2b and S1.3b for details).

Table S1-1. Brief overview of the four types of molecular descriptors used in this study including the number of features for each type and their primary references.

Type	**Number**	**Description**	**Reference**
MACCS fingerprint	166	binary, structural keys of pre-defined sub-structures and fragments	Durant et al. (2002)[11]
MORGAN fingerprint	adjustable	binary, circular extended-connectivity fingerprint (ECFP) covering all possible structural fragments in each atom neighborhood (up to a defined radius), hashed to a defined length (bits)	Rogers & Hahn (2010)[12]
RDKit descriptors	208	continuous/distinct 1D and 2D descriptors covering fragment counts, topological, structural, and physicochemical parameters	RDKit (2022)[9]
CDDD embedding	512	Continuous auto-encoded representation of chemical structure obtained from the latent space	Winter et al. (2019)[10]

		of a SMILES translation model based on a recurrent neural network (RNN)	

S1.2 Conventional ML model training without uncertainty quantification

We performed a systematic evaluation of conventional ML algorithms without uncertainty quantification to benchmark performance of best-practice ML model development protocols training each algorithm with the four types of molecular descriptors

S1.2a Conventional ML algorithms

We selected six algorithms that cover common algorithm families following different functional principles for finding an optimal fit to a given training data set: Multiple linear regression (MLR), k nearest neighbor (KNN), support vector machine (SVM), random forest (RF), extreme gradient boosting machine (XGB) and a simple feed forward neural network (NN) (see Table S1-2 for details). MLR, KNN, SVM and RF were implemented using *scikit-learn* v1.2.2. The SVM was trained with a radial basis kernel function (RBF) for highly flexible, non-linear behavior. The XGB was implemented using *xgboost* v2.0.0. The NN was implemented with *Tensorflow* v2.13.0 with one hidden layer with the same number of neurons as the input layer, following initial tests that showed no significant performance improvements with wider (up to twice the number of input features) and deeper (up to 3 hidden layers) architectures. The hidden layer was activated with a rectified linear unit (ReLU) to enable non-linear behavior. The network was trained for 500 epochs with the Adam optimizer using an optimizable learning rate and a learning rate scheduler that reduced the learning rate by a factor of 0.5 if the validation loss did not reduce by at least $1e^{-5}$ mean squared error (MSE) for 10 epochs (patience). In addition, early stopping was implemented if the validation loss did not reduce by at least $1e^{-5}$ MSE in 20 epochs restoring the weights from the best previous epoch. Learning rate scheduler and early stopping callbacks operated based on a 20% validation split. For regularization, we optimized drop-out at the input layer and weight decay during the nested cross-validation (see Table S1-3 for details). The *Tensorflow* model was made compatible with scikit-learn functionalities using wrappers by *scikeras* v0.11.0.

Table S1-2. Brief overview of six types of machine learning algorithms used in this study including their functional principle and their primary references.

Family	**Algorithm**	**Abbr.**	**Principle**	**References**
Multiple linear regression	Elastic net	MLR	Ordinary least squares regression with a tuneable mix of L1 (Lasso) and L2 (Ridge) regularization	Zou & Hastie (2004)
Instance-based learning	K nearest neighbor regression	KNN	memory-based, non-parametric algorithm that predicts a new data point based on the (weighted) average of the most similar instances (nearest neighbors) in the training data set	Cover & Hart (1967)[13]
Support vector machine	Support vector regression	SVM	Vapnik-Chervonenkis (VC) theory-based algorithm that fits to the training data considering an error-insensitive tube around the fitted function with different kernel options for non-linear behaviour	Boser et al. (1992)[14]
Ensemble (bagged)	Random forest	RF	ensemble of parallel decision trees trained on bootstrapped data samples that predicts a new data point by aggregating predictions from underlying decision trees	Breiman (2001)[15]

Ensemble (boosted)	Extreme Gradient Boosting	XGB	ensemble of sequential decision trees each trained on the residual errors of its predecessor thereby tweaking weak predictions of preceding decision trees (stagewise addition model)	Chen & Guestrin (2016)[16]
Neural network	Feed forward neural network	NN	(simple) artificial neural network, consisting of an input layer, one or more hidden layers and an output layer with optional non-linear activations	Rosenblatt (1958)[17]

S1.2b Cross-validation procedure

For every descriptor-algorithm combination, model performance was assessed through a nested cross-validation procedure with five inner folds to optimize relevant hyperparameters of the respective algorithm and ten outer folds to robustly estimate the generalization error of the model. Hyperparameters and appropriate value ranges were selected focusing on settings controlling the learning behavior and the complexity of each algorithm (see Table S1-3 for details). The space of hyperparameter combinations was searched using Bayesian optimization with testing (10 * number of hyperparameters) samples in each fold. Bayesian optimization was implemented with *scikit-optimize* v0.9.0. The best-performing model across the five inner loops was retrained on all available training data in the inner loop and its external prediction performance tested in the outer loop. Chemicals were assigned to each fold using target value-based stratified sampling based on order of magnitude of PODs, where the upper and lower 1%-ile of the POD value range were aggregated into one stratification class each to ensure a sufficient number of samples per class. This approach ensured a similar distribution of POD values across all folds and compensated impacts from data imbalance. The nested cross-validation was repeated ten times to assess the variation in the generalization error across different training data samples (10 outer folds) with different random initial states (10 seeds). The generalization error was assessed using mean absolute error (MAE), median absolute error (MdAE), root mean squared error (RMSE) and the (adjusted) coefficient of determination (R2, $R2_{adj}$) as error metrics (see Table S1-4 for detailed definitions). Model performances are compared to a mean predictor model as baseline.

Table S1-3. Overview of hyperparameter optimization including a brief description and the optimization ranges for each algorithm. Parameters that were varied within a logarithmic uniform distribution are denoted with (log).

Algorithm	**Hyperparameters**	**Description**
MLR	alpha: 0.0...1.0 l1_ratio: 0.0...1.0	regularization strength mix between L1 and L2 regularization
KNN	n_neighbors: 3…20 weights: uniform, distance metric: minkowski, jaccard	nr of nearest neighbors uniform or distance-weighted average metric for distance calculation
SVM	gamma: 1e-3...10 (log) C: 0.1...10 (log) epsilon: 0.0...1.0	RBF kernel coefficient regularization strength width of no penalty tube
RF	n_estimators: 100…1000 max_depth: 2…20 min_samples_leaf: 1…100 max_samples: 0.5…1.0 max_features: 0.5…1.0 min_impurity_decrease: 1e-3...0.1 (log)	number of trees in the forest maximum depth of tree minimum samples required in each leaf node number of samples to train base estimators number of features to consider at each split minimum loss reduction required to split
XGB	n_estimators: 100…1000	number of boosting rounds

	learning_rate: 0.03...0.3 (log) max_depth: 2...20 min_child_weight: 0...10 subsample: 0.5...1.0 colsample_bytree: 0.5...1.0 gamma: 1e-3...0.1 (log)	step size shrinkage for weight update maximum depth of tree minimum sum of instance weight in child training sample ratio per boosting iteration column sample ratio for creating each tree minimum loss reduction required to split
NN	batch_size: 32, 64, 128 dropout_rate: 0.0…0.5 learning_rate: 1e-4…0.01 (log) weight_decay: 1e-5…0.1 (log)	number of samples per weight update ratio of randomly ignored input nodes step size shrinkage for weight update ratio of weight shrinkage per iteration

Table S1-4. Overview of error metrics used to assess generalization performance of different machine learning methods.

Metric	**Abbr.**	**Formula**	**Definitions**
Mean absolute error	MAE	$\frac{1}{n}\sum_{i=1}^{n}\lvert y_i - \hat{y_i}\rvert$	n: number of data points y_i: measured value for ith data point $\hat{y}_i$: predicted value for i^{th} data point $\bar{y}$: mean measured value across data points
Median absolute error	MdAE	$median\ \lvert y_i - \hat{y_i}\rvert$	
Root mean squared error	RMSE	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2}$	
Coefficient of determination	R2	$1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$	
Adjusted coefficient of determination	$R2_{adj}$	$1 - \frac{(1-R2)\cdot(n-1)}{n-p-1}$	n: number of data points p: number of features

S1.2c Consensus model building

To establish a single, high-performing benchmark for the uncertainty-aware models, we created a consensus model by taking the average across the subset of conventional ML models with the highest statistically significant prediction performances ($p < 0.05$). To estimate the generalization errors and test the statistical significance of the differences in model performance we followed Nadeau and Bengio (2000) to perform pairwise correlated t-tests. The procedure for a single pair-wise correlated t-test between models A and B following *K*-fold cross-validation with *J* repetitions is briefly outlined below:

1. Compute the prediction error (loss) for models A and B for each data point *i* and repetition *j* (we used the squared loss).

   $z_{i,j}^{A} = \left(\hat{y}_{i,j}^{A} - y_{i,j}\right)^2 \quad | \quad z_{i,j}^{B} = \left(\hat{y}_{i,j}^{B} - y_{i,j}\right)^2$

2. Estimate the mean difference $\hat{d}$ in the generalization error from the mean loss difference between models A and B across all J repetitions.

   $d_j = \frac{1}{n_j}\sum_{i=1}^{n_j} z_{i,j}^{A} - z_{i,j}^{B}$

$\hat{d} = \frac{1}{J} \sum_{j=1}^{J} d_j$

3. Estimate the variance in the difference in the generalization error between models A and B accounting for the correlation induced due to overlapping training sets among the crossvalidation folds.

$\hat{\sigma}^2 = \frac{1}{J-1} \sum_{j=1}^{J} (d_j - \hat{d})^2$

$\tilde{\sigma}^2 = \left(\frac{1}{J} + \frac{\rho}{1-\rho}\right) \cdot \hat{\sigma}^2 \quad \text{where } \rho = \frac{1}{K}$

4. Compute the p-value for the null hypothesis that models A and B perform identically from the cumulative student's t-distribution function.

$p = 2 \cdot cdf(-|\hat{t}| \mid \nu = J - 1, \mu = 0, \sigma = 1) \quad \text{with } \hat{t} = \frac{\hat{d}}{\tilde{\sigma}}$

Consensus model predictions were obtained by averaging across all models with high prediction performance that were not statistically distinguishable from the model with the lowest mean RMSE ($p<0.05$). Based on the results from the statistical tests, the models comprising the consensus models are given in **Table S1-5**.

Table S1-5. Overview of conventional ML models building the consensus model when trained on the standardized data set for reproductive/developmental and general non-cancer PODs.

POD endpoint	**Models building consensus model** (algorithm-descriptor combination)
Reproductive/developmental toxicity	SVM-CDDD XGB-RDKIT
General non-cancer toxicity	RF-MACCS SVM-CDDD SVM-MACCS KNN-MACCS XGB-MACCS XGB-RDKIT

S1.2d Conventional ML performance results

This section presents results comparing the prediction performance of the conventional ML algorithms in combination with different molecular descriptors. Figure S1-2 to Figure S1-3 compare the mean prediction performance and its variance across repeated cross-validation folds for the selected metrics for POD_{rd} and POD_{nc}, respectively. The best performance was generally achieved with knowledge-based descriptors (RDKIT, MACCS), while SVM also achieved competitive performance using the embedding-based CDDD. Figure S1-4 shows the prediction performance of the obtained consensus model predictions. The highly flexible algorithms SVM, XGB and RF generally outperformed KNN, MLR and our simple NN. All models outperform the mean predictor model (represented by the dotted line) across all metrics, except for RMSE in the POD_{rd} predictions, where all models exhibit RMSE values exceeding the baseline on some folds. Another exception is the NN model trained with RDKIT descriptors to predict POD_{rd} which showed negative R2 values on some folds.

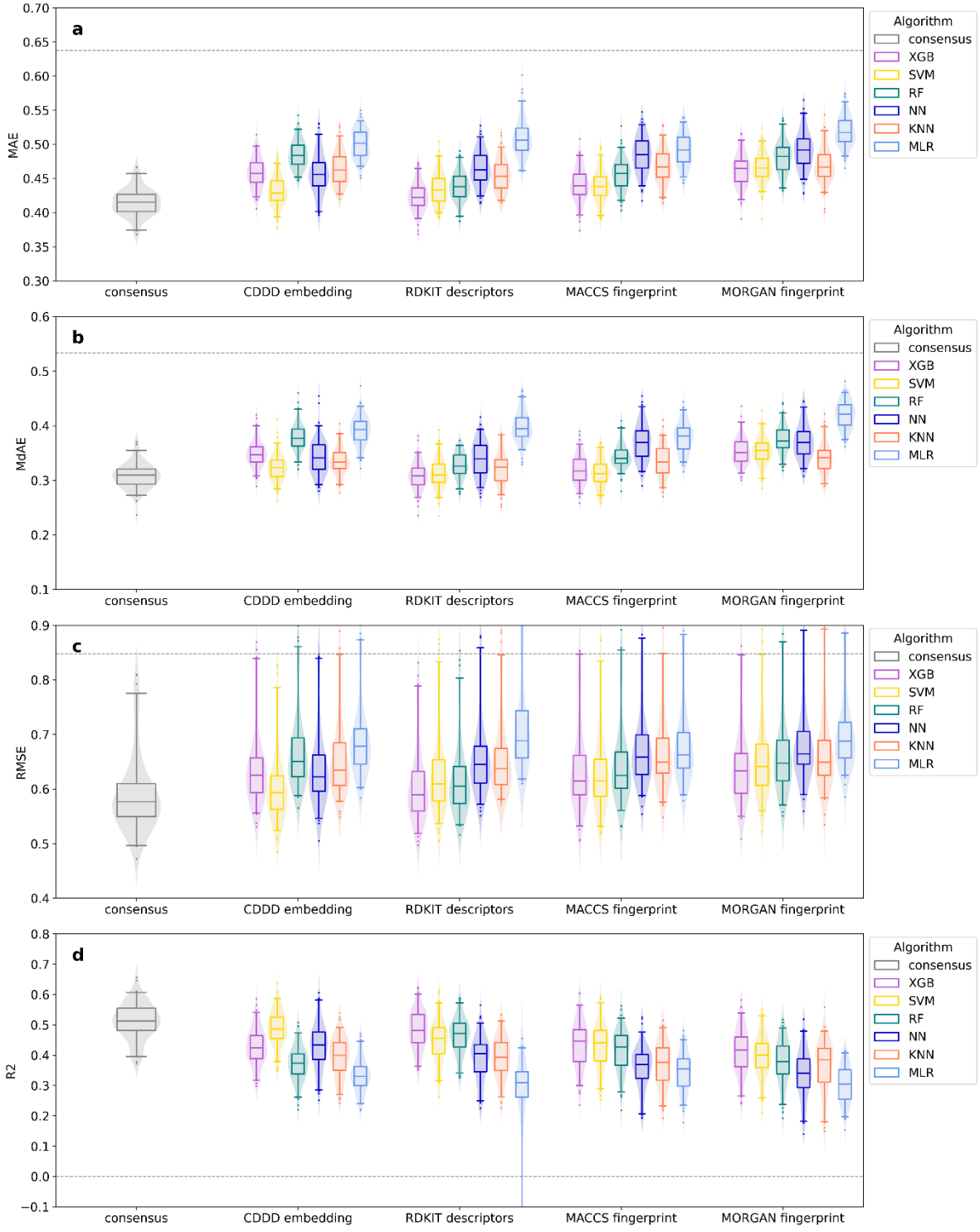

Figure S1-2. Performance comparison of conventional ML models and the consensus model for predicting reproductive-developmental PODs across crossvalidation folds for four metrics (a) mean absolute error (MAE), (b) median absolute error (MdAE), (c) root mean squared error (RMSE), and (d) coefficient of determination (R2). The dotted line represents the performance of the mean predictor model. A lower MAE, MdAE and RMSE indicates higher performance, while a higher R2 indicates higher performance.

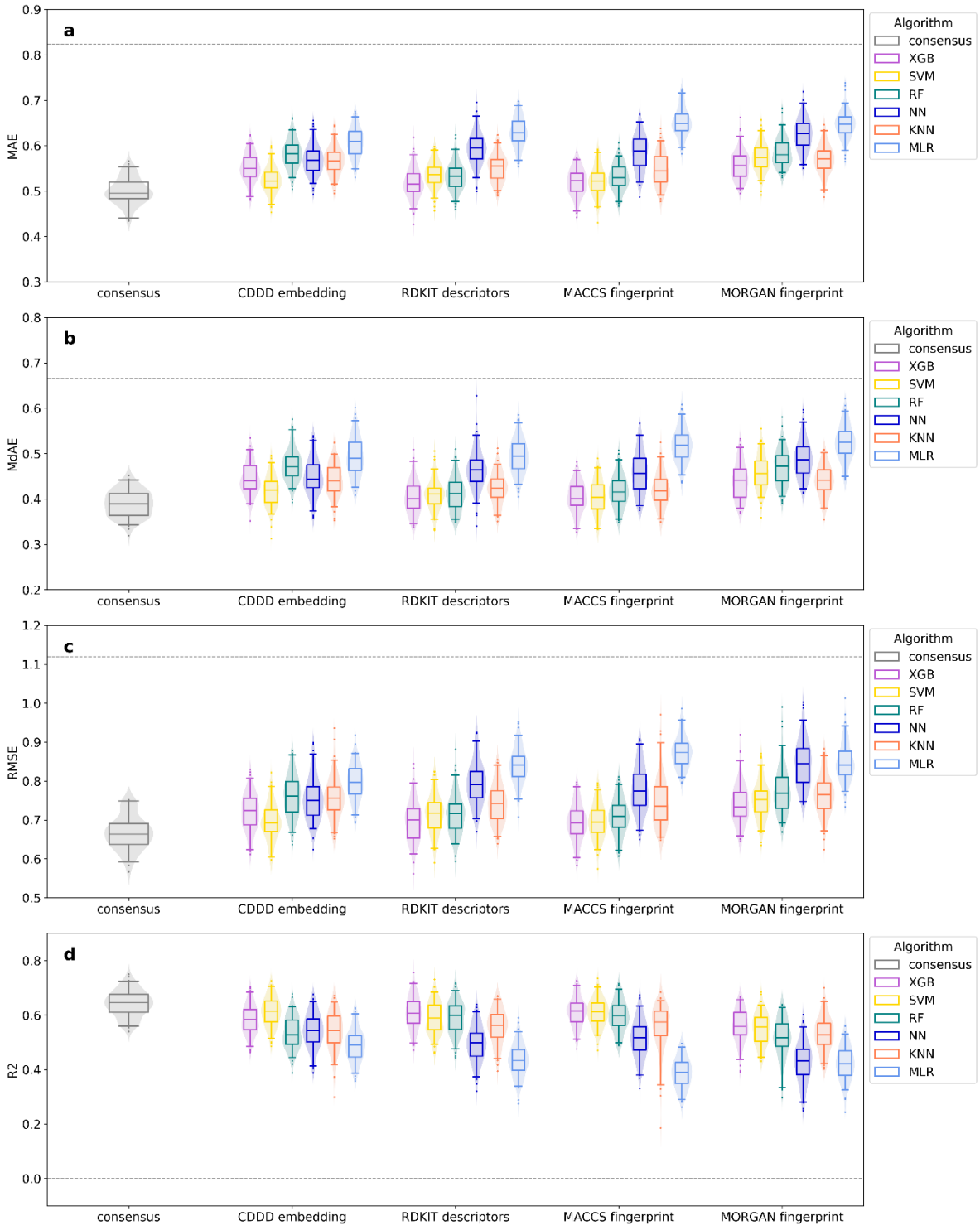

Figure S1-3. Performance comparison of conventional ML models and the consensus model for predicting general non-cancer PODs across crossvalidation folds for four metrics (a) mean absolute error (MAE), (b) median absolute error (MdAE), (c) root mean squared error (RMSE), and (d) coefficient of determination (R2). The dotted line represents the performance of the mean predictor

model. A lower MAE, MdAE and RMSE indicates higher performance, while a higher R2 indicates higher performance.

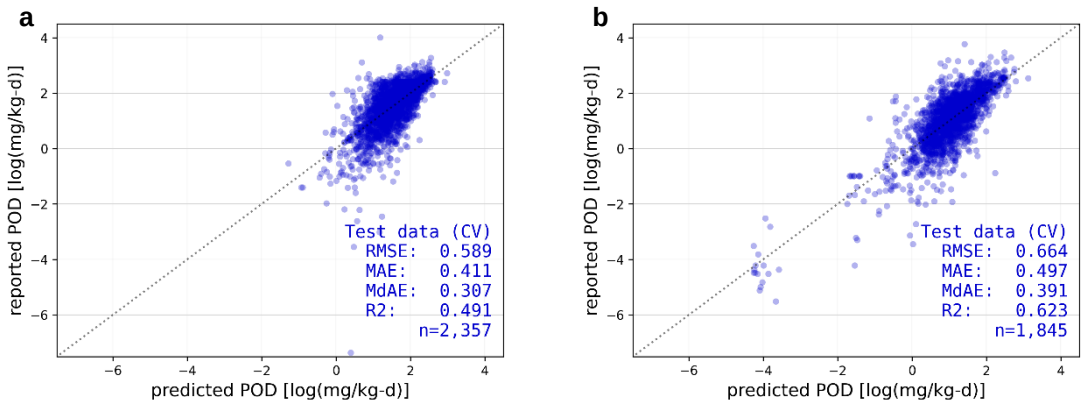

Figure S1-4. Prediction performance of the consensus models for (a) reproductive developmental and (b) general non-cancer PODs.

S1.3 Uncertainty-aware ML model training

S1.3a Uncertainty frameworks

Common approaches to convey model uncertainty often cover only one type of uncertainty. A commonly reported metric describing prediction uncertainty is the average prediction error. When obtained through cross-validation, the prediction error provides a robust estimate of a model's generalization error on unseen data. This metric is mainly concerned with aleatoric uncertainty. In addition, as a summary metric it does not inform about potential variability in uncertainty across predictions, called heteroscedasticity. Some machine learning methods allow capturing data-related heteroscedasticity, e.g. quantile regression-based approaches, providing prediction intervals based on empirically estimated quantiles. Other machine learning methods provide an uncertainty estimate with each point prediction capturing epistemic uncertainty, for example ensemble-based approaches that provide the variance across point predictions produced by a large set of predictors. Taken alone, these uncertainty estimates tend to be overconfident as they fail to adequately account for either epistemic or aleatoric uncertainty. In this study we compared two distinct paradigms within machine learning for quantifying uncertainty, conformal prediction (CP) and Bayesian neural networks (BNNs).

Conformal prediction

CP is rooted in hypothesis testing following classical, frequentist statistics.[18] It builds confidence intervals around point estimates based on a desired confidence level thereby controlling the fraction of erroneous predictions with a statistically valid probability. CP offers a model-agnostic and non-parametric approach that can be combined with any algorithm and provides confidence intervals without assuming a specific uncertainty distribution. However, not all CP-based methods provide full, heteroscedastic uncertainty quantification. In conformal regression, various conformity score definitions are available, each with different coverage guarantees and computational costs. Most commonly used conformity scores do not adapt well to heteroscedastic data, resulting in constant or minimally varying confidence intervals across prediction points. In response, Romano et al. (2019) introduced conformalized quantile regression (CQR), which effectively adapts to varying degrees of aleatoric uncertainty.[19] However, CQR falls short in distinguishing between aleatoric and epistemic uncertainty leading (among other things) to unreliable coverage of out-of-distribution data. Addressing this, Rossellini et al. (2023) proposed uncertainty-aware conformalized quantile

regression (UACQR), refining CQR to effectively address aleatoric and epistemic uncertainty.[20] They included two alternative approaches to estimating epistemic uncertainty – a scaling-based approaches that scales the upper and lower bounds by an explicit estimate of the quantile's standard deviations (UACQR-S) and a percentile-based approach where updated upper and lower bounds are obtained as percentiles from an ensemble of quantile estimates (UACQR-P). In this study, we present results obtained with UACQR-S due to better performance on the calibration metrics. Rossellini et al. implemented their method with four core algorithms: a linear quantile regression, a light gradient boosting machine with quantile loss, a quantile regression forest (qRF) and a quantile regression neural network with two hidden layers. In this study we used qRF as the model algorithm based on our conventional ML model comparison that showed RF among the high-performing algorithms and due to its more comprehensive implementation. We implemented UACQR using the python code provided by Rossellini et al. with a small modification to include median predictions in the model outputs alongside lower and upper bounds. The UACQR model was trained with a 95% confidence level, providing the 2.5^{th} and 97.5^{th} percentiles alongside the median as model outputs. By fitting the inverse cumulative distribution function of a skewed normal distribution to these three percent points, we could express the prediction uncertainty as a standard deviation when necessary.

Bayesian neural network

BNNs apply Bayesian principles to model uncertainty and are the main representative for probabilistic modelling approaches.[18] BNNs move beyond deterministic outputs by attaching probability distributions to each model parameter (biases and weights) and returning a probability distribution for the output prediction itself. This allows them to effectively capture both aleatoric and epistemic uncertainty. We trained our BNN with variational inference implemented in *TensorFlow Probability* (TFP) using Bayes-by-backprop and negative log-likelihood loss.[21] We defined our BNN with one hidden layer with ReLU activation and an output layer with two nodes assigned to the mean and standard deviation of a Gaussian distribution. A softplus function with scaling factor is used for the standard deviation to ensure positivity with computational stability. Following Blundell et al. (2015)[22], the weights and biases of the hidden and output layers were defined by a variational posterior distribution that approximates the true posterior with a multivariate normal distribution. The prior distribution of the weights and biases were defined as the mixture of two zero-mean normal distributions with respectively small and large variances that allow concentrating parameters close to zero without restricting learning flexibility. Our BNN was trained with the Adam optimizer with a learning rate of 0.002 for 1000 epochs with callbacks for early stopping and learning rate scheduling as defined for the base NN. Probabilistic model parameters are inherently regularized, rendering additional regularization methods such as drop-out and weight decay unnecessary in BNNs.[22] Prediction results were assessed based on random samples of 500 draws from the predicted output distributions.

S1.3b Cross-validation procedure

We set up a cross-validation procedure to robustly estimate the prediction performance and uncertainty calibration for each of the two uncertainty modelling approaches. The cross-validation consisted of ten folds with the random states initialized with a seed of 42. Chemicals were assigned to each fold using target value-based stratified sampling to compensate impacts from data imbalance. Due to their unique structures UACQR and TFP models were not compatible with conventional hyperparameter optimization tools. Based on experience with equivalent conventional ML models, we assigned the qRF in UACQR a maximum depth of 15, considering maximum half of all features and requiring a minimum impurity decrease of 0.001 at each split, thus preventing excessive overfitting. The BNN models were assigned a batch size of 32 and a learning rate of 0.002. The drop-out and weight decay parameters were omitted as explained above. The lack of hyperparameter optimization is expected to affect BNNs more than RF-based CP models, as RFs consistently

perform well without extensive tuning. NNs, however, are highly sensitive to hyperparameters, as shown in conventional ML optimization, where their settings varied significantly across folds and features. Despite this, the performance trend between CP and BNN UAM models aligned with that of conventional RF and NN models. While using hyperparameters based on experience with conventional ML models trained with different subsets of the same data poses a slight risk of data leakage, the impact is likely minimal: RFs are less affected by hyperparameter choices, and BNNs are more likely to lose performance due to insufficient tuning rather than gain from unintended information. The overall prediction performance was assessed using mean absolute error (MAE), median absolute error (MdAE), root mean squared error (RMSE) and the coefficient of determination (R2) (see Table S1-4 for details).

The quality of the uncertainty quantification was assessed considering three aspects of calibration: confidence-based, error-based, and distance-based calibration. We defined confidence- and error-based calibration curves following Yang & Li (2023)[23]. The confidence-based calibration curve compares the expected confidence level and the observed fraction of measured data points falling within the predicted confidence interval. It can be summarized by the expected calibration error (ECE) which describes the mean absolute deviation between expected and observed fractions (see Table S1-6). For the CP models, the confidence level is a parameter defined for the construction of the model and thus required training a new model for every confidence level. To limit computational cost, we obtained the confidence-based calibration points of the CP models based on a single 20% test set. For the BNN models, credible intervals (the Bayesian equivalent to frequentist confidence intervals) were obtained for defined probabilities (equivalent to confidence levels) from the corresponding percentiles of the cross-validated output distribution samples. The error-based calibration curve compares the consistency between the mean observed prediction error and the mean predicted uncertainty across batches of increasing prediction uncertainty. This requires expressing the predicted uncertainty as a standard deviation. For BNN the standard deviation was obtained directly from the model output, while for CP it was obtained by fitting the inverse cumulative distribution function of a skewed normal distribution to the median and CI bounds. To form the batches, the n testing data points are sorted by the predicted uncertainty and divided into B batches containing $\frac{n}{B}$ data points each. The number of batches was set to the entropy-based optimal number of batches obtained as $\sqrt{n}$ [24]. The error-based calibration can be summarized by the expected normalized calibration error (ENCE) defined by Levi et al (2022)[25] which describes the mean absolute deviation between observed error and predicted uncertainty across batches (see Table S1-6). As ENCE has been shown to be sensitive towards the chosen number of batches [24] we assessed the change in ENCE as a function of number of batches ranging from 5 to 200 (see Figure S1-5). Inspired by the out-of-distribution analysis by Yin et al (2023)[26], we defined a distance-based calibration curve to investigate the uncertainty calibration for chemicals that are structurally different from training chemicals. It assesses the consistency between mean structural dissimilarity and mean prediction uncertainty across batches of increasing prediction uncertainty, applying the same binning approach as for error-based calibration. The structural dissimilarity was described by the average Jaccard distance of each test chemical from its five nearest neighbors among training chemicals based on Morgan fingerprints (1024 bits, radius 2). The strength of correlation in all calibration curves was indicated by Pearson and Spearman correlation coefficients (see Table S1-6 for details).

Table S1-6. Overview of metrics used to assess calibration quality of the uncertainty modelling approaches.

Metric	Abbr.	Formula	Definitions

Expected calibration error	ECE	$\frac{1}{A}\sum_{a=1}^{A}\lvert CL_a - f_a\rvert$	A: number of confidence levels CL_a: a^{th} confidence level f_a: fraction of measured points falling into the CL_a confidence interval
Expected normalized calibration error	ENCE	$\frac{1}{B}\sum_{b=1}^{B}\frac{\lvert RMSE_b - RMU_b\rvert}{RMU_b}$ with $RMU_b = \sqrt{\frac{1}{N_b}\sum_{n_b=1}^{N_b}\sigma_{n_b}^2}$	B: number of batches $RMSE_i$: RMSE in bin b RMU_i: RMU in bin b N_b: number of data points in bin b σ_{nb}: predicted uncertainty as standard deviation on data point n in batch b
Pearson correlation coefficient	Pearson	$\frac{cov(X,Y)}{\sigma_X\sigma_Y}$	cov: covariance σ_x: standard deviation of X σ_y: standard deviation of Y
Spearman correlation coefficient	Spearman	$\frac{cov(R[X],R[Y])}{\sigma_{R[X]}\sigma_{R[Y]}}$	R[·]: rank variable of · cov: covariance $\sigma_{R[\cdot]}$: standard deviation of rank variable of ·

S1.3c Comparison with conventional ML models

To compare the performance of the UAMs to the conventional ML models, we estimated the generalization errors with confidence intervals and assessed the statistical significance of the differences in model performance. We applied a simpler procedure compared to section S1.2c based on K-fold crossvalidation results without repetition to compare the performance for a specific training set.

To determine the confidence interval of the generalization error for a given model:

1. Compute the prediction error (loss) for a given model for each of *n* data points *i* (we used the squared loss).

   $z_i = (\hat{y}_i - y_i)^2$

2. Estimate the generalization error $\hat{z}$ and its variance $\tilde{\sigma}^2$.

   $\hat{z} = \frac{1}{n}\sum z_i \quad | \quad \tilde{\sigma}^2 = \frac{1}{n(n-1)}\sum_{i=1}^{n}(z_i - \hat{z})^2$

3. Determine the $1-\alpha$ confidence interval of the generalization error from the cumulative student's t-distribution function.

   $z_L = cdf^{-1}\left(\frac{\alpha}{2}\middle|\, \nu = n-1, \hat{z}, \tilde{\sigma}\right) \quad | \quad z_u = cdf^{-1}\left(1-\frac{\alpha}{2}\middle|\, \nu = n-1, \hat{z}, \tilde{\sigma}\right)$

To assess the difference in performance between two models A and B and the statistical significance:

1. Compute the difference in prediction error (loss) between models A and B.

   $d_i = z_i^A - z_i^B$

2. Estimate the mean difference $\hat{d}$ and its variance $\tilde{\sigma}^2$.

   $\hat{d} = \frac{1}{n}\sum d_i \quad | \quad \tilde{\sigma}^2 = \frac{1}{n(n-1)}\sum_{i=1}^{n}\left(d_i - \hat{d}\right)^2$

3. Compute the p-value for the null hypothesis that models A and B perform identically from the cumulative student's t-distribution function.

$$p = 2 \cdot cdf\left(-\left|\hat{d}\right| \mid \nu = n - 1, \mu = 0, \sigma = \tilde{\sigma}\right)$$

S1.3d UAM performance and calibration results

Figure S1-5 shows the trend in ENCE as a function of the selected number of batches for the CP models trained with RDKIT descriptors.

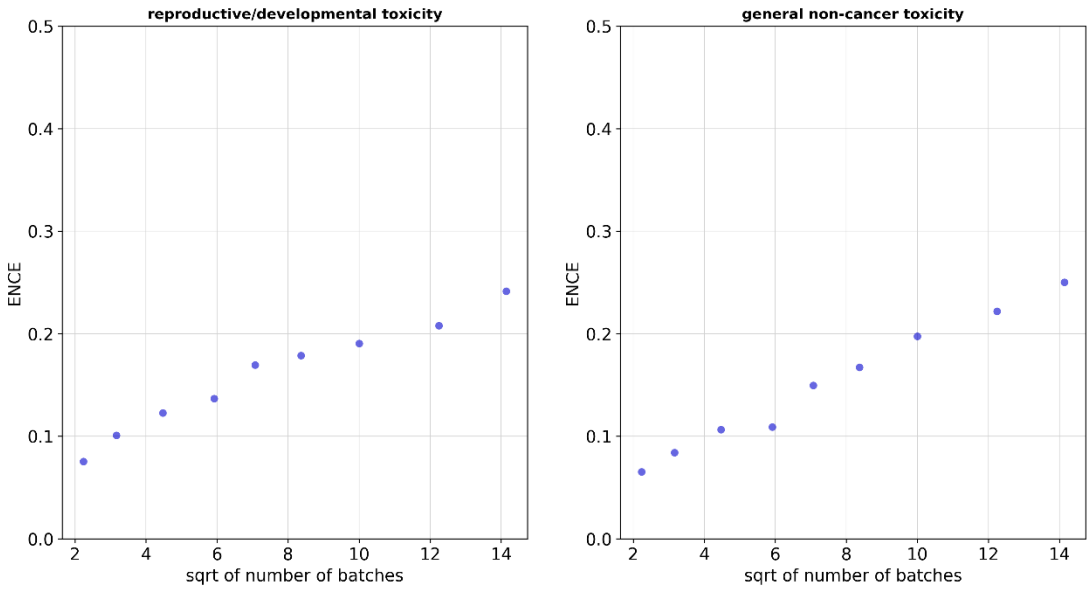

Figure S1-5. ENCE as a function of the square root number of batches for the CP models for (A) reproductive/developmental and (B) general non-cancer PODs.

Figure S1-6 shows the prediction performance and uncertainty calibration results for the BNN models trained with RDKIT descriptors. While both modeling approaches proved well-calibrated for uncertainty quantification along the confidence-based, error-based and distance-based calibration curves, the BNN models had slightly lower prediction performance and provided more conservative estimates of uncertainty compared to CP models.

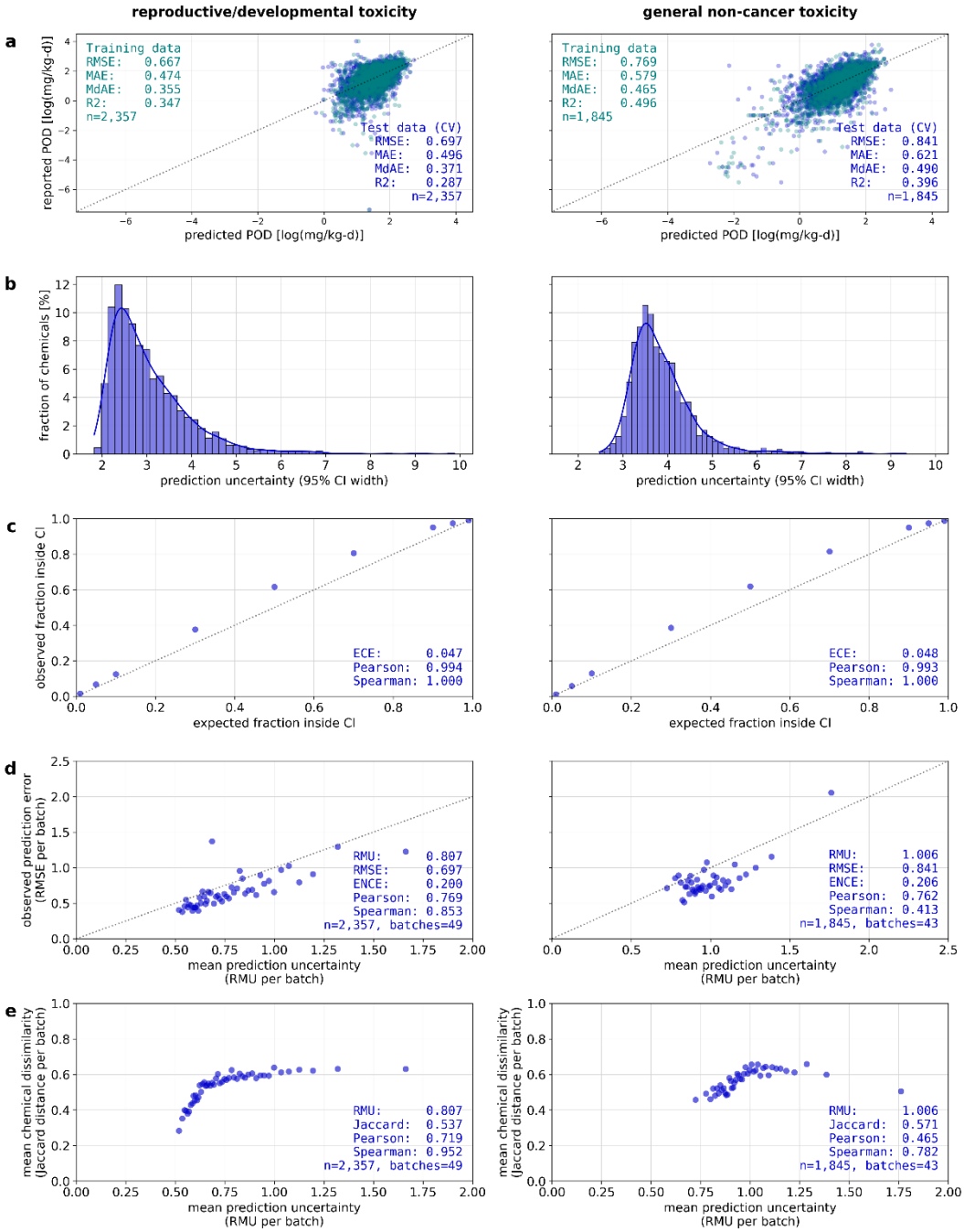

Figure S1-6. Prediction performance and uncertainty calibration of the BNN models trained with RDKIT descriptors. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based calibration curve **(d)** Error-based calibration curve **(e)** Distance-based calibration curve **Abbreviations:** ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

Figure S1-7 to Figure S1-12 show the prediction performance and uncertainty calibration results for the CP and BNN models trained with MACCS keys, MORGAN fingerprints and CDDD embeddings.

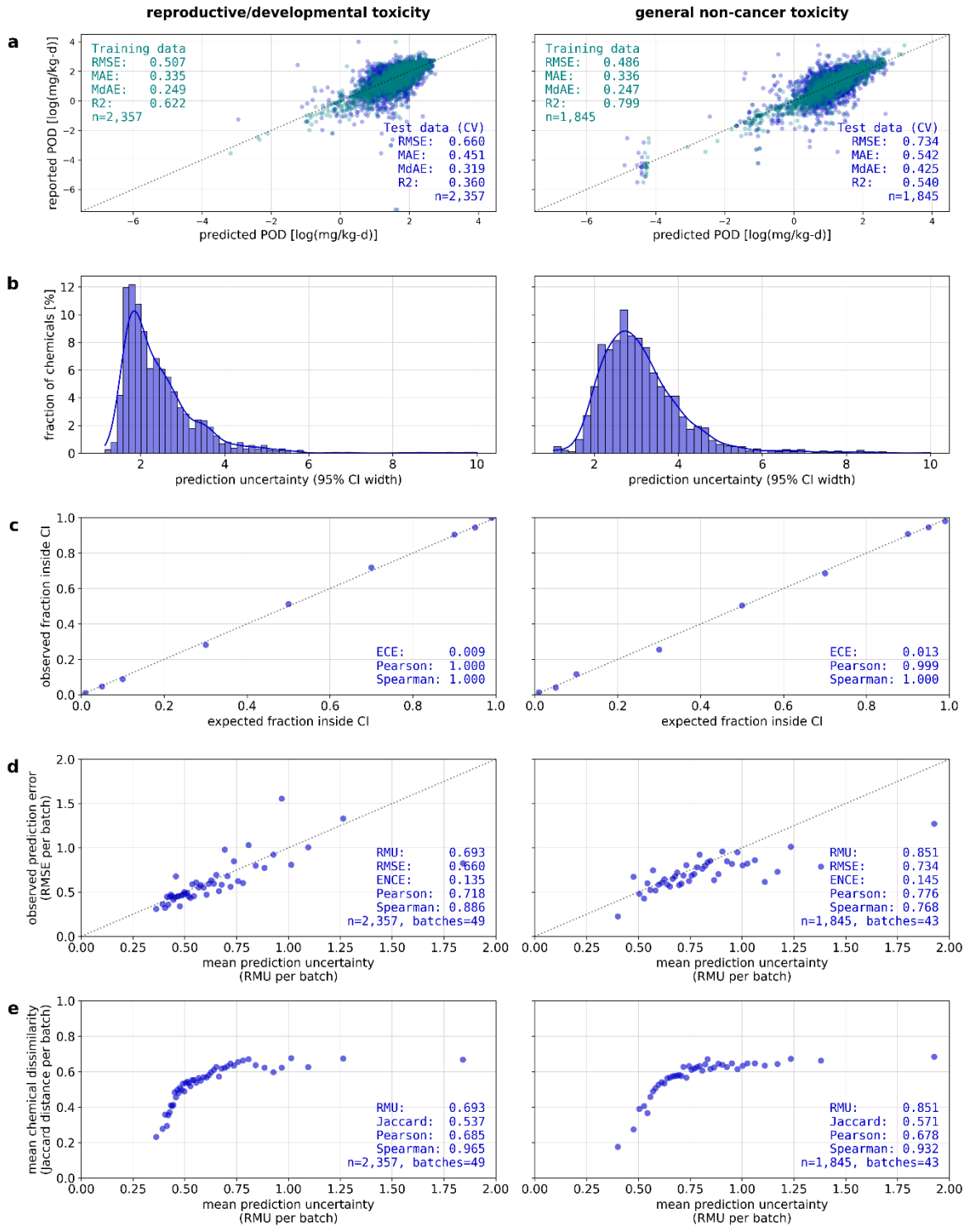

Figure S1-7. Prediction performance and uncertainty calibration of the CP models trained with MACCS keys. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based calibration curve **(d)** Error-based calibration curve **(e)** Distance-based calibration curve **Abbreviations:** ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of

determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

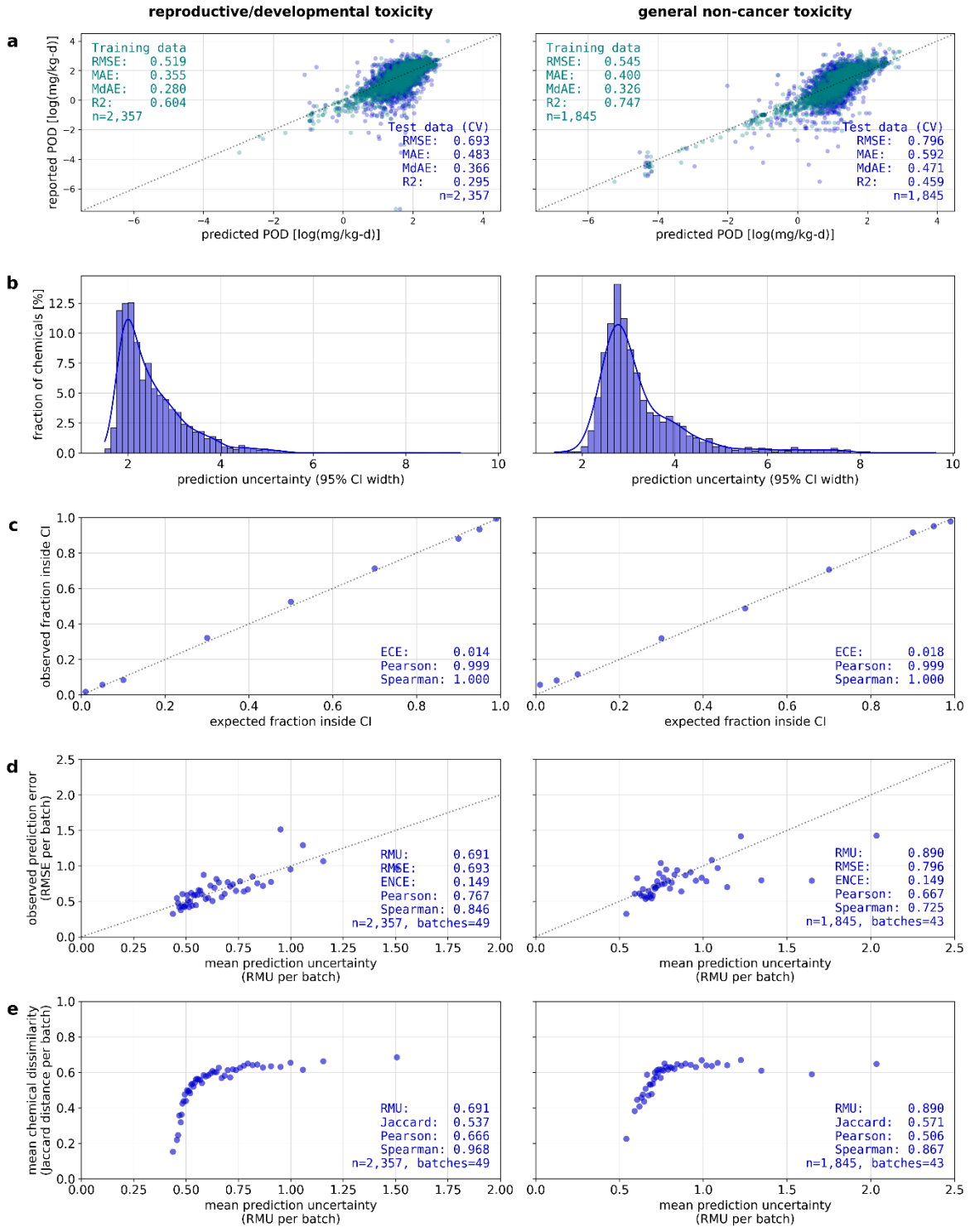

Figure S1-8. Prediction performance and uncertainty calibration of the CP models trained with MORGAN fingerprints. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based calibration curve **(d)** Error-based calibration curve **(e)** Distance-based calibration curve **Abbreviations:** ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient,

POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

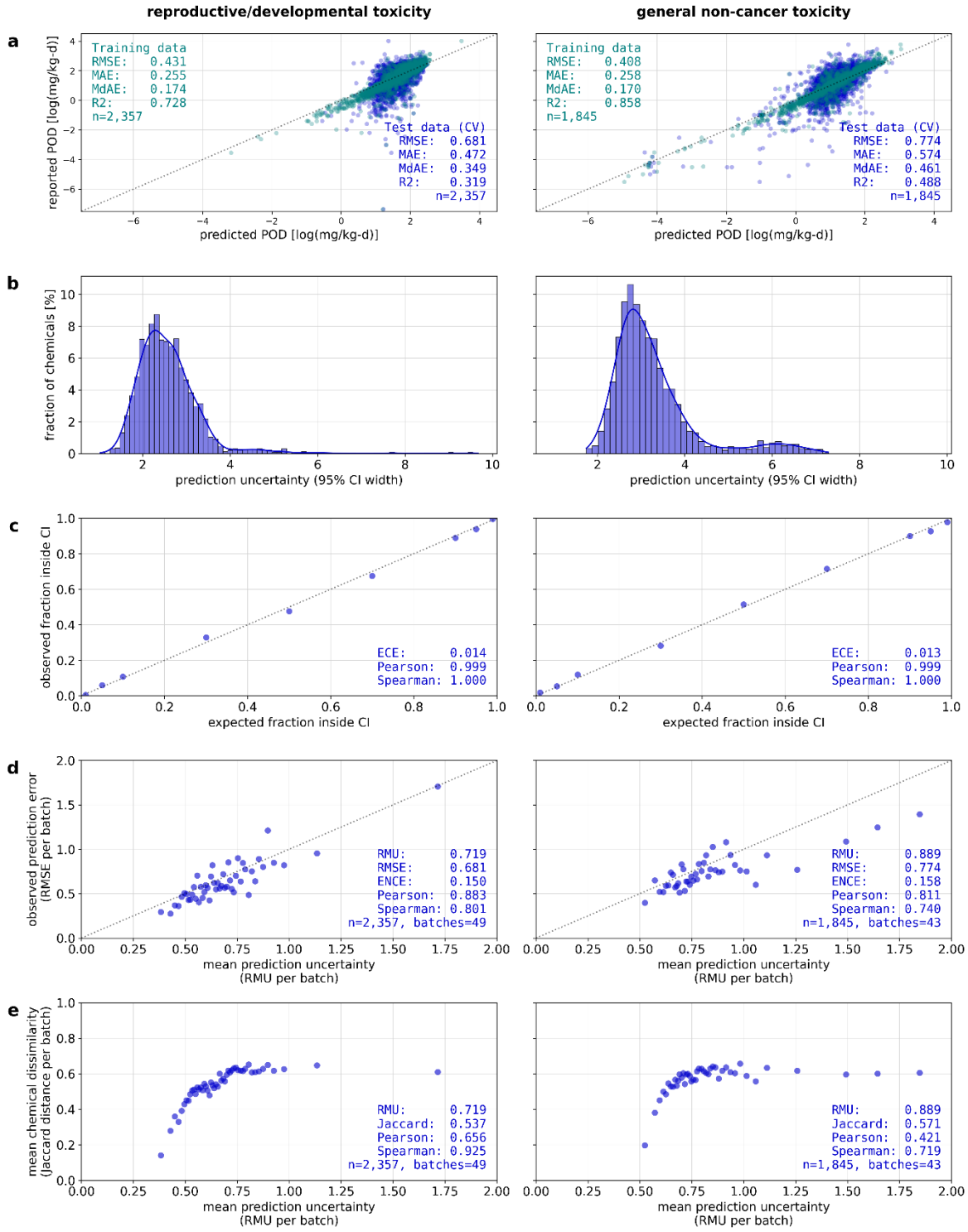

Figure S1-9. Prediction performance and uncertainty calibration of the CP models trained with CDDD embeddings. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based calibration curve **(d)** Error-based calibration curve **(e)** Distance-based calibration curve **Abbreviations:** ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error,

MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

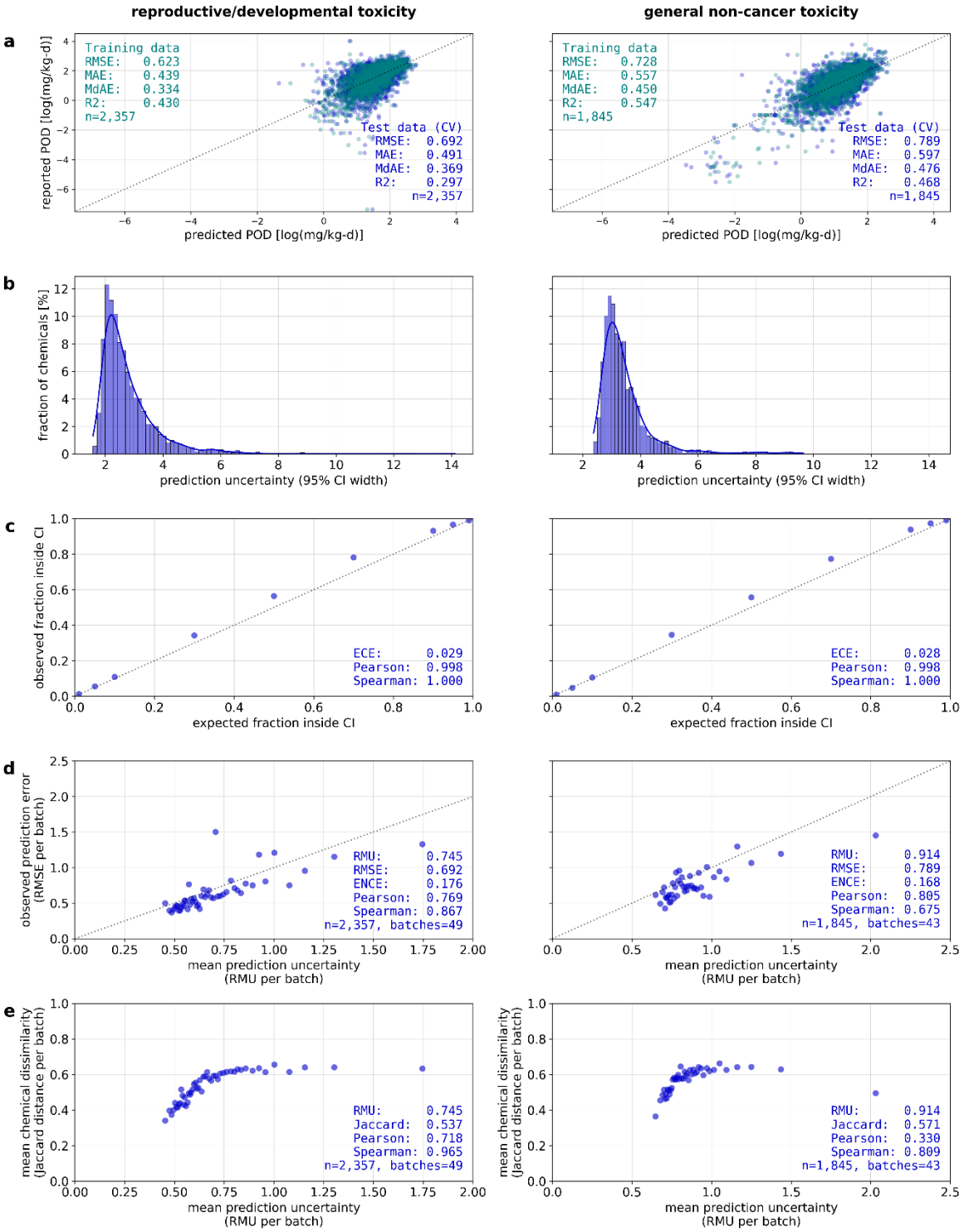

Figure S1-10. Prediction performance and uncertainty calibration of the BNN models trained with MACCS keys. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based calibration curve **(d)** Error-based calibration curve **(e)** Distance-based calibration curve **Abbreviations:** ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-

validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

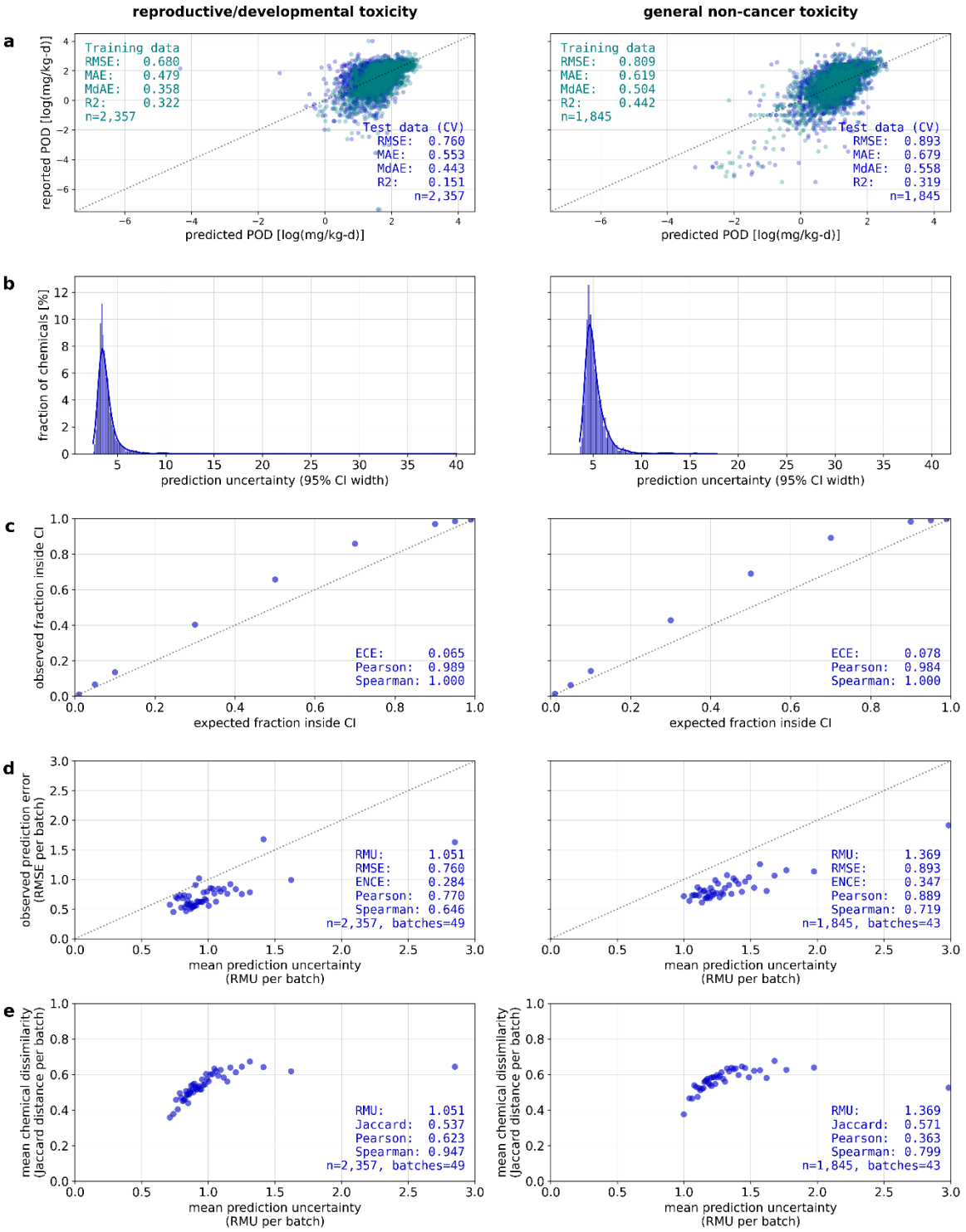

Figure S1-11. Prediction performance and uncertainty calibration of the BNN models trained with MORGAN fingerprints. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based calibration curve **(d)** Error-based calibration curve **(e)** Distance-based calibration curve **Abbreviations:**

ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

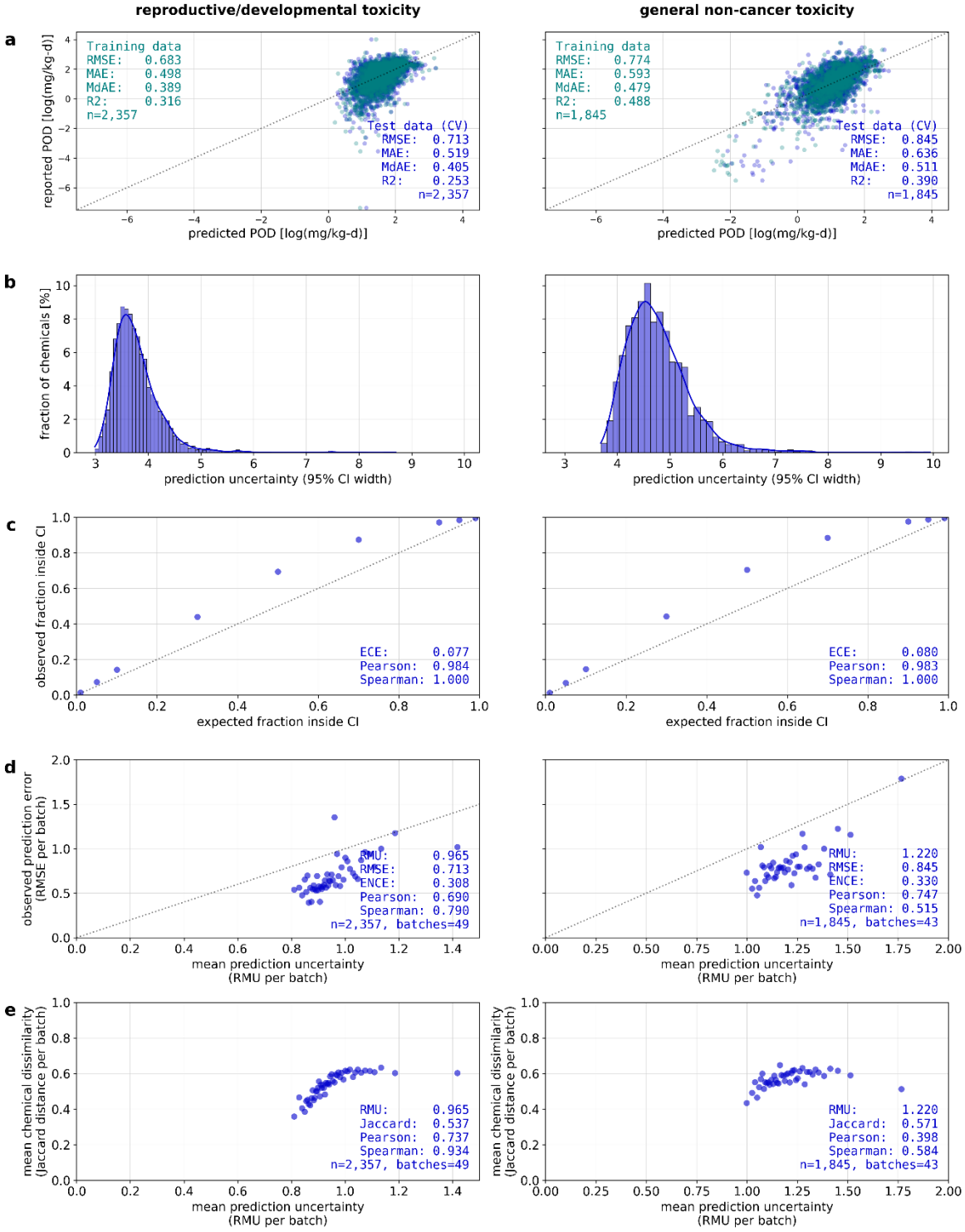

Figure S1-12. Prediction performance and uncertainty calibration of the BNN models trained with CDDD embeddings. **(a)** Prediction performance on training data set (green) and cross-validated test data set (blue). **(b)** Distribution of prediction uncertainty (95% CI width) **(c)** Confidence-based

calibration curve **(d)** Error-based calibration curve **Abbreviations:** ECE=expected calibration error, ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, R2adj=adjusted coefficient of determination, Spearman=Spearman correlation coefficient, std dev=standard deviation.

Figure S1-13 shows the prediction performance and uncertainty calibration results for predicting POD_{rd} for the non-standardized chemical subset.

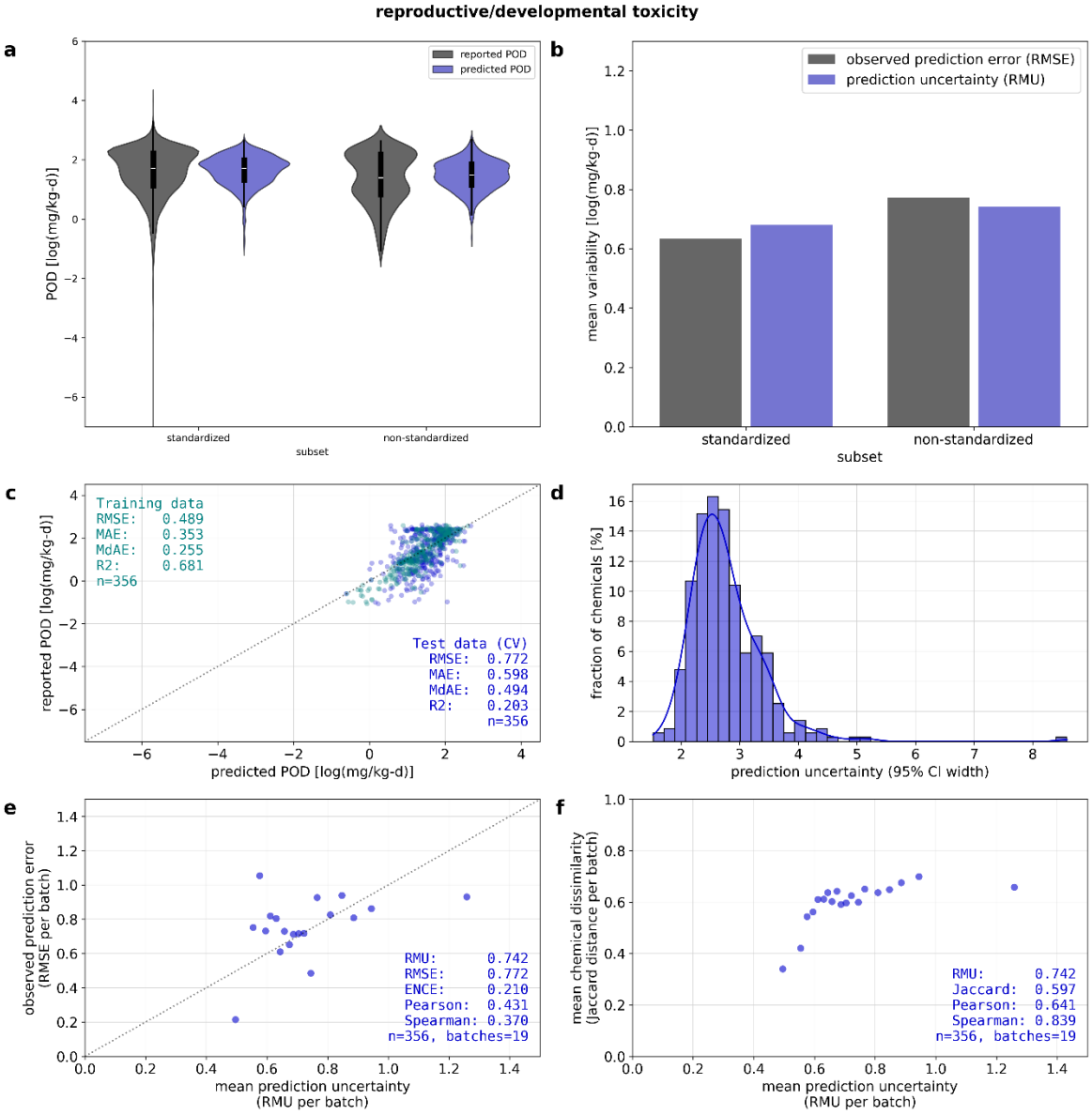

Figure S1-13. Prediction performance and uncertainty calibration for a non-standardized subset of chemicals predicted by CP models trained with an expanded dataset for POD_{rd}. **(a)** Distribution of reported and predicted PODs for standardized and non-standardized chemical subsets. **(b)** The mean prediction uncertainty (RMU) and observed prediction errors (RMSE) for standardized and non-standardized chemical subsets. **(c)** Prediction performance on training data set (green) and cross-validated test data set (blue) **(d)** Distribution of prediction uncertainty (95% CI width) **(e)** Confidence-

based calibration curve **(f)** Error-based calibration curve **Abbreviations:** ENCE=expected normalized calibration error, CI=confidence interval, CV = cross-validation, Jaccard=mean Jaccard distances from 5 nearest neighbors, MAE=mean absolute error, MdAE=median absolute error, n=number of chemicals, Pearson=Pearson correlation coefficient, POD=point of departure, RMSE=root mean squared error, RMU=root mean uncertainty, R2=coefficient of determination, Spearman=Spearman correlation coefficient

S1.4 Model application

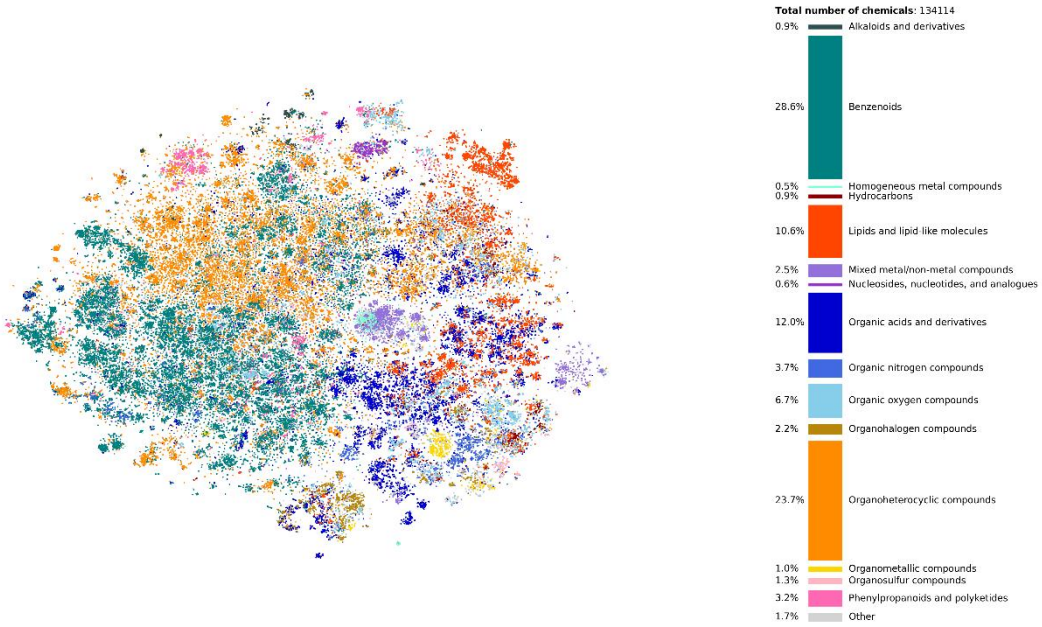

Figure S1-14. Two-dimensional chemical space embedding of 134,114 marketed chemicals obtained through t-SNE clustering colored by the 15 most frequent ClassyFire superclasses following the work of von Borries et al. (2023)

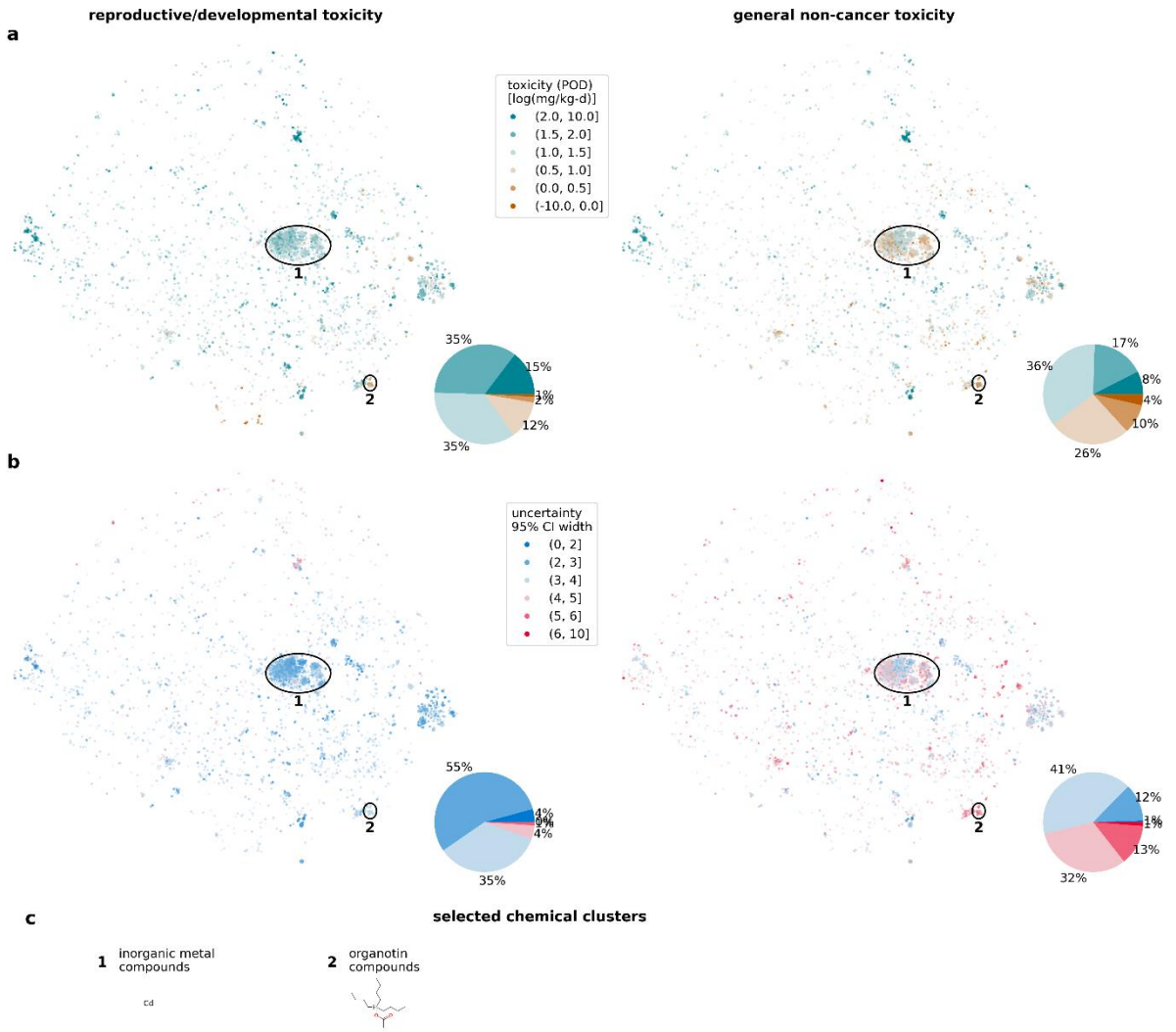

Figure S1-15. 8,054 non-standardized marketed chemicals colored by (A) the reproductive/developmental (left) and general non-cancer toxicity (right) given by the predicted log10-scale POD and (B) the associated prediction uncertainty given by the log10-scale width of the 95% CI visualized in a two-dimensional chemical space map of 134,114 marketed chemicals with (C) illustrative annotations of high toxicity and high uncertainty hotspots. Pie charts illustrate the percentage of marketed chemicals falling into the respective toxicity and uncertainty levels. **Abbreviations:** CI=confidence interval, POD=point of departure

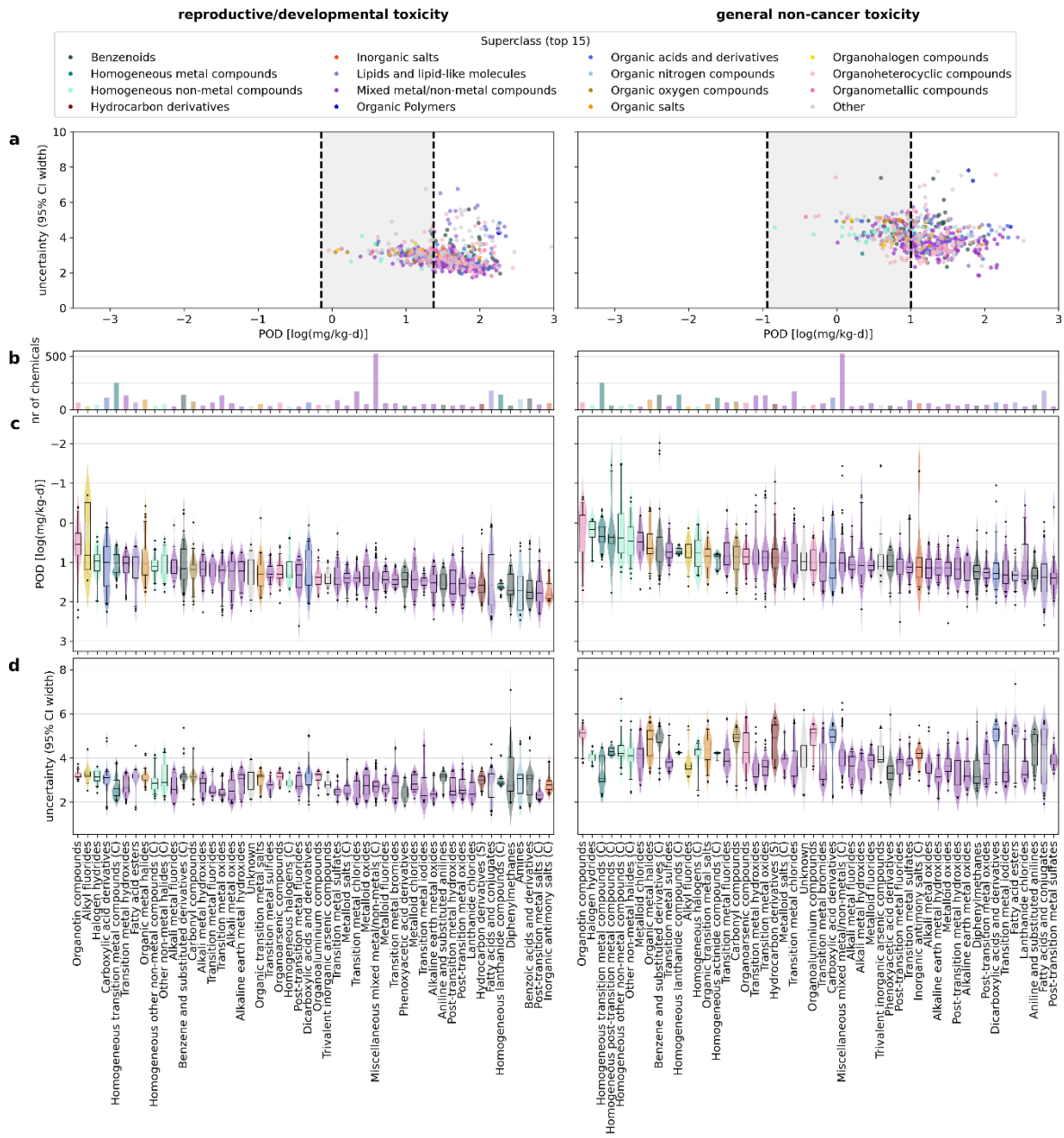

Figure S1-16. Ranking of non-standardized marketed chemicals grouped by chemical classes based on the lowest classification level (subclass, class (C) or superclass (S)) available in the ClassyFire chemical taxonomy showing **(a)** the scatterplot of predicted POD and associated uncertainty given by the 95% CI for all available chemical classes and further highlighting the top 50 chemical classes – ranked by their toxicity potential given by the median predicted POD – covering at least 30 marketed chemicals with **(b)** the number of chemicals in the top 50 chemical classes, **(c)** the distribution of predicted PODs in the top 50 chemical classes and **(d)** the distribution of associated prediction uncertainty given by the 95% CI in the top 50 chemical classes. All data points are colored by their respective superclass among the 15 most frequent ClassyFire superclasses across the set of marketed chemicals. **Abbreviations**: CI = confidence interval, POD = point of departure

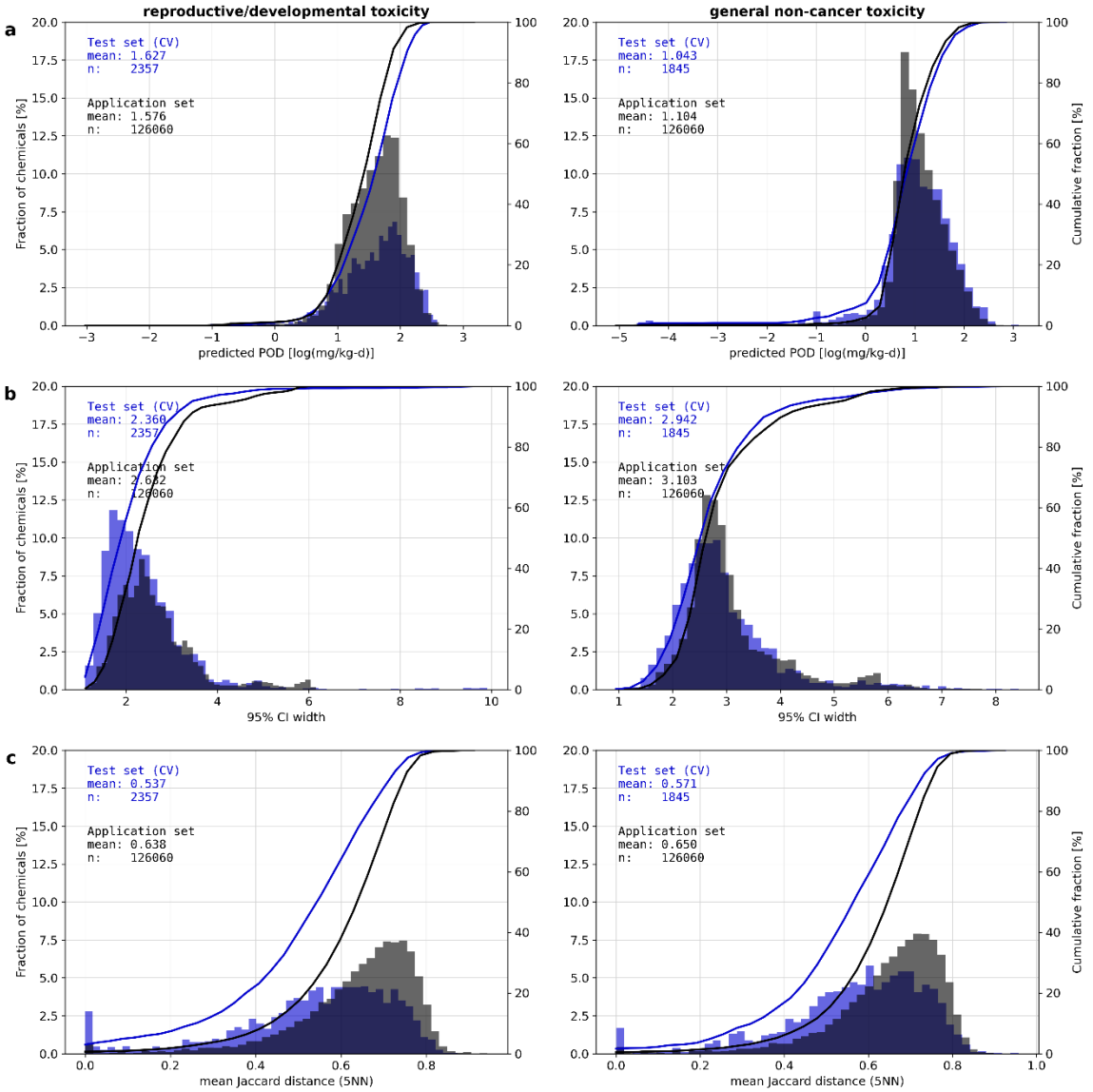

Figure S1-17. Histogram and cumulative distributions of **(a)** predicted POD, **(b)** prediction uncertainty (95% CI width) and **(c)** chemical dissimilarity (mean Jaccard distances) predictions of for reproductive/developmental (left) and general non-cancer (right) PODs for test and application chemicals (standardized marketed chemicals).

References

1. Fantke, P. *et al.* Exposure and toxicity characterization of chemical emissions and chemicals in products: global recommendations and implementation in USEtox. *International Journal of Life Cycle Assessment* **26**, 899–915 (2021).

2. DeVito, M. *et al.* The 2022 world health organization reevaluation of human and mammalian toxic equivalency factors for polychlorinated dioxins, dibenzofurans and biphenyls. *Regulatory Toxicology and Pharmacology* **146**, (2024).

3. Williams, A. J. *et al.* The CompTox Chemistry Dashboard: A community data resource for environmental chemistry. *J Cheminform* **9**, (2017).

4. Kim, S. *et al.* PubChem 2023 update. *Nucleic Acids Res* **51**, D1373–D1380 (2023).

5. Swain, M. *et al.* PubChemPy. Preprint at https://pubchempy.readthedocs.io/en/latest/index.html (2023).

6. PubChem. PubChem Identifier Exchange Service. https://pubchem.ncbi.nlm.nih.gov/idexchange/idexchange.cgi (2023).

7. Cherkasov, A. *et al.* QSAR modeling: Where have you been? Where are you going to? *Journal of Medicinal Chemistry* vol. 57 4977–5010 Preprint at https://doi.org/10.1021/jm4004285 (2014).

8. Gasser, L., Schür, C., Perez-Cruz, F., Schirmer, K. & Baity-Jesi, M. Machine learning-based prediction of fish acute mortality: implementation, interpretation, and regulatory relevance. *Environmental Science: Advances* (2024) doi:10.1039/d4va00072b.

9. RDKit. RDKit: Open-source cheminformatics (2022_09_05). Preprint at https://doi.org/10.5281/zenodo.7671152 (2022).

10. Winter, R., Montanari, F., Noé, F. & Clevert, D. A. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chem Sci* **10**, 1692–1701 (2019).

11. Durant, J. L., Leland, B. A., Henry, D. R. & Nourse, J. G. Reoptimization of MDL keys for use in drug discovery. *J Chem Inf Comput Sci* **42**, 1273–1280 (2002).

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

13. Cover, T. M. & Hart, P. E. *Approximate Formulas for the Information Transmitted Bv a Discrete Communication Channel*. *IEEE TRANSACTIONS ON INFORMATION THEORY* vol. 24 (1952).

14. Boser, B. E., Guyon, I. M. & Vapnik, V. N. *A Training Algorithm for Optimal Margin Classifiers*.

15. Breiman, L. *Random Forests*. vol. 45 (2001).

16. Chen, T. & Guestrin, C. *XGBoost: A Scalable Tree Boosting System*. https://github.com/dmlc/xgboost.

17. Rosenblatt, F. *The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain*. *Psychological Review* vol. 65 (1958).

18. Hüllermeier, E. & Waegeman, W. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Mach Learn* **110**, 457–506 (2021).

19. Romano, Y., Patterson, E. & Candès, E. J. Conformalized Quantile Regression. (2019).

20. Rossellini, R., Barber, R. F. & Willett, R. Integrating Uncertainty Awareness into Conformalized Quantile Regression. (2023).

21. Tran, D., Dusenberry, M. W., van der Wilk, M. & Hafner, D. Bayesian Layers: A Module for Neural Network Uncertainty. (2018).

22. Blundell, C., Cornebise, J., Kavukcuoglu, K. & Wierstra, D. Weight Uncertainty in Neural Networks. (2015).

23. Yang, C. I. & Li, Y. P. Explainable uncertainty quantifications for deep learning-based molecular property prediction. *J Cheminform* **15**, (2023).

24. Pernot, P. Properties of the ENCE and other MAD-based calibration metrics. (2023).

25. Levi, D., Gispan, L., Giladi, N. & Fetaya, E. Evaluating and Calibrating Uncertainty Prediction in Regression Tasks. *Sensors* **22**, (2022).

26. Yin, T., Panapitiya, G., Coda, E. D. & Saldanha, E. G. Evaluating uncertainty-based active learning for accelerating the generalization of molecular property prediction. *J Cheminform* **15**, (2023).