Chemical Science

EDGE ARTICLE

View Article Online
View Journal | View Issue

Cite this: *Chem. Sci.*, 2019, **10**, 8154

All publication charges for this article have been paid for by the Royal Society of Chemistry

Bayesian semi-supervised learning for uncertainty-calibrated prediction of molecular properties and active learning

Yao Zhang[ab] and Alpha A. Lee *[a]

Received 3rd February 2019
Accepted 4th July 2019

DOI: 10.1039/c9sc00616h

rsc.li/chemical-science

Predicting bioactivity and physical properties of small molecules is a central challenge in drug discovery. Deep learning is becoming the method of choice but studies to date focus on mean accuracy as the main metric. However, to replace costly and mission-critical experiments by models, a high mean accuracy is not enough: outliers can derail a discovery campaign, thus models need to reliably predict when it will fail, even when the training data is biased; experiments are expensive, thus models need to be data-efficient and suggest informative training sets using active learning. We show that uncertainty quantification and active learning can be achieved by Bayesian semi-supervised graph convolutional neural networks. The Bayesian approach estimates uncertainty in a statistically principled way through sampling from the posterior distribution. Semi-supervised learning disentangles representation learning and regression, keeping uncertainty estimates accurate in the low data limit and allowing the model to start active learning from a small initial pool of training data. Our study highlights the promise of Bayesian deep learning for chemistry.

1 Introduction

Predicting physiological properties and bioactivity from molecular structure – quantitative structure–property relationships (QSPR) – underpins a large class of problems in drug discovery. Classical QSPR workflows[1] separate descriptor generation – mapping a 2D[2–5] or 3D molecular structure[6,7] into a vector of real numbers using some handcrafted rules – and the machine learning method that connects descriptors to a property. Pioneering advances in machine learning such as graph neural networks directly take a molecular graph as input and infer the optimal structure-to-descriptor map from data,[8,9] outperforming classical machine learning methodologies with handcrafted descriptors.[10]

Nonetheless, graph neural networks are usually developed using frequentist maximum likelihood inference, with the benchmark being the mean error on a test set. However, if the goal of QSPR is to replace mission-critical but expensive experiments, a low mean error is insufficient: the user needs to have an estimate of uncertainty and know when the model is expected to fail. This is because typically only a small number of top-ranked predictions are selected to test experimentally, thus outliers can ruin a discovery campaign. Moreover, cost limits the number of experiments that can be run, thus an approach that judiciously designs the training set to maximise information gained is needed.

Uncertainty quantification, or domain applicability, has been extensively considered in the QSPR literature but not in the context of graph neural networks and not in a statistically complete way. Previous works estimate uncertainty of prediction as the distance in descriptor space between the input molecule and the training set, or training an ensemble of models and evaluating the variance.[11–14] More recent works consider conformal regression,[15,16] which trains two models, one for the molecular property and one for the error. However, there are two sources of uncertainty: epistemic uncertainty arises due to insufficient data in the region of chemical space that the model is asked to make predictions on. Aleatoric uncertainty arises due to noise in the measurements themselves (*e.g.* noisy biochemical assays).[17] Distance to the training set and variance within a model ensemble approximately capture epistemic uncertainty, whilst employing an ancillary model for prediction error approximately captures aleatoric uncertainty. We will show that the Bayesian statistical framework captures both sources of uncertainty in a unified and statistically principled manner.

Active learning strategies have been considered in the drug discovery literature.[18,19] However, those pioneering works considered *a priori* defined molecular descriptors, and estimate uncertainty *via* variance within an ensemble of models. Notwithstanding the shortcomings with incomplete modelling of uncertainty discussed above, employing graph neural networks in active learning presents unique opportunities and

[a]*Cavendish Laboratory, University of Cambridge, Cambridge CB3 0HE, UK. E-mail: aal44@cam.ac.uk*

[b]*Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge CB3 0WA, UK*

challenges: high model accuracy in the big data limit comes at the cost of being data-hungry. As the descriptor is fully data-driven, the model cannot estimate how "far" a compound is from the test set in the low-data limit, leading to poor uncertainty estimate and breaking down the active learning cycle. Low-data drug discovery has been considered in the context of one-shot learning[20] which estimates distance in chemical space by pulling data from related tasks. Nonetheless, this approach requires *a priori* knowledge on which tasks are related. Works on generative molecular design overcome this problem[21] by starting the active learning cycle with <1000 quantitative measurements, which impose a significant upfront experimental cost.

In this paper, we combine Bayesian statistics – a principled framework for uncertainty estimation – with semi-supervised learning which learns the representation from unlabelled data. We show that Bayesian semi-supervised graph convolutional neural networks can robustly estimate uncertainty even in the low data limit and drive an active learning cycle, and overcome dataset bias in the training set. Further, we demonstrate that the quality of posterior sampling is directly related to accuracy of the uncertainty estimates. As different Bayesian inference methods can be mixed and matched with different models, our study opens up a new dimension in the design space of uncertainty-calibrated QSPR models.

2 Methods and data

A machine learning method has two independent components: model and inference. The model is a function with parameters that relate the input to the output. Inference pertains to the methodology by which the model parameters are inferred from data. In terms of model, we focus on graph convolutional neural network models that take molecular graphs as input. In terms of inference, we focus on the Bayesian methodology.

2.1 Supervised graph convolutional neural network

Our baseline model is the graph convolutional fingerprint model.[9] The salient idea is the message passing operation,[22] which creates a vector that summarises the local atomic environment around each atom while respecting invariance with respect to atom relabelling. A molecule is described by a graph, where the nodes are atoms and the edges are bonds. Atom i is described by a vector of atomic properties $\mathbf{x}_v$, and a bond connecting i and j is described by bond properties $\mathbf{e}_{vw}$. The algorithm is iterative: at step t, each atom has a hidden state $\mathbf{h}_v^t$, which depends on "messages" $\mathbf{m}_v^t$ received from surrounding atoms as well as $\mathbf{h}_v^{t-1}$. The hidden states can be interpreted as descriptors of local atomic environment, and the messages allow adjacent atoms to comprehend the environment of its neighbours. Each atom is initialised to its atomic features, $\mathbf{h}_v^0 = \mathbf{x}_v$, and

$$\mathbf{m}_v^t = \sum_{w \in \mathcal{N}(\mathbf{v})} \left(\mathbf{h}_w^{t-1}, \mathbf{e}_{vw}\right), \tag{1}$$

and

$$\mathbf{h}_v^t = \sigma\left(\mathbf{H}_{t-1}^{\deg(v)} \mathbf{m}_v^t\right), \tag{2}$$

where $\mathcal{N}(\mathbf{v})$ denotes the set of atoms bonded to atom v, $\sigma(\cdot)$ is the sigmoid function, $\mathbf{H}_t^N$ is a learned matrix for each step t and vertex degree N. The algorithm is run T times, with T being a hyperparameter. In the final step, the output is given by a multilayer neural network $f(\cdot)$ that takes a weighted average of the hidden states at each step as input return a prediction,

$$y = f\left(\sum_{v,t} \mathrm{softmax}\left(\mathbf{W}_t \mathbf{h}_v^t\right)\right) \tag{3}$$

where $\mathbf{W}_t$ are learned readout matrices, one for each step t.

We use the implementation reported in the repository.† In all experiments, we consider $T = 3$, hidden layer at each level has 128 units, fingerprint length 256 (*i.e.* $\mathbf{W}_t \in \mathbb{R}^{128\times256}$), and $f(\cdot)$ is a two layer neural network with 128 units each and *relu* units.

2.2 Semi-supervised graph convolutional neural network

The fully supervised approach learns molecular descriptors directly from data. This is an advantage if one has a lot of data but a disadvantage in the data-limited settings such as active learning applications, where the objective is to design informative experiments starting from a small pool of initial training data.

The insight behind the semi-supervised approach is that significant amount of chemical knowledge is contained within the molecular structures themselves, without any associated molecular properties (*i.e.* unlabelled data). Thermodynamic stability puts constrains on what bonds are possible, and tends to put certain bonds near each other, forming persistent chemical motifs. For example, just by looking at drug molecules, one would immediately spot ubiquitous motifs such as amide group, benzene rings *etc.*, and some motifs often occur together as scaffolds.[23] The key assumption is that those persistent chemical motifs contribute to the molecular property that we want to predict. We can make mathematical progress by constructing a descriptor akin to eqn (1)–(3). However, the objective is no longer trying to fit a particular property. Rather, the hidden states $\mathbf{h}_v^t$, which summarises the atomic environment around atom v within radius t, are constructed such that they are predictable from the hidden states of the surrounding atoms. Therefore, the model learns a descriptor that clusters similar environments.

Specifically, we use the semi-supervised approach developed by Nguyen *et al.*,[24] which builds on the paragraph vector approach in natural language processing.[25] Given a set of molecular structures $\mathcal{M}$, the hidden states $\mathbf{h}_v^t$ maximise the log-likelihood

$$L = \sum_{m \in \mathcal{M}} \sum_{v \in m} \sum_{t=1}^{T} \log P\left(\mathbf{h}_v^t | \mathbf{u}_m\right), \tag{4}$$

$$P\left(\mathbf{h}_v^t | \mathbf{u}_m\right) = \frac{\exp\left(\left(\mathbf{h}_v^t\right)^T \mathbf{u}_m\right)}{\sum_{n \in \mathcal{M}} \exp\left(\left(\mathbf{h}_v^t\right)^T \mathbf{u}_n\right)}, \tag{5}$$

where $\mathbf{u}_n$ is the molecular identifier, obtained by maximising eqn (4), with $\mathbf{h}_v^t$ defined by eqn (1) and (2). We can interpret $\mathbf{u}_n$ as a vector that describes the "type" of molecule, and the objective encourages the hidden states $\mathbf{h}_v^t$ to take values such that similar molecules have similar atomic environments.

After finding parameters that maximise the objective (4), $\{\mathbf{h}_v^t\}$ are then passed to a neural network, eqn (3). The parameters of the neural network as well as the readout matrices $\mathbf{W}_t$ are learned in a supervised manner. Note that this formalism infers descriptors using unsupervised learning and uses supervised learning to relate descriptors to molecular properties.

We use the implementation reported in the Github repository‡ accompanying ref. 24. In all experiments, we consider $T = 3$, hidden layer at each level has 128 units, fingerprint length 256 (*i.e.* $\mathbf{W}_t \in \mathbb{R}^{128\times256}$), and $f(\cdot)$ is a two layer neural network with 128 units each and *relu* units.

2.3 Bayesian deep learning

In Bayesian inference, the aim is to determine the distribution of model parameters that conforms to the data, the so-called posterior distribution. Let $\boldsymbol{\theta}$ be model parameters, $\mathbf{x}_i$ the dependent variables and y_i the independent variable, such that

$$y_i = F(\mathbf{x}_i,\boldsymbol{\theta}) + \varepsilon_i. \tag{6}$$

where $\varepsilon_i \sim \mathcal{N}(0, {\sigma_i}^2)$ is the measurement noise. Bayes theorem states the posterior distribution, $P(\boldsymbol{\theta}|\{\mathbf{x}_i\},\{y_i\})$, is related to the likelihood, $P(\{y_i\}|\boldsymbol{\theta},\{\mathbf{x}_i\})$ and the prior $P(\boldsymbol{\theta})$ *via*

$$P(\boldsymbol{\theta}|\{\mathbf{x}_i\},\{y_i\}) = \frac{1}{Z}P(\{y_i\}|\boldsymbol{\theta},\{\mathbf{x}_i\})P(\boldsymbol{\theta}), \tag{7}$$

where Z is a normalising constant. The prediction for an unknown input $\hat{\mathbf{x}}$ is obtained by averaging over the posterior

$$\langle\hat{y}\rangle = \int P(\boldsymbol{\theta}|\{\mathbf{x}_i\},\{y_i\})F(\hat{\mathbf{x}},\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}. \tag{8}$$

The uncertainty of model predictions can be readily derived from this Bayesian formalism. There are two types of uncertainties.[17] First, the epistemic uncertainty, is given by the variance of the prediction with respect to the posterior

$$\mathrm{var}(\hat{y}) = \int P(\boldsymbol{\theta}|\{\mathbf{x}_i\},\{y_i\})(\langle\hat{y}\rangle - F(\hat{\mathbf{x}},\boldsymbol{\theta}))^2\mathrm{d}\boldsymbol{\theta}. \tag{9}$$

Second, the aleatoric uncertainty, is the intrinsic noise of the measurement ${\sigma_i}^2$. This aleatoric noise can depend on the input, ${\sigma_i}^2 = \sigma(\mathbf{x})^2$, as certain areas of the chemical space can be intrinsically more variable.

We note that the log posterior is, up to a constant,

$$-\log P(\boldsymbol{\theta}|\{\mathbf{x}_i\},\{y_i\}) = \sum_i\left(\frac{1}{2{\sigma_i}^2}(y_i - F(\hat{\mathbf{x}},\boldsymbol{\theta}))^2 + \frac{1}{2}\log {\sigma_i}^2\right) - \log P(\boldsymbol{\theta}), \tag{10}$$

which is exactly the mean-squared loss if σ_i is constant, with $\log P(\boldsymbol{\theta})$ being the regulariser. Therefore, maximum likelihood inference is a special case of Bayesian inference.

The Bayesian formalism is easy to state but computationally expensive. The numerical bottleneck is the numerical evaluation of the high dimensional integrals (8) and (9). A plethora of approximate numerical methods have been developed in the literature to overcome this bottleneck. However, there is no free lunch, and methods which approximate the posterior well are usually computationally expensive. In this paper, we will consider two approximate methods spanning the cost-accuracy spectrum.

2.3.1 Dropout variational inference. Variational inference seeks to approximate the posterior distribution by a distribution that is much easier to sample from. Ref. 17 and 26 show that a popular way to regularise neural networks – dropout – is equivalent to approximate Bayesian inference. The algorithm is simple: the neural network is forked at the last layer to have two outputs, the predicted aleatoric uncertainty ${\sigma_i}^2$ and dependent variable y_i, and trained to minimise the loss (10). However, during training, each unit has a probability p of being set to 0.

For a neural network with M units, ref. 17 and 26 show that the above algorithm is approximately equal to finding parameters $\boldsymbol{\theta} = (\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2\cdots\boldsymbol{\Theta}_M)$ that fit the distribution

$$q(\boldsymbol{\theta}) = \prod_{m=1}^{M}\boldsymbol{\Theta}_m z_m, \quad z_m \sim \mathrm{Bernoulli}(p), \tag{11}$$

to the posterior distribution $P(\boldsymbol{\theta}|\{\mathbf{x}_i\},\{y_i\})$, where $\boldsymbol{\Theta}_m$ is the parameter vector associated with the m^{th} unit.

Distribution (11), although not the same as the true posterior distribution, is significantly easier to sample: in the prediction phase, the model is run N times, and akin to the training phase each unit has probability p of being set to 0. The final prediction and total uncertainty is taken to be the mean over N different predictions of depending variable and variance, $\{y^i,(\sigma^i)^2\}_{i=1}^N$,

$$\langle y\rangle = \frac{1}{N}\sum_{m=1}^{N}y^m, \quad \mathrm{var}(y) = \frac{1}{N}\sum_{m=1}^{N}(y^m - \langle y\rangle)^2 + \frac{1}{N}\sum_{m=1}^{N}(\sigma^m)^2. \tag{12}$$

The first term in eqn (12) is the epistemic uncertainty and the second term is the aleatoric uncertainty.

In our numerical experiments, dropout is applied to every unit that is trained using supervised learning, *i.e.* every unit in the supervised graph convolutional neural network is subjected to dropout, whereas for the semisupervised case the layers on top of the hidden states are trained with dropout.

2.3.2 Stein variational gradient descent (SVGD). Rather than fitting a distribution to the posterior, Stein Variational Gradient Descent (SVGD)[27] directly draws samples from the posterior *via* gradient descent. Specifically, let $\{\boldsymbol{\theta}_i^0\}_{i=1}^N$ be parameters randomly and independently initialised in parameter space. We want to evolve parameters such that, after T steps, $\{\boldsymbol{\theta}_i^T\}_{i=1}^N$ are N independent samples drawn from $P(\boldsymbol{\theta}_j^t|\{\mathbf{x}_i\},\{y_i\})$. Ref. 27 shows that the following dynamical system does the trick:

$$\boldsymbol{\theta}_i^{t+1} = \boldsymbol{\theta}_i^t + \eta\phi(\boldsymbol{\theta}_i^t), \tag{13}$$

where

$$\phi(\boldsymbol{\theta}) = \frac{1}{N}\sum_{j=1}^{N}\left[k\left(\boldsymbol{\theta}_j^t, \boldsymbol{\theta}\right)\nabla_{\boldsymbol{\theta}_j^t}\log P\left(\boldsymbol{\theta}_j^t|\{\mathbf{x}_i\},\{y_i\}\right) + \nabla_{\boldsymbol{\theta}_j^t}k\left(\boldsymbol{\theta}_j^t,\theta\right)\right]. \tag{14}$$

and $k(\cdot,\cdot)$ is a generic kernel function and η is the learning rate. eqn (13) and (14) can be interpreted as free energy minimisation of an interacting particle system: a "particle" (parameter vector) is subjected to a "force" $\phi(\boldsymbol{\theta})$, which drives particles to regions of low energy (low loss), whilst forcing the particles apart to maximise entropy. The total uncertainty is evaluated also with eqn (12), except $\{y^m\}_{m=1}^N$ are predictions from different model parameters $\{\boldsymbol{\theta}_i\}_{i=1}^N$.

The key advantage of eqn (13) and (14) is that it is a well-defined approximation: frequentist inference (*c.f.* eqn (10)) is recovered if $N = 1$, whereas when $N \to \infty$ the system exactly samples from the posterior. Therefore, for finite N, the algorithm interpolates between frequentist and full Bayesian inference. The computational cost and memory demands increase with N, and in this paper we use $N = 50$.

To illustrate the computational demands of SVGD, Fig. 1 shows the wall clock time, on a Nvidia P100 GPU, as a function of the number of gradient updates steps for graph convolution with dropout, semi-supervised with dropout, and semi-supervised with SVGD on the melting point dataset discussed below. Both SVGD and Stochastic Gradient Descent use back-propagation to optimize the neural network parameters. For models trained using Stochastic Gradient Descent, the computational complexity of back-propagation at each iteration is $O(BM)$, where B is the number of training samples at each iteration and M is the number of parameters in the model. The semi-supervised model has less parameters than the fully supervised model, thus the wall-clock time is less per iteration. In SVGD, we need to update N Stein particles per iteration, thus wall clock time per iteration scales as $O(BMN)$.

2.3.3 Architectures and hyperparameters. As the objective of this paper is to demonstrate the types of chemical problems that Bayesian deep learning can tackle, we adopt common parameters for graph convolutional neural networks taken from the literature rather than performing extensive hyperparameter optimisation and neural architecture search. For both supervised and semi-supervised graph convolutional neural networks, we keep the number of hidden layers the same as the original implementation in the GitHub repositories cited above. Following the implementations in the repositories we keep the dimension of the fingerprint twice of n_h, the number of neurons in the hidden layers, and only optimise the n_h by a grid search over {32, 64, 128, 256}, choosing the value of n_h with the best averaged 5-fold cross-validation root mean squared error over all the datasets. This leads to the architecture of $T = 3$, $N = 50$, hidden units = 128 and two layers.

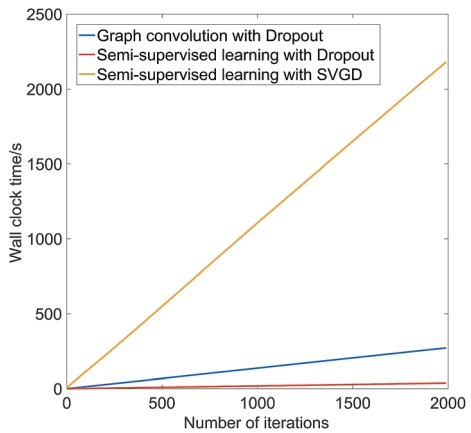

Fig. 1 The wall clock time on a Nvidia P100 GPU of graph convolution with dropout, semi-supervised with dropout, and semi-supervised with Stein Variational Gradient Descent on the melting point dataset.

2.4 Datasets

We consider a set of common regression benchmarks for physical properties prediction and bioactivity prediction. The melting point dataset is a collection of 3025 melting point measurements of drug-like molecules used in a benchmark study.[28] The ESOL dataset is a set of 1128 measured aqueous solubilities of organic molecules,[29] and the FreeSolv dataset is a set of 643 hydration free energy measurements of small molecules in water.[30] The ESOL and FreeSolv datasets are used in the MoleculeNet benchmark.[10] The CatS dataset comprises half-maximal inhibitory concentration ($\log IC_{50}$) measurements of 595 molecules against Cathepsin S, taken from the D3R Grand Challenge 3 and 4.[31] The Malaria dataset is a set of *in vitro* half-maximal effective concentration ($\log EC_{50}$) measurement of 13 417 molecules against a sulfide-resistant strain of *P. falciparum*, the parasite that causes malaria[32] used in benchmark.[9] The p450 dataset is a dataset of half-maximal effective concentration measurements of 8817 molecules against Cytochrome P450 3A4, a key enzyme for metabolism and clearance of xenobiotics, taken from the PubChem assay AID 1851.

To give the reader a sense of how "hard" the different datasets are, we consider a simple baseline model of XGBoost[33] on ECFP6 fingerprints.[34] We split the data into 80/10/10 (training/validation/testing). We take MaxDepth = 5, LearningRate = 0.01, and optimise the number of estimators (nEstimators = [50, 100, 150, 200, 250]) using the validation set. Table 1 shows the coefficient of determination R^2 and root mean squared error (RMSE). Judging from the coefficient of determination, the

Table 1 The performance of the baseline XGBoost model on the datasets considered in this paper

Dataset	Number of data points	R^2	RMSE
FreeSolv	643	0.765	1.90
ESOL	1128	0.704	1.14
CatS	595	0.654	0.367
MeltingPoint	3025	0.725	50.2
p450	8817	0.291	0.996
Malaria	13 417	0.499	0.655

dataset difficulty is (from easiest to hardest) FreeSolv < Melting < ESOL < CatS < Malaria < p450.

3 Results

3.1 Uncertainty quantification

We first consider how well can the model estimate its own uncertainty given the full dataset, split into training (80%) and test (20%) sets. The quality of the uncertainty estimate is operationalised by asking what is the model accuracy when the most uncertain predictions are removed, with uncertainty quantified by the variance computed from eqn (12). Our baseline method is graph convolution with dropout, which has recently been implemented in the DeepChem package as a feature,[10] although to our knowledge this is the first study that benchmarks Bayesian graph convolutional neural networks in terms of uncertainty quantification.

Fig. 2 shows that semi-supervised learning with SVGD accurately estimates uncertainty and significantly outperforms the baseline on every dataset. The plots show how the test set error varies as a function of confidence percentile – *i.e.* what is the error if we only consider the top n% of compounds in the test set ranked by confidence (note that confidence is inverse of the uncertainty, quantified by eqn (12)); the shaded region is one standard deviation, estimated by analysing 5 random partitions of the data into training and test sets. In every case, the error is a decreasing function of model confidence, thus the model successfully estimates which predictions are likely to be correct and which predictions are outliers.

Another metric that we can evaluate is the shape of the confidence–error curve. For ESOL and FreeSolv, the error is a steeply decreasing at the low confidence limit before plateauing, suggesting that most predictions are accurate but for a few outliers, which the Bayesian method can identify. The situation is different for MeltingPoint, Malaria and p450 – the error is slowly decreasing at the low confidence limit before sharply decreasing when it approaches the 100% confidence percentile limit (see also insets of Fig. 2). This suggests that a few predictions are very accurate, and Bayesian method can pick out those accurate predictions amid many less accurate ones. We note that our Bayesian model is well-suited for virtual screening applications, where the challenge is ensuring that the top-ranked actives picked out by the model are indeed actives, since only a very small proportion of the compounds ranked will actually be screened experimentally (the "early recognition problem").[35,36]

A lingering question whether the quality of the uncertainty estimate is due to a set of good descriptors (obtained *via* semi-supervised learning) or accuracy of the Bayesian methodology. Fig. 2 also shows that replacing SVGD with dropout significantly reduces model performance. At the same confidence percentile, SVGD consistently outperforms dropout. This suggests that the quality of posterior sampling drastically impacts the quality of uncertainty estimation.

The quality of uncertainty estimates can also be gauged by the correlation between the predicted uncertainty on test data points and the error that the model incurs. Table 2 shows the Spearman correlation coefficient between the predicted variance and model error. As expected, combining semi-supervised learning with SVGD leads to method with the highest rank correlation. This result is consistent with Fig. 2, which shows that semi-supervised learning with SVGD has the lowest confidence–error curve. Moreover, the rank correlation between predicted uncertainty and error broadly (although not exactly) follows the "difficulty" of the data (*c.f.* Table 1) – FreeSolv and ESOL have the highest rank correlation, and p450 has the lowest.

Previous works on domain applicability have focused on either building an auxiliary model to predict the error,[15,16] or

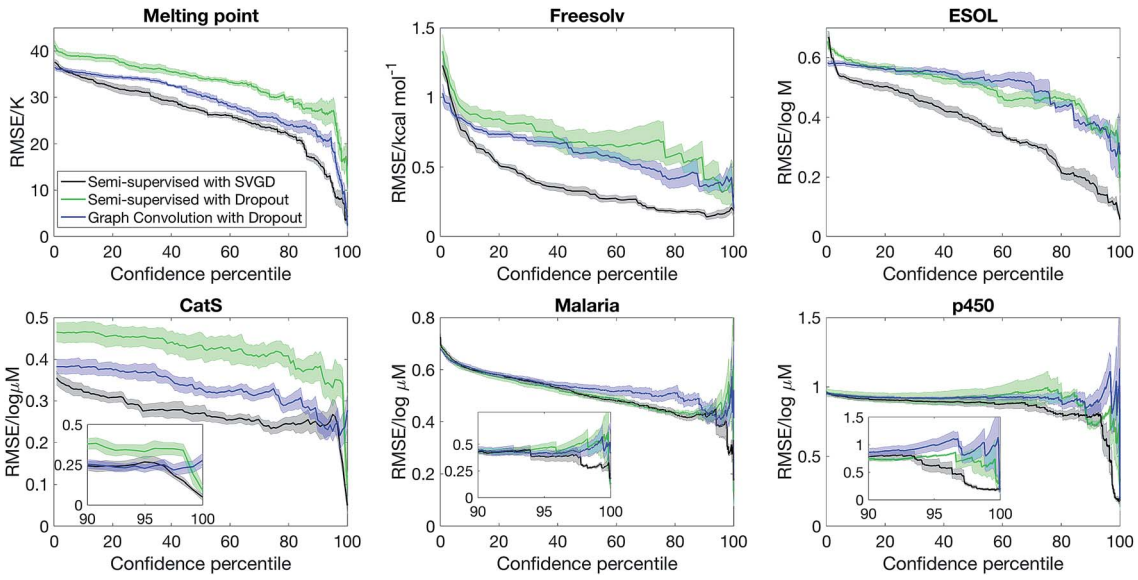

Fig. 2 Bayesian semi-supervised learning accurately predicts molecular properties and uncertainty. The plots show the model accuracy on the test set as a function of confidence percentile. The inset highlights the fact that predictions which the model are confident about are indeed accurate.

Table 2 Combining semi-supervised learning with SVGD yields uncertainty estimates that are strongly correlated with actual model error. The table shows the Spearman correlation coefficient between the variance predicted by the model and the error on test data points

Dataset	Graph convolution with dropout	Semi-supervised with dropout	Semi-supervised with SVGD
FreeSolv	0.531 ± 0.061	0.439 ± 0.093	**0.688 ± 0.053**
ESOL	0.112 ± 0.035	0.306 ± 0.079	**0.553 ± 0.026**
CatS	0.049 ± 0.036	0.066 ± 0.044	**0.310 ± 0.019**
MeltingPoint	0.192 ± 0.016	0.284 ± 0.035	**0.337 ± 0.013**
p450	0.167 ± 0.015	0.185 ± 0.049	**0.213 ± 0.010**
Malaria	0.315 ± 0.028	0.317 ± 0.031	**0.378 ± 0.019**

estimating the uncertainty of a prediction *via* the distance of the input to the training set.[11,12,14] The former models aleatoric uncertainty whereas the latter approximately captures epistemic uncertainty. Our Bayesian method captures both sources of uncertainty in a statistically principled manner. However, our model also provides independent estimates of epistemic and aleatoric uncertainties. As such, we can ask the question: is knowing epistemic or aleatoric uncertainty alone sufficient to estimate whether a prediction is accurate?

Fig. 3 shows that the confidence–error curve for semi-supervised learning with SVGD obtained by considering both epistemic and aleatoric uncertainty is below (or matches) that obtained by considering epistemic or aleatoric uncertainty alone. Considering both sources of uncertainty leads to much more accurate predictions at the high confidence limit for ESOL and p450. Moreover, there is no consistent trend as to whether epistemic or aleatoric uncertainty is more important – for ESOL, epistemic uncertainty is a better estimate of error than aleatoric uncertainty, whereas the opposite is true for p450 and CatS. As such, one cannot overlook epistemic or aleatoric uncertainty *a priori*, and our approach of combining both sources of uncertainty leads to an accurate uncertainty estimate.

3.2 Overcoming dataset bias

Our Bayesian methodology also overcomes dataset bias, which has be noted in the recent literature as the leading cause for overly optimistic results on benchmarks.[37] Most ligand-based benchmarks are biased in the sense that the molecules reported are tightly clustered around a few important chemical scaffolds, such that when the dataset is randomly split into training set and test set, the molecules in the test set are structurally very similar to the training set. Therefore, a model that only memorises the training set will still achieve a high accuracy on the test set yet cannot generalise to other regions of chemical space. Methods such as scaffold splitting[10] and attribution[38] attempt to estimate what would be the true performance of the model if the dataset were not biased. However, bias is fundamental in chemical data – an "uniform distribution" in chemical space does not exist because chemical space does not have a well-defined metric. Regardless of how one preprocesses the dataset or train the model, model predictions will always be awry for scaffolds that are not represented in the dataset. As such, rather than "unbiasing" the data, the practical question is whether the model can estimate whether it is likely

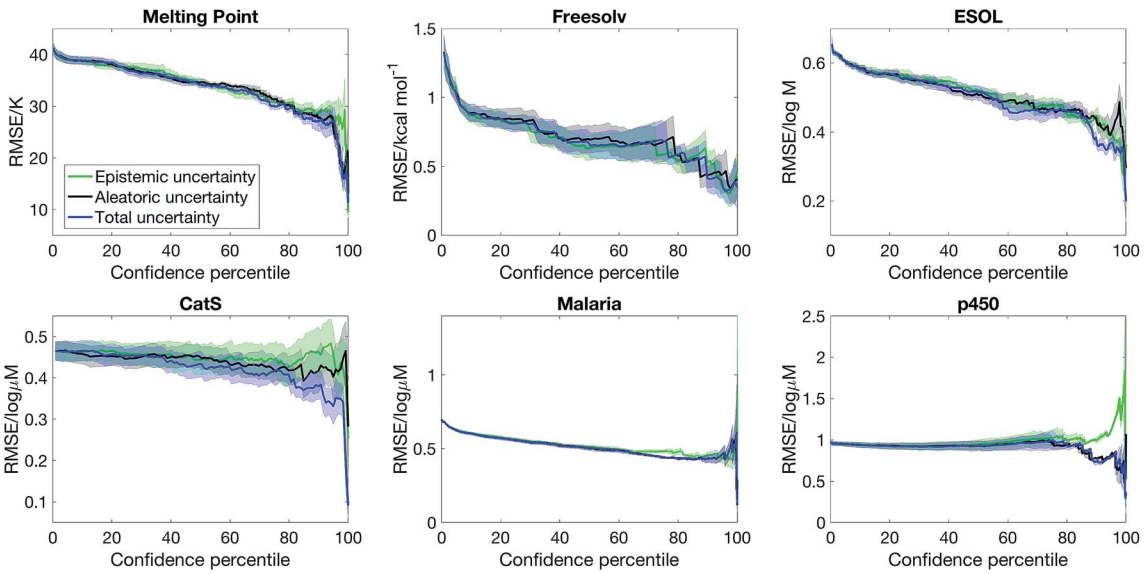

Fig. 3 Epistemic uncertainty and aleatoric uncertainty are distinct sources of uncertainty, and a combination of them is needed to obtain a good estimate of model error. We plot the confidence–error curve for semi-supervised learning with Stein Variational Gradient Descent where the confidence is estimated from combining epistemic and aleatoric uncertainty, epistemic uncertainty alone, and aleatoric uncertainty alone.

to make a correct prediction for an unseen molecule given a biased training set.

To show that Bayesian uncertainty estimation overcomes dataset bias, we consider a toy problem where we know the ground truth and deliberately introduce bias: we consider the problem of predicting octanol–water partition coefficient (log *P*) values, and use computed ACD log *P* values as a surrogate. We construct a dataset comprising all molecules on ChEMBL with either a beta-lactam or a benzodiazepine scaffold. The dataset is obviously very biased as it only contains 2 scaffolds. We then train a model using SVGD with semi-supervised learning, with the standard 8 : 2 split between training and test set on the biased dataset. Fig. 4 (left) show that model is reasonably accurate on the biased test set. We now simulate how an user might unwittingly fall foul of dataset bias – suppose we use the model to predict log *P* of all molecules with a steroid scaffold on ChEMBL. Fig. 4 (middle) show that the model performance, perhaps unsurprisingly, is poor. Steroids are not part of the training set, thus the model cannot predict its physiochemical properties. Bayesian uncertainty estimation provides a way out of this quandary – Fig. 4 (right) shows that the estimated uncertainty of log *P* prediction on steroids is significantly greater than log *P* prediction on the test set of beta-lactams or a benzodiazepines. In other words, the model can inform the user when it is inaccurate, thus mitigating the impact of dataset bias.

3.3 Low data active learning

Having considered the quality of the uncertainty estimates in the data-abundant limit, our next question is whether we can estimate uncertainty in the low data limit and drive an active learning cycle. We consider the objective of obtaining a low model error with a small training set. The model is first trained from a small initial pool of data (25% of the full training set, picked randomly), the model then selects a batch of molecules (2.5% of the full training set) that has the largest epistemic uncertainty to put into the training set, and then the model is retrained to suggest other additions, and the cycle continues. The test set is always 20% of the full dataset, held out at the beginning of the experiment. We note that other acquisition functions have been suggested in the literature,[39] and the objective function is problem-dependent.[18,19] Nonetheless, the goal of our experiment is to evaluate the quality of uncertainty estimate, thus we focus on a simple objective and acquisition functions. Further, as active learning requires constant retraining of the model, and SVGD is significantly more computationally intensive than dropout, we will only consider dropout variational inference.

Fig. 5 shows that semi-supervised learning significantly outperforms full supervised learning in the low data limit. The mean learning curves and error bars are obtained by analysing 20 active learning runs starting from random dataset splits. Moreover, in the case of full supervised learning, active learning is unable to deliver a better learning curve than random sampling, whereas for semi-supervised learning there is a sizeable gap between the learning curves of random sampling and active learning. This is because the full supervised method generates molecular descriptors directly from data. Therefore, in the low data limit, it is unable to learn descriptors that describe the structure of chemical space and chemical similarity between compounds, thus cannot generate meaningful uncertainty estimates to drive active learning.

The importance of choosing diverse compounds in the initial screen has been discussed extensively in the literature,[40–42] and the performance of our active learning method also depends on the chemical diversity in the initial screen. Fig. 6 shows that active learning does not outperform random sampling when the initial training set biased and contain only a small number of scaffolds. We model scaffold bias by splitting the data using scaffold splitting implemented in DeepChem,[10] and consider the Malaria example where active learning most clearly outperforms random sampling in Fig. 5. The underperformance of active learning is perhaps unsurprising – if the initial screen only consists of one scaffold, the knowledge that the model has on the other scaffolds would be minimal, *i.e.* the model is equally ignorant about all the other scaffolds. As such, randomly sampling the other scaffolds becomes a reasonable strategy.

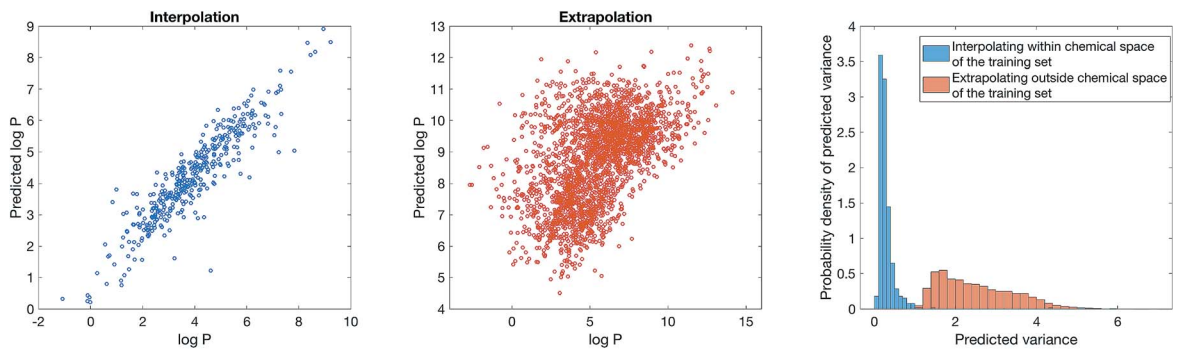

Fig. 4 Dataset bias can be mitigated with Bayesian uncertainty estimation. We consider a toy problem of predicting computationally calculated log *P* using Stein Variational Gradient Descent and semi-supervised learning, with a biased dataset comprising all beta-lactams or a benzodiazepines from ChEMBL. (Left) The model performance when the test set is also drawn from beta-lactams and benzodiazepines. (Middle) The model performance when the test set is all steroids from ChEMBL. (Right) The distribution of predicted uncertainty for the model applied to steroids and the model applied to beta-lactams and benzodiazepines.

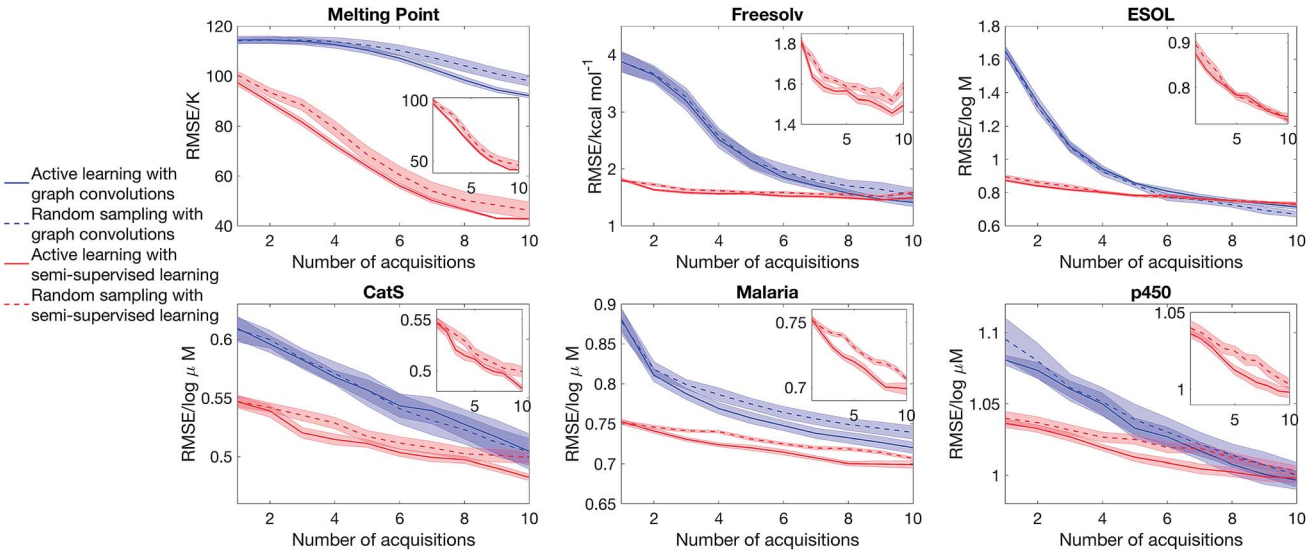

Fig. 5 Semi-supervised learning significantly outperforms full supervised learning in active learning. The model starts with 25% of the full training set, selected randomly, and at each iteration 2.5% of the full training set is added to the training set. The molecules added are picked randomly (random sampling) or picked because they have the largest predicted epistemic uncertainty (active learning). The curves show the mean model error and standard error of the mean, averaged over 20 active learning runs starting from random dataset splits, as a function of iteration. The insets focus on the performance of semi-supervised learning with SVGD.

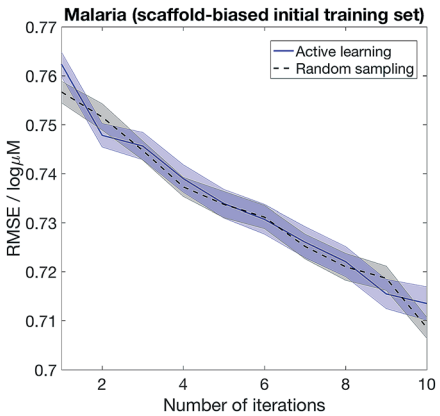

Fig. 6 Choosing diverse compounds in the initial screen is crucial to successful active learning. We first randomly split the Malaria dataset into training (80%) and test (20%) sets. We then scaffold-split the training set to obtain a biased initial set (25% of the total training set), and at each iteration 2.5% of the training set is given to the model, selected randomly (random sampling) or based on highest epistemic uncertainty (active learning).

4 Discussion and conclusion

We propose a novel method to quantify uncertainty in molecular properties prediction. We show that our methodology significantly outperforms the baseline on a range of benchmark problems, both in terms of model accuracy and in terms of uncertainty estimates. Our method also overcomes dataset bias by returning a large uncertainty estimate when the test set is drawn from a different region of chemical space compared to the training set. Moreover, our methodology can drive an active learning cycle, maximising model performance while minimising the size of the training set. The key to the success of our method is the combination of semi-supervised learning and Bayesian deep learning. Semi-supervised learning allows us to learn informative molecular representation in the low data limit. Bayesian deep learning allows us to estimate aleatoric and epistemic uncertainty in a statistically principled manner. We exemplified our methodology on regression as it is generally more challenging than classification, although it can be readily extended to classification problems.

Our observation that the choice of Bayesian inference methodology significantly impacts the quality of the uncertainty estimate suggests an evident followup that probes the mathematical limit of Bayesian inference – *i.e.* benchmarking approximate inference techniques against importance sampling of the posterior using Markov Chain Monte Carlo till convergence, which is computationally expensive but mathematically exact. Moreover, we note that most approximate inference techniques in the literature have been benchmarked in terms of RMSE error or log-likelihood,[43] rather than explicitly considering the quality of the uncertainty estimate in a manner relevant for chemoinformatics such as the confidence–error curve. An open question is the design of appropriate approximate inference techniques for graph convolutional neural networks that solves the trilemma between computational cost, model accuracy, and the quality of uncertainty estimate.

Another open question is whether the model has accurately disentangled aleatoric and epistemic uncertainty. Answering this question would require estimates of the ground truth aleatoric uncertainty, which is obtainable *via* repeating the experimental measurement and reporting the variance. Benchmark datasets which provide accurate experimental uncertainty estimates will be invaluable to advancing the Bayesian methodology.

Finally, our active learning methodology performs well when the initial screen covers diverse compounds. To successfully perform active learning on a scaffold-biased initial set, the model needs information on the bioactivity of those unseen scaffolds. We speculate that strategies such as multitask learning,[44,45] which pools information from other cognate assays which have explored the unseen scaffolds, will be a fruitful avenue.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

AAL acknowledges the support of the Winton Programme for the Physics of Sustainability.

Notes and references

† https://github.com/debbiemarkslab/neural-fingerprint-theano

‡ https://github.com/pfnet-research/hierarchical-molecular-learning

1 A. Cherkasov, E. N. Muratov, D. Fourches, A. Varnek, I. I. Baskin, M. Cronin, J. Dearden, P. Gramatica, Y. C. Martin, R. Todeschini, *et al.*, *J. Med. Chem.*, 2014, **57**, 4977–5010.

2 M. Randić, *J. Math. Chem.*, 1991, **7**, 155–168.

3 O. Ivanciuc, *J. Chem. Inf. Comput. Sci.*, 2000, **40**, 1412–1422.

4 J. L. Durant, B. A. Leland, D. R. Henry and J. G. Nourse, *J. Chem. Inf. Comput. Sci.*, 2002, **42**, 1273–1280.

5 D. Rogers, R. D. Brown and M. Hahn, *J. Biomol. Screening*, 2005, **10**, 682–686.

6 R. D. Cramer, D. E. Patterson and J. D. Bunce, *J. Am. Chem. Soc.*, 1988, **110**, 5959–5967.

7 J. Verma, V. M. Khedkar and E. C. Coutinho, *Curr. Top. Med. Chem.*, 2010, **10**, 95–115.

8 F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, *IEEE Trans. Neural Netw.*, 2009, **20**, 61–80.

9 D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik and R. P. Adams, *Advances in neural information processing systems*, 2015, pp. 2224–2232.

10 Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, *Chem. Sci.*, 2018, **9**, 513–530.

11 R. P. Sheridan, B. P. Feuston, V. N. Maiorov and S. K. Kearsley, *J. Chem. Inf. Comput. Sci.*, 2004, **44**, 1912–1928.

12 I. Sushko, S. Novotarskyi, R. Körner, A. K. Pandey, A. Cherkasov, J. Li, P. Gramatica, K. Hansen, T. Schroeter, K.-R. Müller, *et al.*, *J. Chem. Inf. Model.*, 2010, **50**, 2094–2111.

13 R. P. Sheridan, *J. Chem. Inf. Model.*, 2012, **52**, 814–823.

14 M. Toplak, R. Močnik, M. Polajnar, Z. Bosnić, L. Carlsson, C. Hasselgren, J. Demšar, S. Boyer, B. Zupan and J. Stålring, *J. Chem. Inf. Model.*, 2014, **54**, 431–441.

15 U. Norinder, L. Carlsson, S. Boyer and M. Eklund, *J. Chem. Inf. Model.*, 2014, **54**, 1596–1603.

16 F. Svensson, U. Norinder and A. Bender, *Toxicol. Res.*, 2017, **6**, 73–80.

17 A. Kendall and Y. Gal, *Advances in neural information processing systems*, 2017, pp. 5574–5584.

18 D. Reker and G. Schneider, *Drug Discovery Today*, 2015, **20**, 458–465.

19 D. Reker, P. Schneider, G. Schneider and J. Brown, *Future Med. Chem.*, 2017, **9**, 381–402.

20 H. Altae-Tran, B. Ramsundar, A. S. Pappu and V. Pande, *ACS Cent. Sci.*, 2017, **3**, 283–293.

21 R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2018, **4**, 268–276.

22 J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, arXiv preprint arXiv:1704.01212, 2017.

23 M. E. Welsch, S. A. Snyder and B. R. Stockwell, *Curr. Opin. Chem. Biol.*, 2010, **14**, 347–361.

24 H. Nguyen, S.-i. Maeda and K. Oono, arXiv preprint arXiv:1711.10168, 2017.

25 Q. Le and T. Mikolov, *International Conference on Machine Learning*, 2014, pp. 1188–1196.

26 Y. Gal and Z. Ghahramani, arXiv preprint arXiv:1506.02157, 2015.

27 Q. Liu and D. Wang, *Advances In Neural Information Processing Systems*, 2016, pp. 2378–2386.

28 C. W. Coley, R. Barzilay, W. H. Green, T. S. Jaakkola and K. F. Jensen, *J. Chem. Inf. Model.*, 2017, **57**, 1757–1772.

29 J. S. Delaney, *J. Chem. Inf. Comput. Sci.*, 2004, **44**, 1000–1005.

30 D. L. Mobley and J. P. Guthrie, *J. Comput.-Aided Mol. Des.*, 2014, **28**, 711–720.

31 Z. Gaieb, C. D. Parks, M. Chiu, H. Yang, C. Shao, W. P. Walters, M. H. Lambert, N. Nevins, S. D. Bembenek, M. K. Ameriks, T. Mirzadegan, S. K. Burley, R. E. Amaro and M. K. Gilson, *J. Comput.-Aided Mol. Des.*, 2019, **33**, 1–18.

32 F.-J. Gamo, L. M. Sanz, J. Vidal, C. de Cozar, E. Alvarez, J.-L. Lavandera, D. E. Vanderwall, D. V. Green, V. Kumar, S. Hasan, *et al.*, *Nature*, 2010, **465**, 305.

33 T. Chen and C. Guestrin, *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

34 D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.

35 A. Bender and R. C. Glen, *J. Chem. Inf. Model.*, 2005, **45**, 1369–1375.

36 J.-F. Truchon and C. I. Bayly, *J. Chem. Inf. Model.*, 2007, **47**, 488–508.

37 I. Wallach and A. Heifets, *J. Chem. Inf. Model.*, 2018, **58**, 916–932.

38 K. McCloskey, A. Taly, F. Monti, M. P. Brenner and L. Colwell, *Proc. Natl. Acad. Sci. U. S. A.*, 2019, **116**, 11624–11629.

39 Y. Gal, R. Islam and Z. Ghahramani, arXiv preprint arXiv:1703.02910, 2017.

40 D. J. Huggins, A. R. Venkitaraman and D. R. Spring, *ACS Chem. Biol.*, 2011, **6**, 208–217.

41 G. A. Bakken, A. S. Bell, M. Boehm, J. R. Everett, R. Gonzales, D. Hepworth, J. L. Klug-McLeod, J. Lanfear, J. Loesel, J. Mathias and T. P. Wood, *J. Chem. Inf. Model.*, 2012, **52**, 2937–2949.
42 S. Paricharak, O. Mendez-Lucio, A. C. Ravindranath, A. Bender, A. P. IJzerman and G. J. P. van Westen, *Briefings Bioinf.*, 2018, **19**, 277–285.
43 J. Mukhoti, P. Stenetorp and Y. Gal, arXiv preprint arXiv:1811.09385, 2018.
44 B. Ramsundar, B. Liu, Z. Wu, A. Verras, M. Tudor, R. P. Sheridan and V. Pande, *J. Chem. Inf. Model.*, 2017, **57**, 2068–2076.
45 J. Wenzel, H. Matter and F. Schmidt, *J. Chem. Inf. Model.*, 2019, **59**, 1253–1268.