Uncertainty Estimation Methods to Support Decision-Making in Early Phases of Drug Discovery

Philipp Renz
LIT AI Lab & Institute for Machine Learning
Johannes Kepler University Linz
renz@ml.jku.at

Sepp Hochreiter
LIT AI Lab & Institute for Machine Learning
Johannes Kepler University Linz
hochreit@ml.jku.at

Günter Klambauer
LIT AI Lab & Institute for Machine Learning
Johannes Kepler University Linz
klambauer@ml.jku.at

Abstract

It takes about a decade to develop a new drug by a process in which a large number of decisions have to be made. Those decisions are critical for the success or failure of a multi-million dollar drug discovery project, which could save many lives or increase life quality. Decisions in early phases of drug discovery, such as the selection of certain series of chemical compounds, are particularly impactful on the success rate. Machine learning models are increasingly used to inform the decision making process by predicting desired effects, undesired effects, such as toxicity, molecular properties, or which wet-lab test to perform next. Thus, accurately quantifying the uncertainties of the models' outputs is critical, for example, in order to calculate expected utilities, to estimate the risk and the potential gain. In this work, we review, assess and compare recent uncertainty estimation methods with respect to their use in drug discovery projects. We test both, which methods give well calibrated prediction and which ones perform well at misclassification detection. For the latter, we find the entropy of the predictive distribution performs best. Finally, we discuss the problem of defining out-of-distribution samples for prediction tasks on chemical compounds.

1 Introduction

The drug development process involves testing relevant properties of molecules in wet-lab tests. This process is expensive and has, in part, been replaced by machine learning models that predict these properties at a fraction of the costs [9, 14, 21, 34, 16] . Based on the results of these predictive models decisions have to be made if a compound is further investigated or not. As wrong decisions can lead to high costs, it would be of advantage to have models that give well calibrated confidence estimates about their predictions. As drug discovery is a very complex process it could be of advantage to rationalize it in the framework of expected utility theory [2], in which uncertainties substantially influence choices.

Although the task of uncertainty prediction has been treated to quite some extent in the machine learning community [24, 22, 33, 8, 19, 10], it has received less attention in chemical property prediction tasks. In [6, 7] approaches are proposed where the uncertainties in a regression task are derived from the spread of predictions of a snapshot-ensemble [17] or those of multiple forward runs using MC-dropout [8]. In [18] confidence intervals are determined from k-nearest neighbour

Workshop on Safety and Robustness in Decision-making at 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

distances in the latent space of DNN, but some of the experiments are not informative. For example, the confidence intervals for ensembles and MC-dropout are not calibrated in contrast to the latent space distance and confidence intervals are rated only by the fraction of data contained within them and not their sizes. [30] also apply MC-dropout to molecular prediction tasks and show largely inconclusive results on regression and do not establish that MC-dropout gives better results than DNNs not using it.

In light of these hitherto partly inconclusive results, we evaluate predictive uncertainty on 39 molecular property prediction tasks. First we evaluate which methods lead to well calibrated predictions as those are directly of interest for decision making and find that post-hoc calibration methods improve results. Secondly, we investigate which uncalibrated uncertainty scores [10, 22] perform well at misclassification detection. While such uncertainty scores cannot be employed in a probabilistic decision making framework directly, they can potentially be used to improve calibrated predictions. In this work, we do not treat this problem in detail however.

Uncertainty estimation is of special interest under distribution shift and is often evaluated using out-of-distribution (OOD) samples [24, 22]. OOD detection can be especially relevant when generative models for molecules [32] are optimized towards a particular property or activity given by a predictive model, as these generators are unaware of training data distribution of the predictive model.

We find that it is hard to meaningfully define OOD-samples for molecular prediction tasks and describe in a general way why this is the case.

2 Uncertainty Estimation

Here we focus on uncertainty estimation for binary classification tasks, such as toxic versus non-toxic molecules. Given a sample $x \in \mathcal{X}$ we want to predict its label $y \in \mathcal{Y} = \{0, 1\}$. While in classification tasks one often is only interested in finding the Bayes optimal classifier $\hat{y}(x) = \operatorname{argmax}_y p(y|x)$, in uncertainty estimation one tries to approximate the whole predictive distribution $p(y|x)$ over $\mathcal{Y}$.

The training set is given by a set of N i.i.d. samples $\{(x_i, y_i)\}_{i=1}^N$, drawn from a joint probability distribution $p(x, y)$. The goal is to train a model $f : \mathcal{X} \to [0, 1]$ by approximating $p(y = 1|x)$. The model is trained by minimizing the empirical risk over the training set $\frac{1}{N} \sum_{i=1}^N l(f(x_i), y_i)$, where $l : [0, 1] \times \{0, 1\} \to R^+$ is a loss function.

The quality of the predictive distribution can then be evaluated using proper scoring rules [11] which capture the quality of predictive uncertainty in contrast to metrics such as classification accuracy. In this work we use two such scoring rules, namely the negative log-likelihood (NLL) and the Brier score (BS) [5]. The NLL of a model f is given by

$$\text{NLL}(f) = -E_{p(x,y)}[y \ \log(f(x)) + (1 - y) \ \log(1 - (f(x)))], \tag{1}$$

and the Brier score by

$$\text{BS}(f) = E_{p(x,y)}[(y - f(x))^2]. \tag{2}$$

We include both of these metrics as the NLL can be very sensitive to misclassification with high confidence.

While optimally calibrated predictions give maximal information about uncertainties, we are also interested in scores $\alpha(x|f)$ that induce a partial order over points in $\mathcal{X}$ as proposed in [10, 22]. Under such an ordering samples with high expected loss should have a high uncertainty score such that optimally for a each pair of inputs (x_1, x_2):

$$\alpha(x_1|f) \leq \alpha(x_2|f) \iff E_{p(y|x)}[l(f(x_1), y)] \leq E_{p(y|x)}[l(f(x_2), y)] \tag{3}$$

This score can be used to construct a classifier with a reject option [3, 10] where a function g indicates if a sample is rejected or not. An input x is rejected if the uncertainty is higher than a chosen threshold t:

$$g(x, f, t) = \begin{cases} 1, & \text{if } \alpha(x|f) \leq t \\ 0, & \text{if } \alpha(x|f) > t. \end{cases} \tag{4}$$

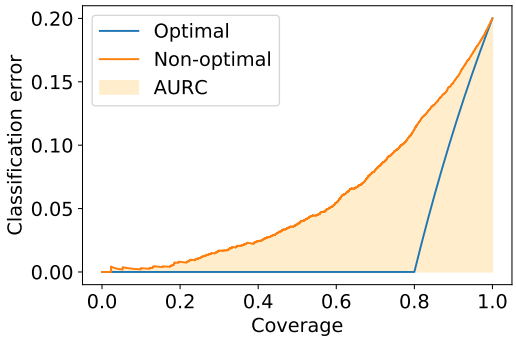

Figure 1: Example of two rejection curves. From right to left samples with high uncertainty are rejected, which decreases the error. The blue line corresponds to an optimal ordering in which misclassified samples are rejected first and the orange line to an imperfect one. The shaded area is the AURC and can be used to quantify the performance of an uncertainty ranking.

We define the fraction of the data not rejected as the *coverage* $\phi(g, f, t) = E_{p(x)}[g(x, f, t)]$ and the loss of non-rejected data as the *selective risk*

$$R(f, g) = \frac{E_{p(x,y}[l(f(x), y)g(x, f, t)]}{\phi(g, f, t)} \tag{5}$$

If the uncertainty score is informative one can trade off coverage with selective risk by varying t.

We evaluate uncalibrated uncertainties by calculating the *Area under rejection curve* (AURC)[10]. This metric measures how well the uncertainties detect erroneous predictions. An example is given in Figure 2 where we plot coverage against selective risk. Given a test set $T = \{(x_i, y_i)\}_{i=1}^{M}$ of size M that is sorted according to $\alpha(x_i|f)$ such that $\alpha(x_i|f) < \alpha(x_j|f)$ for $i < j$, we can calculate the area empirically by:

$$\text{AURC}(T, f, \alpha) = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{i} \sum_{j=1}^{i} l(\hat{y}_f(x_j), y_j), \tag{6}$$

where we did not explicitly denote the dependency of i on alpha to keep the notation uncluttered. If there are ties, we replace the loss of the respected points by their average in the sum above. For this metric we empirically infer significance values from the empirical distribution obtained from random orderings.

3 Methods

3.1 Calibrated uncertainty prediction

First we compare several methods that were proposed to obtain well calibrated predictions. For this comparison we use neural networks (NN) and random forests (RF)[4] as base classifiers. For each of these two we compare the quality of the following post-hoc calibration methods put on top of them:

- Raw: Sigmoid output of DNN or fraction of trees voting for a class for RFs
- Platt: Logistic regression trained on the log probabilities of the raw predictions on a calibration set [27]
- Temp: Same as Platt scaling but omitting the intercept term [12]
- Iso: Isotonic regression: Non-parametric regression from raw predictions with labels as targets on a calibration set [28]

Secondly we investigate MC-dropout [8], which has been proposed to obtain better uncertainty estimates. For MC-dropout, predictions are obtained by calculating multiple stochastic forward passes through a network trained with dropout by randomly dropping units just as is done during

training and subsequently averaging the predictions of those forward passes. This is in contrast to the original procedure proposed in [13] which only uses one forward pass after rescaling the network's weights.

3.2 Uncalibrated uncertainty prediction

Apart from testing which methods give the best calibrated predictions, we test the following ways to obtain uncalibrated uncertainty estimates:

- Entropy: Entropy of the predictive distribution
- k-NN: Average distance to k-nearest neighbours measured by Jaccard distance between ECFP6-fingerprints [29]
- k-FSNN: Average distance to k-nearest neighbours in the feature space (last layer before the softmax layer) of a NN using L2-distance as proposed in [18].
- Density: The estimated density of an input given by a generative likelihood model trained on the training set.

For the density method, we used a two layer LSTM [15] with 1024 units per layer for next-character prediction on SMILES-strings [35] as proposed in [32]. See Appendix B for implementation details.

4 Experiments

For our experiments, we used data from 39 assays from the ChEMBL database [1] and preprocessed the data as described in [23] to obtain classification tasks. The 39 tasks where chosen by requiring that there are more than 3000 positive as well as negative samples. For each task we randomly split the data set into 20 folds and took four of them as a test set. We ran a grid search to find suitable hyperparameters for DNNs and RFs. See Appendix A for tested values and for the exact training procedure. As post-hoc calibration methods also need a calibration set to be fit on, we introduced the size of the validation set as a hyperparameter as it might be the case that eventual benefits from calibration are outweighed by the negative consequences of a reduced training set.

4.1 Calibrated predictions: Negative log-likelihood

We aim at measuring how well the raw predictions are calibrated using the mentioned 39 tasks. To this end, we select the best hyperparameters for the models using the grid search in terms of NLL for each task and then calculate how many percent the best DNN or RF lags behind for each task. Overall, the raw predictions of RFs on average exhibit a 2.5% better NLL than DNNs indicating that the raw outputs of RFs typically represent probabilities better than the raw outputs of DNNs.

Next we compared post-hoc calibration methods. To this end we determined the best RF/DNN models for each calibration method and task and compared them to the best model on each task in Figure 2. For Random Forests, we observe that Platt scaling performs best, while the raw predictions often perform far worse. Isotonic regression even seems to be detrimental in comparison to the raw predictions. The second best performer is temperature scaling. From this we can infer that RF predictive distributions are biased towards one class which needs to be corrected by an intercept term. For DNNs we can see that the performance of raw predictions is closer to optimal than for RFs, which could be due to the fact that they are explicitly trained to optimize NLL. We can also observe that temperature scaling beats Platt scaling, which means that the predictions are not skewed towards one class and is in line with the results reported in [12]. The results look similar if BS is used to evaluate the models instead of NLL. This is of importance as isotonic regression optimizes BS in contrast to Platt and temperature scaling which optimize NLL. Thus we find that it is recommendable to use Platt or Temperature scaling for RFs or DNNs, respectively, as they robustly result in better performances.

MC-dropout has been introduced in the context of improved uncertainty predictions. Indeed we observe better NLL values for DNNs using MC-dropout, but it is unclear if this is caused by better calibration or by improved predictive performance. To disentangle these two effects, we select the best DNN architecture with/without MC-dropout for each task and compare NLL and accuracies of

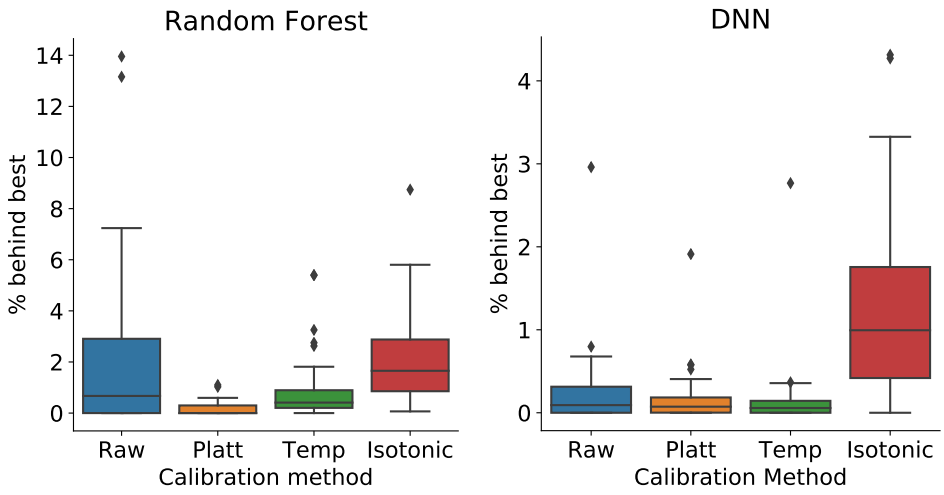

Figure 2: Comparison of calibration methods. We show the distribution of how many percent the best model using a certain calibration method is behind the performance of the overall best performance (determined separately for RFs/DNNs).

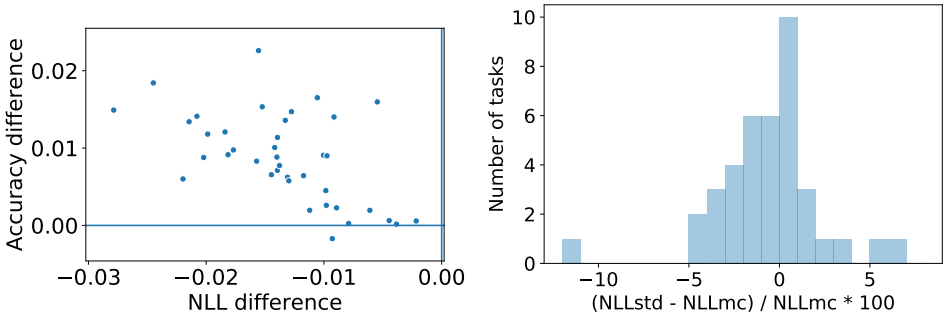

Figure 3: **Left**: Plot of the differences of NLL and accuracy across the learning tasks. Both axes show the score of the best models w/ dropout subtracted by the score the best model w/o dropout. That means lower/higher values an the x/y-axis correspond to dropout models being better. **Right**: Distribution of relative differences between standard dropout inference and MC dropout across the tasks. Negative values correspond to standard inference being better than MC-dropout. No trend towards either positive or negative values can be seen.

these models in Figure 3. We observed that using dropout results in both improved accuracy as well as NLL, from which we can infer that NLL gains are not necessarily only due to improved calibration.

Next we checked if MC-dropout improves uncertainty estimation by comparing networks that use MC-dropout to ones that use the "standard" evaluation mode proposed in [13]. To this end, we compare the performances of the best models using dropout, with standard inference mode and MC-dropout in Figure 3. From visual inspection we cannot make out which method is better across the tasks. A one sample t-test gives that there is no statistically significant difference between the standard inference method and MC dropout.

4.2 Misclassification detection: area under rejection curve

Next we performed tests to determine which uncalibrated uncertainty scores perform best. To do this we chose the best trained model according to NLL for each task and evaluated different uncertainty methods listed in the previous section. For each task and method we calculated the AURC and ranked the different methods on each task. We show the rank distribution of each method in 4.2. It can be observed that the entropy method performs best in all cases, which can be explained by the

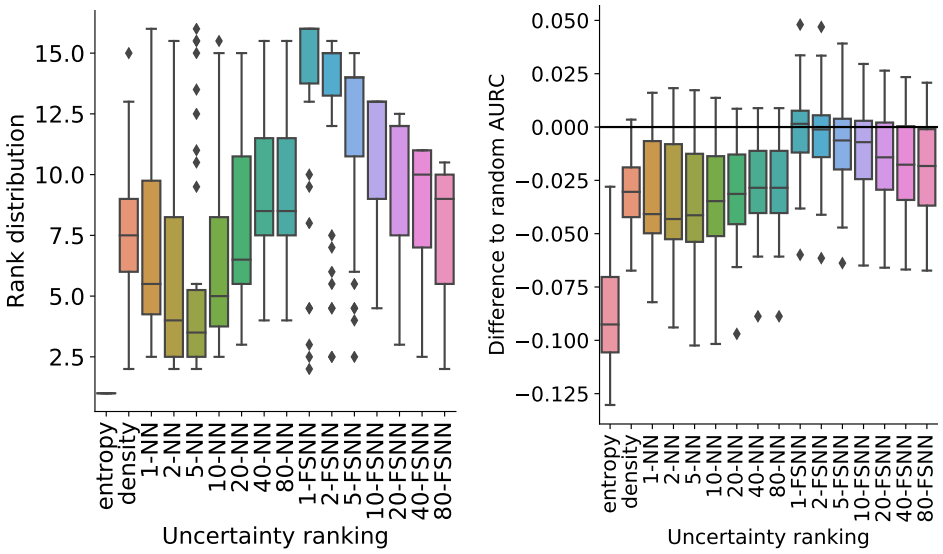

Figure 4: **Left**: Distribution of ranks for each method across all the tasks (lower is better). Entropy always performs best. **Right**: How the methods compare to the AURC of a random ranking. Values lower than zero indicate results that are better than random. Entropy provides significantly better results than other methods.

Table 1: Results of whether a method performs better or worse than random at a significance value of 0.05. The results are highly variable for some methods.

	entropy	density	2-NN	5-NN	1-FSNN	80-FSNN
Sig. better	39	31	30	30	12	24
Sig. worse	0	1	6	5	11	3

fact that it is the only method which was, at least implicitly, trained to solve this task. The nearest k-NN distances to the training set work second best, with 2-NN having the second best median rank. For k-FSNN we even found counterintuitive results, as test data with lower distances to training points have higher error rates for most settings of k. Therefore we inverted the ranking and gave higher uncertainty to points with low average distances to nearest neighbours. All results for k-FSNN presented here use this inverted ranking. We found that using a the average distance to a large number of neighbours to perform best across the tasks.

What is intruiging is the fact that for some methods the results are highly variable across tasks. To investigate this we determined significance values of the AURCs by looking at the empirical AURC-distribution obtained by sampling random uncertainty orderings. In Table 1 we observe that entropy performed significantly better than random in all the tasks at a 0.05 significance level. Density performed better than random in 31 tasks and significantly worse in one. We see that for a considerable amount of tasks are significantly worse than random for the k-NN approach. For 1-FSNN this effect is very pronounced. These results show that the tasks apparently have different characteristics in terms of the structure of the learned feature spaces and that method evaluation on single tasks can be misleading.

5 Uncertainty estimation under distribution shift

Uncertainty predictions are often evaluated under distribution shift. [24] evaluate predictive uncertainty using perturbed inputs and out-of-distribution (OOD) data, on which optimally only low confidence predictions should be made. These tests are especially meaningful as they explicitly capture the models ability to react to inputs that are semantically dissimilar to the in-distribution classes.

In [24] image and text classification tasks are considered. In contrast to molecular prediction tasks, for such tasks OOD-samples can be often meaningfully defined as there often is structure in the distribution of the inputs. The origin of this structure can be explained the causal structure of the data generation process [31]. Consider, for example, a handwritten digit classification task. The data is generated by persons thinking of a digit and then writing it down. Due to this, clusters that are semantically meaningful for the task form in input space. OOD-samples can then be defined as samples not belonging to those clusters.

In molecular prediction tasks the data is generated in a fundamentally different way. Inputs are drawn in a completely arbitrary way and then labels for them are determined. Therefore it is not evident that there are semantically meaningful clusters. Additionally we have the problem that in contrast to vision or NLP tasks, we do not have an intuitive understanding of chemistry. These facts make it hard to formulate informative OOD-detection tasks.

6 Conclusion

In this work we reviewed and compared uncertainty estimation methods in the context of decision-making in drug discovery. We found that for RFs, Platt scaling yields well calibrated predictions, while for DNNs temperature scaling is sufficient. We investigated MC-dropout and found that gains in uncertainty prediction are mainly attributable to increased accuracy, as MC-dropout does not give significantly better results than deterministic forward passes with rescaling.

Currently, our study is limited to RandomForests and descriptor-based feed-forward neural networks, which can be considered the two default methods. Furthermore, comparisons and assessments are performed on tasks with many active and inactive compounds, whereas in drug discovery oftentimes few actives are available. We plan to extend our study by including a large number of tasks covering different characteristics. This will inherently introduce the problem of how to best split the data into different sets for training, validation, calibration of the uncertainty measures and test set. Additionally we plan to include ensembles in the comparison which have been reported to yield good results in uncertainty estimation [6, 24].

We think that the results for k-NN/k-FSNN are particularly interesting, as it is often argued that uncertainty should increase "far" from the training data [8], which was not the case for all tasks considered in this study. The results show that this notion of "far" is very subjective and often not helpful in practice or that it is difficult to find an appropriate distance measure. We also note that the k-FSNN results are counter-intuitive and that distances in the feature space of a DNN are not easy to interpret in the context of uncertainty estimation. We also observed that the results vary strongly across tasks and that results on few or single tasks can be misleading. In future work we plan to include more methods into this comparison.

Overall we argue that obtaining good uncertainty estimates is important in the context of decision making and that there had been a lack of empirical evidence of which methods perform best in the area of drug discovery a and predictive models for molecules. With this study we hope to fill a part this void.

Acknowledgments. This work was supported by Janssen Pharmaceutica (MadeSMART), LIT grant DeepToxGen and AI-SNN. We thank the NVIDIA Corporation, Audi.JKU Deep Learning Center, Audi Electronic Venture GmbH, FFG grant 871302, and FWF grant P 28660-N31.

References

[1] Bento, A. P., Gaulton, A., Hersey, A., Bellis, L. J., Chambers, J., Davies, M., Krüger, F. A., Light, Y., Mak, L., McGlinchey, S., Nowotka, M., Papadatos, G., Santos, R., and Overington, J. P. (2014). The ChEMBL bioactivity database: An update. *Nucleic Acids Research*, 42(D1):D1083–D1090.

[2] Bernoulli, D. (1954). Exposition of a New Theory on the Measurement of Risk. *Econometrica*, 22(1):23–36.

[3] Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag, New York.

[4] Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

[5] Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3.

[6] Cortés-Ciriano, I. and Bender, A. (2019a). Deep Confidence: A Computationally Efficient Framework for Calculating Reliable Prediction Errors for Deep Neural Networks. *Journal of Chemical Information and Modeling*, 59(3):1269–1281.

[7] Cortés-Ciriano, I. and Bender, A. (2019b). Reliable Prediction Errors for Deep Neural Networks Using Test-Time Dropout. *Journal of Chemical Information and Modeling*.

[8] Gal, Y. and Ghahramani, Z. (2015). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. *arXiv:1506.02142 [cs, stat]*.

[9] Gawehn, E., Hiss, J. A., and Schneider, G. (2016). Deep learning in drug discovery. *Molecular informatics*, 35(1):3–14.

[10] Geifman, Y., Uziel, G., and El-Yaniv, R. (2018). Bias-Reduced Uncertainty Estimation for Deep Neural Classifiers. *arXiv:1805.08206 [cs, stat]*.

[11] Gneiting, T. and Raftery, A. E. (2007). Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*, 102(477):359–378.

[12] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. *arXiv:1706.04599 [cs]*.

[13] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580 [cs]*.

[14] Hochreiter, S., Klambauer, G., and Rarey, M. (2018). Machine learning in drug discovery. *Journal of Chemical Information and Modeling*, 58(9):1723–1724. PMID: 30109927.

[15] Hochreiter, S. and Schmidhuber, J. (1997). Long Short-term Memory. *Neural computation*, 9:1735–80.

[16] Hofmarcher, M., Rumetshofer, E., Clevert, D.-A., Hochreiter, S., and Klambauer, G. (2019). Accurate prediction of biological assays with high-throughput microscopy images and convolutional networks. *Journal of chemical information and modeling*, 59(3):1163–1171.

[17] Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. (2017). Snapshot Ensembles: Train 1, get M for free. *arXiv:1704.00109 [cs]*.

[18] Janet, J. P., Duan, C., Yang, T., Nandy, A., and Kulik, H. J. (2019). A quantitative uncertainty metric controls error in neural network-driven chemical discovery. *Chemical Science*.

[19] Kendall, A. and Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? *arXiv:1703.04977 [cs]*.

[20] Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*.

[21] Klambauer, G., Hochreiter, S., and Rarey, M. (2019). Machine learning in drug discovery. *Journal of Chemical Information and Modeling*, 59(3):945–946.

[22] Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2016). Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *arXiv:1612.01474 [cs, stat]*.

[23] Mayr, A., Klambauer, G., Unterthiner, T., Steijaert, M., Wegner, J. K., Ceulemans, H., Clevert, D.-A., and Hochreiter, S. (2018). Large-scale comparison of machine learning methods for drug target prediction on ChEMBL. *Chemical Science*.

[24] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. (2019). Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift. *arXiv:1906.02530 [cs, stat]*.

[25] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch. *NIPS-w*.

[26] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[27] Platt, J. (2000). Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Adv. Large Margin Classif.*, 10.

[28] Robertson, T., Wright, F. T., and Dykstra, R. (1988). *Order Restricted Statistical Inference*. Wiley.

[29] Rogers, D. and Hahn, M. (2010). Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754.

[30] Ryu, S., Kwon, Y., and Kim, W. Y. (2019). Uncertainty quantification of molecular property prediction with Bayesian neural networks. *arXiv:1903.08375 [cs, stat]*.

[31] Schoelkopf, B., Janzing, D., Peters, J., Sgouritsa, E., Zhang, K., and Mooij, J. (2012). On Causal and Anticausal Learning. *arXiv:1206.6471 [cs, stat]*.

[32] Segler, M. H. S., Kogej, T., Tyrchan, C., and Waller, M. P. (2017). Generating Focussed Molecule Libraries for Drug Discovery with Recurrent Neural Networks. *arXiv:1701.01329 [physics, stat]*.

[33] Tran, D., Dusenberry, M. W., van der Wilk, M., and Hafner, D. (2018). Bayesian Layers: A Module for Neural Network Uncertainty. *arXiv:1812.03973 [cs, stat]*.

[34] Unterthiner, T., Mayr, A., Klambauer, G., Steijaert, M., Wegner, J. K., Ceulemans, H., and Hochreiter, S. (2014). Deep learning as an opportunity in virtual screening. In *Proceedings of the deep learning workshop at NIPS*, volume 27, pages 1–9.

[35] Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36.

Table 2: Search grid for RF models

Hyperparameter	Values
Number of validation splits	0, 1, 2, 3
Number of trees	256, 1024, 2048
Splitting criterion	Gini, Entropy
Maximal features	Sqrt, log2, All
Maximal depth	4, 8, Unlimited
Minimal samples split	2, 8
Minimal samples per leaf	1, 4
Bootstrap	True, False

Table 3: Search grid for NNs

Hyperparameter	Values
Number of validation splits	0, 1, 2, 3
Optimizer	Adam [20], SGD w/ momentum
Learning rate	$10^{-2}, 10^{-3}, 10^{-4}$
Momentum (only for SGD)	0.5, 0.9
Number of hidden units	128, 256, 512, 1024
Number of layers	1
Hidden dropout rate	0.0, 0.25, 0.5, 0.75
Input dropout rate	0.0, 0.25, 0.5, 0.75

A Training procedure predictive models

To train the predictive models we followed the following procedures: For RFs we used the scikit-learn v0.21.2 [26] and tested the hyperparameters in Table 2.

DNNs were implemented in pytorch v1.1.0 [25]. Models were trained until validation NLL did not improve for 1200 training steps and then we reverted the model to the checkpoint with lowest NLL. For models trained without a validation set we trained we trained for 30000 steps which was sufficient to reach stable training NLL for most runs.

Calibration methods were fit on the validation data, if there is any for each model and the performances metrics were recorded for each of them. We will release our code upon publication.

B Training procedure for density model

For the density method we used a two layer LSTM [15] with 1024 units per layer implemented in pytorch v1.1.0. As optimizer we used Adam [20] with default parameters in pytorch. The network was trained in an autoregressive fashion to predict the next character in SMILES-strings [35] given all the previous ones. We pretrained the model on compounds from ChEMBL [1] that were not contained in any of the 39 prediction tasks. Then for each task we fine-tuned the model on the training set.