npj | computational materials

www.nature.com/npjcompumats

ARTICLE OPEN

Check for updates

Calibration after bootstrap for accurate uncertainty quantification in regression models

Glenn Palmer[1], Siqi Du[2], Alexander Politowicz[2], Joshua Paul Emory[2], Xiyu Yang[2], Anupraas Gautam[1], Grishma Gupta[1], Zhelong Li[2], Ryan Jacobs[2] and Dane Morgan[2]✉

Obtaining accurate estimates of machine learning model uncertainties on newly predicted data is essential for understanding the accuracy of the model and whether its predictions can be trusted. A common approach to such uncertainty quantification is to estimate the variance from an ensemble of models, which are often generated by the generally applicable bootstrap method. In this work, we demonstrate that the direct bootstrap ensemble standard deviation is not an accurate estimate of uncertainty but that it can be simply calibrated to dramatically improve its accuracy. We demonstrate the effectiveness of this calibration method for both synthetic data and numerous physical datasets from the field of Materials Science and Engineering. The approach is motivated by applications in physical and biological science but is quite general and should be applicable for uncertainty quantification in a wide range of machine learning regression models.

npj Computational Materials (2022)8:115 ; https://doi.org/10.1038/s41524-022-00794-8

INTRODUCTION

Machine learning is seeing an explosion of application in physical and biological science for predicting properties of chemical and materials systems, expanding the widely used tools of Quantitative Structure Property/Activity Relationship (QSPR/QSAR) modeling. Machine learning applications frequently make use of supervised regression to predict the desired target property as a function of easily accessible features. However, uncertainty quantification (UQ) remains a major challenge for these models. UQ approaches for regression models generally fall into four categories that include methods based on: (1) feature data, (2) ensembles of models, (3) a statistical estimator to approximate the variance of predictions, and (4) Bayes' theorem. Recent work on UQ for QSPR/QSAR has investigated all of these approaches[1–7]. Despite this work, methods appear to vary in effectiveness between datasets[1], and there is no obvious choice for a UQ method that is simultaneously easy to implement, generalizable, and empirically validated.

A widely used and extremely general approach to UQ in machine learning is based on the ensemble method, which is a technique that makes predictions using multiple machine learning models and outputs a weighted average as the final prediction. A common technique to construct an ensemble of predictors is through bootstrap aggregating, which is also called bagging. In the bagging approach, multiple models are each trained on a subset of data from the training set, where each subset is constructed by choosing points uniformly at random with replacement[8]. A well-established machine learning approach that uses bagging is random forests, which are made up of a bootstrap ensemble of decision trees[9].

Because ensemble methods work by computing a distribution of predicted values, they naturally lend themselves to computing estimates of uncertainty. There has been substantial interest in developing theoretically sound and computationally efficient methods for UQ in ensemble methods in the past few decades[10–18]. Particular attention has been given to the use of bootstrap and jackknife resampling methods in classification[11–14], including methods of rescaling estimated error rates to make them more accurate[12]. Work has also been done on the use of ensemble methods for variance estimation in regression. One thread of research has focused on applying a secondary resampling technique such as bagging or the related jackknife method to bagged learners to estimate variance[10,11,16]. This resampling on top of the original bootstrapping can be very computationally intensive, so a major focus has been on making these methods more efficient. However, efforts toward making these resampling techniques more computationally efficient can lead to biased estimates of variance, and efforts to correct these biases appear to have mixed results empirically[16]. In the second thread of related work, for the specific context of random forests and other tree-based models, Lu and Hardin[18] provided a promising framework for estimating the distribution of prediction errors for a test-set observation x using a weighted sum of prediction errors for out-of-bag cohabitants of x in the trees comprising the model. However, there is no way at present to apply this approach outside of tree-based models.

One general and common UQ technique is to use the standard deviation of the predicted values from the ensemble as an uncertainty estimate[1,2,5,6]. Such an approach to uncertainty calculation has multiple advantages, including being simple to implement and parallelize, and maintaining desirable features of the original regression model (e.g., perhaps being differentiable with respect to certain features). While this technique has in some cases been shown to correlate well with the uncertainty of predictions[19], it is not clear that this standard deviation should accurately predict the standard error of predictions in general. In fact, if the test set includes points very different from training data, the ensemble standard deviation has been shown to substantially underestimate uncertainty[2,3].

In the context of this work, calibration refers to any transformation of an uncertainty estimate to make it more accurate that can then be applied to new predictions, and it is a natural approach to take when uncertainty estimates can be assessed in some manner. In particular, once an uncertainty estimate has been computed, it is almost always possible to use

[1]Department of Computer Sciences, University of Wisconsin-Madison, Madison, WI, USA. [2]Department of Materials Science and Engineering, University of Wisconsin-Madison, Madison, WI, USA. ✉email: ddmorgan@wisc.edu

some form of cross-validation data to judge its accuracy and, if necessary, modify the estimate to be more accurate. For example, Kuleshov et al.[20] proposed a calibration method for deep learning uncertainty estimates by generalizing the method of Platt scaling[21] used in classification and demonstrated that it could be used to produce accurate confidence intervals. Beyond improving accuracy, one benefit of using some sort of calibration post-processing step is that it can allow for more interpretable values from UQ methods which initially only provide estimates of relative uncertainty. For example, when using uncertainty estimates based on distances in feature space, Hirschfeld et al.[1] proposed calibrating the estimates so that they can be interpreted as variances. In their calibration method, the distance-based uncertainty estimate $U(x)$ is assumed to be linearly related to the prediction variance $\hat{\sigma}^2(x)$ such that $\hat{\sigma}^2(x) = aU(x) + b$. The values of a and b can be found by minimizing the sum of the negative log-likelihoods that each predicted value in the validation set was drawn from a normal distribution with a mean equal to the true value of the point, and variance equal to $aU(x) + b$. (See equations (11) and (12) in Hirschfeld et al.[1]) Janet et al.[2] showed that for neural network predictions, the distance to available data in the latent space of the neural network, when calibrated by a linear rescaling similar to Hirschfeld et al., provided a more accurate error estimate than using the standard deviation of predictions by an ensemble of neural networks. Similarly, Levi et al.[22] demonstrated that the same calibration approach can yield accurate uncertainties when applied to standard deviations predicted directly by a neural network, and Busk et al.[23] further refined this result by showing that a nonlinear scaling function is also effective. All of these works provide calibrations for specific cases to improve uncertainty quantification from feature distances or neural network predicted variances. However, they do not demonstrate that the approaches can be used more generally or propose directly calibrating ensemble errors.

Musil et al.[24] proposed using the same type of log-likelihood optimization as Hirschfeld et al.[1] for calibrating ensemble standard deviation as a UQ metric. They demonstrated that such calibration was effective for a large dataset with ensembles of sparse Gaussian process regression models generated by subsampling, and proposed that the calibration could also be applied to other models and methods of generating ensembles, such as bootstrapping. In this paper, we extend this work to show that this calibration technique is extremely general. In particular, we demonstrate that the approach works for multiple model and dataset types, many of which are quite small, and therefore one might expect to have difficulty providing robust error estimation. We also demonstrate the effectiveness of this calibration approach in the presence of varying amounts of Gaussian noise. Finally, we provide approaches to visualize the accuracy of uncalibrated and calibrated models, including distinguishing effects of true failures in the method as opposed to failures simply due to the presence of poor sampling.

We begin by investigating the accuracy of the standard deviation of bootstrap ensemble predictions in estimating the true standard error of predicted values. We refer to this bootstrap estimate as $\hat{\sigma}_{uc}$ (where the subscript *uc* refers to uncalibrated and we use the ^ notation to refer to predicted values from the machine learning model(s)). Then, we implement a method of calibrating $\hat{\sigma}_{uc}$ to yield a calibrated estimate $\hat{\sigma}_{\mathrm{cal}}$ (where the subscript *cal* refers to calibrated), and demonstrate the effectiveness of this calibration in producing highly accurate uncertainty estimates across multiple sets of both synthetic and physical data. We will use the general symbol $\hat{\sigma}$ when referring nonspecifically to one or both of $\hat{\sigma}_{uc}$ and $\hat{\sigma}_{\mathrm{cal}}$. We evaluate $\hat{\sigma}$ for random forests, bootstrap ensembles of linear ridge regression models, bootstrap ensembles of Gaussian process regression (GPR) models, and bootstrap ensembles of neural networks. We perform these evaluations at some level for a total of 10 datasets.

We use two methods of evaluating the accuracy of $\hat{\sigma}$. First, we examine the distribution of the ratios of test-set residuals (differences between observed and predicted values) to corresponding values of $\hat{\sigma}$, which we call the r-statistic distribution:

$$r = \frac{\text{residual}}{\hat{\sigma}} \tag{1}$$

as proposed by Ling et al.[25] and further applied by Lu et al.[26] If $\hat{\sigma}$ accurately represents the standard deviation of residuals and the model has no bias, the r-statistic distribution should be normally distributed with a mean of zero and a standard deviation of one. Thus, the closeness of the r-statistic distribution to a standard normal distribution is a method of assessing the accuracy of $\hat{\sigma}$. One weakness of assessment with the r-statistic distribution is that it does not provide a way to directly evaluate whether larger (smaller) values of $\hat{\sigma}$ correspond to larger (smaller) residuals. To address this weakness, we also plot binned values of $\hat{\sigma}$ (x-axis) against the root mean square of the residuals (y-axis) in each bin (see Methods for details). We call this an RMS residual vs. $\hat{\sigma}$ plot, or, put more succinctly, a residual vs. error (RvE) plot, and it was first introduced for this type of analysis by Morgan and Jacobs[19]. A similar visualization was independently proposed by Levi et al.[22]. For perfectly accurate $\hat{\sigma}$ and infinite sampling, we would expect this plot to have a slope of one and an intercept of zero, i.e., the magnitude of $\hat{\sigma}$ should correlate perfectly with the root mean square of the model residuals. By comparing the points on this plot to the identity function, one can understand whether $\hat{\sigma}$ is underestimating or overestimating the standard deviation of residuals at various levels of uncertainty. We show that $\hat{\sigma}_{uc}$ correlates with the actual standard deviation of residuals (i.e., larger ensemble standard deviations correspond to larger standard deviations of residuals), but that for different models and datasets, they systematically underestimate or overestimate the true values.

After evaluating $\hat{\sigma}_{uc}$ using the above methods, we implement a log-likelihood optimization calibration scheme similar to the one from Hirschfeld et al[1]. described above, and evaluate the quality of $\hat{\sigma}_{\mathrm{cal}}$ after calibration. The only difference between our log-likelihood optimization method and the log-likelihood optimization method described above from Hirschfeld et al. is that we assume the estimated standard deviation is linearly related to the true standard deviation, rather than the variances being linearly related. We show that this calibration method is highly effective at calibrating the ensemble standard deviation as an estimate of uncertainty. Finally, we compare our values of $\hat{\sigma}_{\mathrm{cal}}$ for GPR to the Bayesian uncertainty estimates typically used for a single GPR model, which we refer to as $\hat{\sigma}_{\mathrm{GPR}}$ and show that $\hat{\sigma}_{\mathrm{cal}}$ is more accurate than either uncalibrated or calibrated values of $\hat{\sigma}_{\mathrm{GPR}}$. This is consistent with previous findings of Musil et al.[24].

Overall, our results suggest that the calibrated ensemble standard deviation as an uncertainty estimate can be used across many different machine learning models and datasets. The method can be applied to any model where the bootstrap and CV ensemble generation is not computationally prohibitive. The approach is therefore likely to be of significant utility for many machine learning applications.

RESULTS

Comparison of uncalibrated and calibrated estimates

In this section, we show the effects of calibration on select combinations of models and datasets to illustrate several key points. The complete set of all model and dataset combinations we considered are included in the Supplementary Information and show results completely consistent with those shown in the main text. First, we used the r-statistic distribution and RMS residual vs. $\hat{\sigma}$ plots to evaluate our calibration method for random forest models using the synthetic[27], diffusion[26], and perovskite[28]

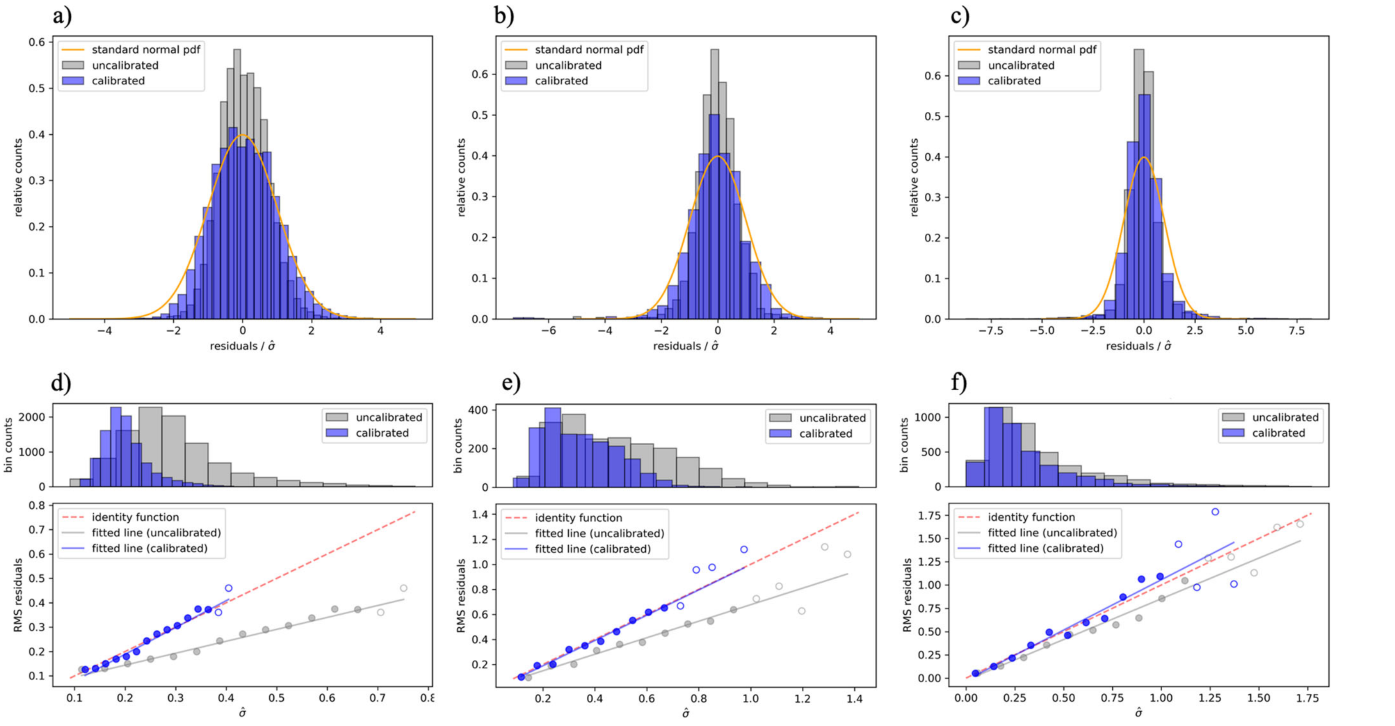

Fig. 1 Calibrated and uncalibrated r-statistic and RvE plots for random forest. Distributions of r-statistic values (top row) and RMS residual vs. $\hat{\sigma}$ plots (bottom row) for random forest, using the synthetic (panels **a** and **d**), diffusion (panels **b** and **e**), and perovskite (panels **c** and **f**) datasets, shown with both uncalibrated and calibrated $\hat{\sigma}$ values. Markers which are not filled in represent bins with fewer than 30 points. Statistics for each plot are summarized in Table 1.

datasets (see Datasets sub-section in the Methods for more information). These results are shown in Fig. 1. Note that before calibration, there is a consistent positive slope in the RMS residual vs. $\hat{\sigma}$ plots as $\hat{\sigma}_{uc}$ becomes larger—that is, larger uncertainty estimates correspond to larger residual values. However, for all three datasets, the uncalibrated points on the RMS residual vs. $\hat{\sigma}$ plots fall below the identity function, meaning they are overestimating the uncertainty of predictions. After calibration, all three r-statistic distributions appear to be reasonably close to standard normal distributions, and the binned uncertainty estimates lie very closely on the identity function line. Note that points in Fig. 1, which are not filled in represent bins that have fewer than 30 points and therefore may deviate from the identity function just due to poor sampling. In general, we find that calibrated points with good sampling are quite close to the identity function line. Overall, the results in Fig. 1 demonstrate that our calibration method performs very well for ensembles of random forest models. To further illustrate the power and general applicability of our method, we have performed our recalibration method using random forests on seven additional datasets from the materials science community (see Datasets sub-section in the Methods for more information), for a total of 10 datasets (1 synthetic, 9 materials datasets). In all cases, the $\hat{\sigma}_{uc}$ values are greatly improved with our calibration approach (also see Supplementary Table 1). See Supplementary Figs. 55–68 in the Supplementary Information for r-statistic and RMS residual vs. $\hat{\sigma}$ plots for these additional datasets.

Calibrated ensembles of GPR and linear models

In Fig. 2, we show the r-statistic and RMS residual vs. $\hat{\sigma}$ plots for the diffusion dataset using a bootstrap ensemble of 200 GPR models and a bootstrap ensemble of 500 linear ridge regression models. Before calibration, the GPR values of $\hat{\sigma}_{uc}$ appear to underestimate the true uncertainty, while after calibration, the r-statistic distribution appears very close to standard normal, and the well-sampled points lie closely on the identity function line. We did not expect our method to work as well for linear ridge regression since the uncertainty in linear regression models is typically dominated by bias, while the bootstrap standard deviation attempts to capture variance (see further discussion of this issue below). As shown in Fig. 2, the r-statistic distribution for the calibrated linear ridge regression uncertainty estimates appears to be close to a standard normal distribution. However, the fitted line to the calibrated points on the RMS residual vs. uncertainty-estimate plot has a slope of 0.597, substantially less than 1. We also ran a test using neural network models, using a smaller ensemble and just one dataset as the fitting is much more computationally demanding. Specifically, we used an ensemble of 25 neural networks on the diffusion dataset and found our calibration scheme performed well for well-sampled cases, with calibrated errors close to the identity line (See Supplementary Figs. 53–54). Additional study is needed to more thoroughly explore the UQ behavior of neural network ensembles, specifically, to assess how well our approach performs for different types of network architectures and non-bootstrap methods of generating ensembles (e.g., selection of different models during training, restarting from different initial weights, and dropout).

Comparison of ensemble and Bayesian error estimates of GPR

To provide a comparison to our bootstrap ensemble method of uncertainty estimation for GPR, in Fig. 3 we show the results of our calibration method applied to the Bayesian standard deviation estimates $\hat{\sigma}_{GPR}$, which are often used for UQ for GPR. In the RMS residual vs. $\hat{\sigma}_{GPR}$ plot, the uncalibrated $\hat{\sigma}_{GPR}$ values do not seem to be strongly correlated with the residuals. Because of this, after calibration, several well-sampled points deviate substantially from the identity function line. Note that as shown in Table 1, the uncalibrated and calibrated fitted lines have slopes of 0.151 and 0.343, respectively. Furthermore, both the uncalibrated and calibrated r-statistic distributions have a sharp peak close to 0,

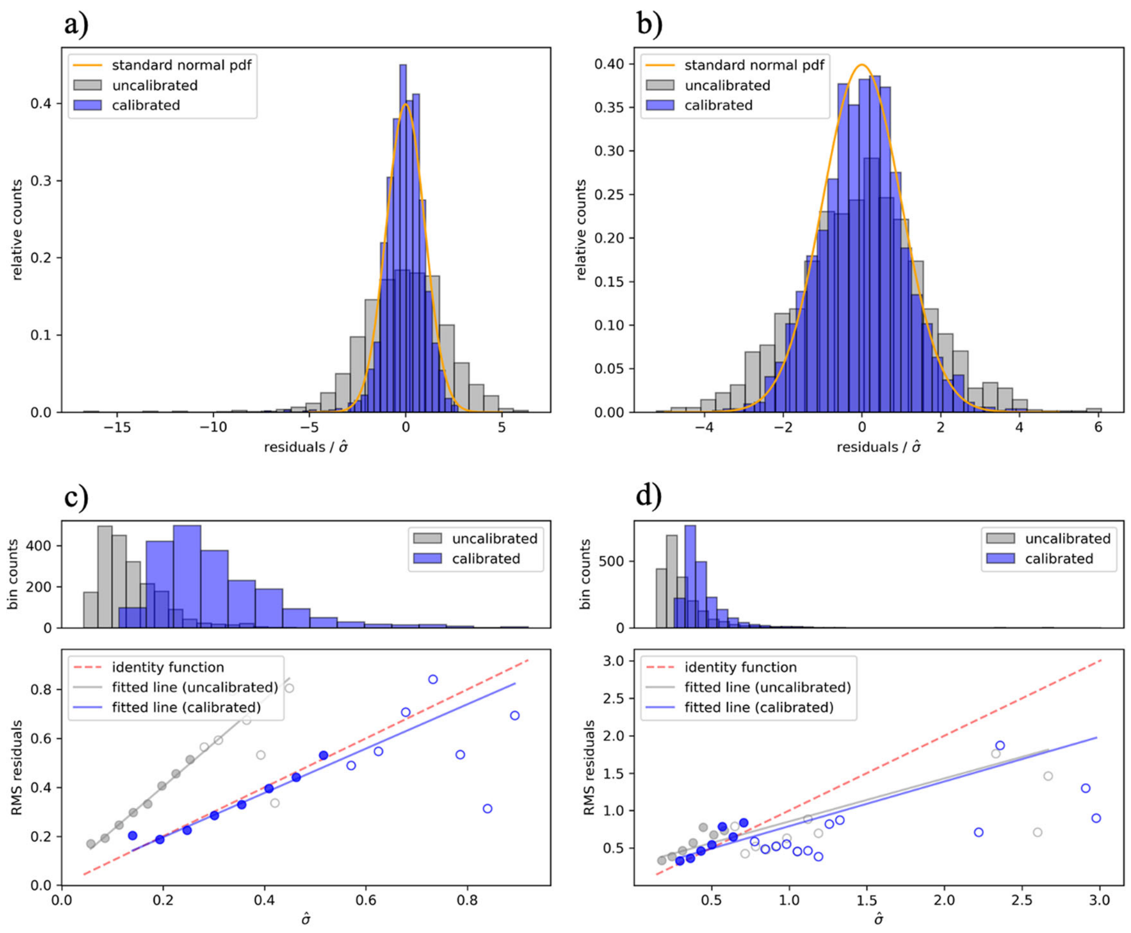

Fig. 2 Calibrated and uncalibrated r-statistic and RvE plots for GPR and linear ridge regression. Distributions of r-statistic values (top row) and RMS residual vs. $\hat{\sigma}$ plots (bottom row) for Gaussian process regression (panels **a** and **c**) and linear ridge regression (panels **b** and **d**), using the diffusion dataset, shown with both uncalibrated and calibrated $\hat{\sigma}$ values. Markers which are not filled in represent bins with fewer than 30 points. Statistics for each plot are summarized in Table 1.

suggesting that the $\hat{\sigma}_{GPR}$ values may be overestimating the true uncertainty.

When interpreting this comparison of $\hat{\sigma}_{GPR}$ to our method based on the bootstrap ensemble (see Fig. 2), it is important to also consider the prediction error, since in Fig. 3, the predictions are being made by a single GPR model trained on the entire training set, rather than an ensemble of models trained on bootstrap samples of the training set. Note in Table 1 that the RMSE of the single model is somewhat lower than that of the bootstrap predictions (0.269 vs. 0.310). To address this issue, the right panels of Fig. 3 show the results of our calibration method for predictions made by a single GPR model, but with uncertainty estimates still obtained as the standard deviation of the predictions of a bootstrap ensemble, as in Fig. 2. This slightly different method of developing calibrated error estimates appears to be less accurate for the error estimates in this particular case, although more study would be needed to determine if this is a general effect. However, the approach is still more accurate than the Bayesian uncertainty estimates, while yielding identical prediction accuracy. It is also important to note that the superiority of our method is confined to the setting in which we hope to obtain accurate uncertainty estimates for predictions on a test set we know is similar to our training set. When test-set values are substantially different from a training set, the $\hat{\sigma}_{cal}$ will likely underestimate the true uncertainty, perhaps severely. In that setting, $\hat{\sigma}_{GPR}$ may do a better job of indicating a higher degree of uncertainty, although $\hat{\sigma}_{GPR}$ converges asymptotically to zero (or a constant) for features far from the training data due to the kernel becoming zero. However, for our present goal of obtaining accurate uncertainty estimates for predictions of reasonably similar data, this test demonstrates that our approach is significantly more accurate than the widely used $\hat{\sigma}_{GPR}$ values, even when they are calibrated.

Impact of noise on calibration performance

In Fig. 4, we show the results of adding varying amounts of noise to the synthetic dataset, and then using our calibration method with a random forest. We generated seven noisy training and test sets by adding Gaussian noise with mean 0 and standard deviation equal to 0.1, 0.2, 0.3, 0.4, 0.5, 1.0, and 2.0 times the standard deviation of the original training set (cases 0.1, 0.2, 0.3, and 0.5 are shown in Fig. 1 and all the cases are in the Supplementary Information in Supplementary Figs. 23–36). For the 0.1 and 0.2 noise cases, our method appears to work similarly well to the no-noise case. For the 0.3 noise case, the calibrated points on the RMS residual vs. $\hat{\sigma}$ plot appear to diverge somewhat from the identity function line. Then, for the 0.5 noise case, the values of $\hat{\sigma}_{cal}$ have started collapsing to a constant value. For the 1.0 and 2.0 noise cases, the $\hat{\sigma}_{cal}$ values approximately converge to a constant of 1 standard deviation of the noisy dataset. The trend of convergence toward a constant $\hat{\sigma}_{cal}$ is expected, since, as the noise increasingly dominates the underlying true values, this trend in the calibration is the only option to obtain accurate uncertainty estimates. In the limiting case, when the training and test values are normally distributed and totally independent of the features, the best a model can do is to predict the training set mean and

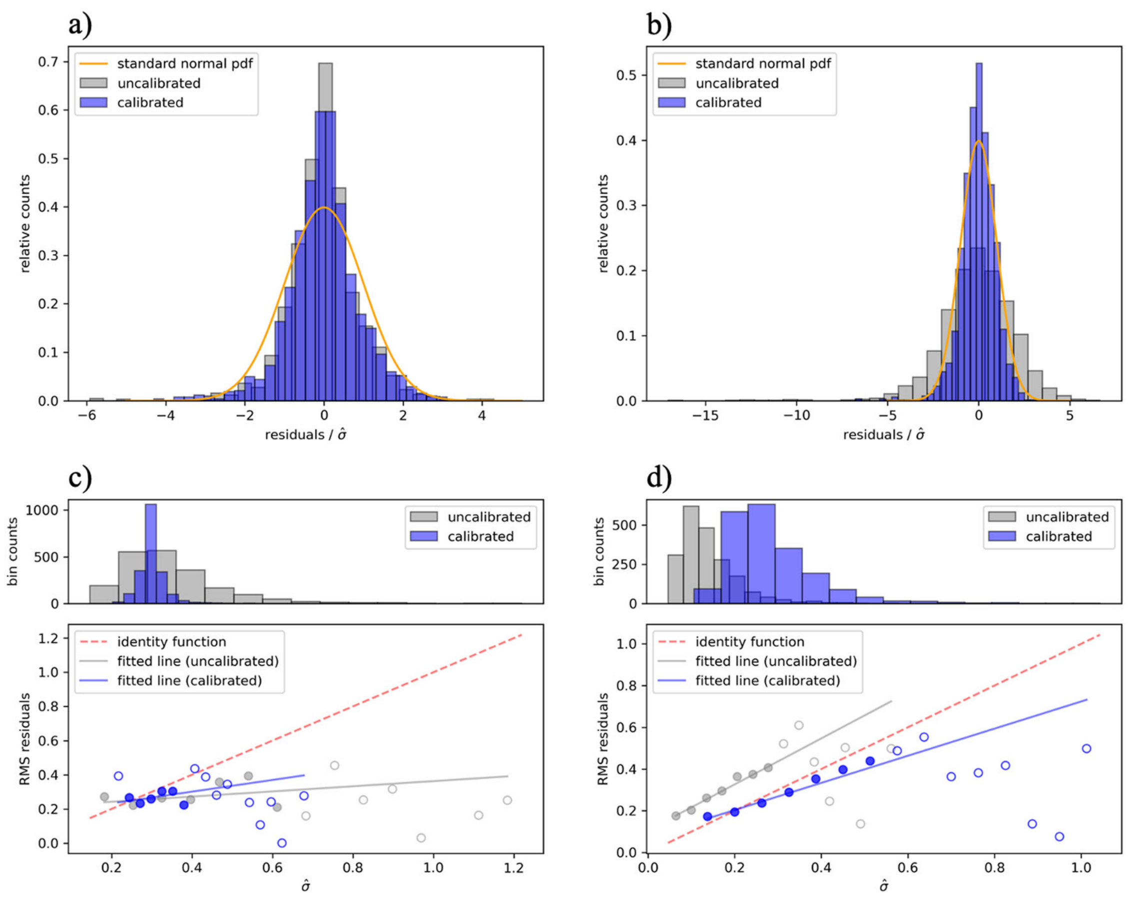

Fig. 3 Bayesian vs. bootstrap calibrated and uncalibrated r-statistic and RvE plots for a single GPR model. Distributions of r-statistic values (top row) and RMS residual vs. $\hat{\sigma}_{GPR}$ plot (bottom row) for Bayesian uncertainty estimates obtained from a single Gaussian process regression model (panels **a** and **c**) and bootstrap uncertainty estimates obtained for the same single Gaussian process regression model (panels **b** and **d**), using the diffusion dataset. Both uncalibrated and calibrated $\hat{\sigma}_{GPR}$ values are shown. Markers which are not filled in represent bins with fewer than 30 points. Statistics for each plot are summarized in Table 1.

estimate the training set standard deviation as the uncertainty. Overall, these results indicate that our calibration method is robust to small and even quite significant (0.2 times the standard deviation of the original training set) amounts of noise and handles very large amounts of noise in the way we would expect. It is somewhat unclear how well the approach works with intermediate amounts of noise of about 0.3 times the standard deviation of the original training set, and additional study is needed for this case.

In Table 1, we present the r-statistic mean and standard deviation, the fitted slopes, y-intercepts, and R^2 values, the calibration factors a and b, and the root-mean-square error (with prediction errors normalized by training set standard deviation) for all the plots in Figs. 1–4. For additional plots and a complete table of these values for all tests we ran, please see Supplementary Table 1 and plots in the Supplementary Information.

DISCUSSION

It is useful to consider the nature of machine learning model errors to better understand why our calibrated bootstrap approach is so effective and some of the limitations of our method. The total expected squared error in a model can be represented as[29]:

$$E\left[\left(F(X)+\epsilon-\hat{F}(X)\right)^2\right]=\left(E\left[\hat{F}(X)\right]-F(X)\right)^2+E\left[\left(\hat{F}(X)-E\left[\hat{F}(X)\right]\right)^2\right]+\sigma^2 \tag{2}$$

Here, the expectation is the average over all possible training datasets of size n, which we can imagine to be randomly sampled from the total possible space of data. The three right-hand side terms from left to right are the model bias squared, model variance, and noise variance, respectively. The model bias (or just bias) is the difference between the expected value of our model averaged over all training set samplings $E[\hat{F}(X)]$ and the underlying true function $F(X)$. The model variance (or just variance) is the squared spread in $\hat{F}(X)$ relative to its average, again taken over all training set samplings. In the absence of noise, the bootstrap approach is an estimate of the variance term in this expression. However, it is an imperfect estimate as the sampling is not all possible training datasets of size n but instead the finite number of bootstrap resampling datasets. We have attempted to use enough bootstrap samples to remove the finite sampling error from being significant, but the nature of the bootstrap datasets is still a source of error. For models with little bias, as is the case for almost all of the models studied here, our approach can be understood to be calibrating the variance estimate. It is perhaps not totally surprising that the approximate sampling from bootstrap gives a variance estimate that is off but strongly correlated with the correct answer, given that it is an approximation to the proper sampling to estimate this correct answer. These considerations also suggest that the present method may be less accurate or fail entirely when bias is a dominant source of error. Such situations would include cases where the predicted test data are far outside the domain of the model. As an example of our

Table 1. Summary of computed values from Figs. 1–4.

		r-stat mean	r-stat stdev	RvE slope	RvE intercept	RvE R^2	a	b	Overall RMSE
Fig. 1									
RF, Synthetic	Uncalibrated	0.043	0.675	0.489	0.046	0.964	0.445	0.070	0.196
	Calibrated	0.060	0.938	1.099	−0.031	0.964			
RF, Diffusion	Uncalibrated	−0.022	0.707	0.660	0.017	0.971	0.647 ± 0.047	0.033 ± 0.025	0.368
	Calibrated	−0.027	0.979	1.010	−0.013	0.970			
RF, Perovskite	Uncalibrated	−0.014	0.766	0.875	−0.023	0.972	0.807 ± 0.038	0.002 ± 0.003	0.377
	Calibrated	−0.019	0.941	1.080	−0.024	0.953			
Fig. 2									
LR, Diffusion	Uncalibrated	0.001	1.574	0.575	0.279	0.561	0.927 ± 0.115	0.154 ± 0.041	0.481
	Calibrated	0.002	1.058	0.597	0.193	0.521			
GPR, Diffusion	Uncalibrated	−0.094	2.182	1.779	0.047	0.963	1.867 ± 0.207	0.048 ± 0.029	0.310
	Calibrated	−0.039	0.963	0.905	0.015	0.929			
Fig. 3									
GPR Bayesian, Diffusion	Uncalibrated	0.001	0.873	0.151	0.212	0.154	0.214 ± 0.122	0.227 ± 0.043	0.269
	Calibrated	−0.001	0.901	0.343	0.164	0.149			
GPR Single Predictor, Diffusion	Uncalibrated	−0.078	2.004	1.101	0.106	0.869	1.545 ± 0.208	0.071 ± 0.031	0.269
	Calibrated	−0.032	0.932	0.650	0.073	0.846			
Fig. 4									
RF, Synthetic, 0.1 noise	Uncalibrated	0.037	0.775	0.394	0.101	0.926	0.394	0.105	0.225
	Calibrated	0.047	0.979	1.001	−0.005	0926			
RF, Synthetic, 0.2 noise	Uncalibrated	0.020	0.914	0.292	0.187	0.914	0.290	0.194	0.288
	Calibrated	0.008	0.977	1.008	−0.009	0.914			
RF, Synthetic, 0.3 noise	Uncalibrated	0.038	1.017	0.279	0.256	0.804	0.253	0.264	0.366
	Calibrated	0.036	1.004	1.103	−0.035	0.804			
RF, Synthetic, 0.5 noise	Uncalibrated	0.039	1.081	0.086	0.451	0.166	0.173	0.405	0.493
	Calibrated	0.026	1.009	0.495	0.250	0.166			

model failing for out-of-domain test data, in Supplementary Figs. 93–96 we show our recalibration method and associated parity plots on the diffusion dataset for different test sets that are outside the training data, which show, as expected, that the model error cannot be accurately predicted. We believe there is an opportunity for further research in this area to construct a combined model domain and uncertainty estimation framework which provides the user information on whether test data points are expected to be inside, outside, or near the boundary of the model domain of applicability, and, if it is within the domain, also provide model predictions and calibrated uncertainty estimates. One possible avenue toward realizing this may be to combine uncertainty predictions from Bayesian methods like GPR to assess the model domain and use the ensemble models detailed throughout this study to provide accurate calibrated uncertainty estimates. Formulation of such a method, if shown to be reliable, would represent a major step forward for reliable ML model predictions for many applications in applied science.

The bulk of the testing in this work was done on materials datasets and we believe that our calibration approach has immediate practical implications for materials scientists employing ML for a variety of applications, including both materials property prediction and iterative materials discovery through active learning. Present predictive models in materials science generally do not have model prediction uncertainties for a given data point (although they often report a single average uncertainty on test data). For those studies that do report uncertainties on each data point, the most commonly used approaches at present to obtain these uncertainties in the materials field are GPR and ensembles from the random forest or boosted trees. Our work shows (i) that these values are likely to be incorrect without calibration and thus may be misleading, and (ii) that the values can be easily corrected with a simple scheme easily adopted by materials domain researchers. Obtaining accurate ML model prediction uncertainties is essential for rational guidance of materials design and informed materials property prediction, e.g., to determine if a material is worth pursuing with costly or time-consuming experiments. A concrete example is to consider searching for new photovoltaic materials with electronic bandgap energy near the desired value of 1.4 eV for high-efficiency single-junction solar cell applications. To be of interest, a researcher might need a material within 0.2 eV of the target value. Therefore, a material with a predicted bandgap of 1.4 ± 0.1 eV is much more sensible to pursue than one with a predicted bandgap of 1.4 ± 1 eV, provided the uncertainties can be trusted.

As a second example, iterative materials discovery methods based on active learning approaches have gained increasing popularity in the past several years. In materials applications, active learning has been used as a design of experiments approach to discover new materials with the desired property (e.g., high entropy alloys with high hardness) in the fewest iteration steps possible[30,31]. Here, an ML model (commonly GPR or an ensemble model like a random forest), is fit to available data with the goal of identifying the most promising next material to explore, where a cost function, such as the expected improvement[32], is used to balance exploration of areas of large uncertainty with the exploitation of regions in the design space that are better predicted by the model. Within the active learning approach, having more accurate uncertainty estimates is expected to directly lead to more efficient convergence to a new promising material. Based on the success of our calibration method on the 10 datasets

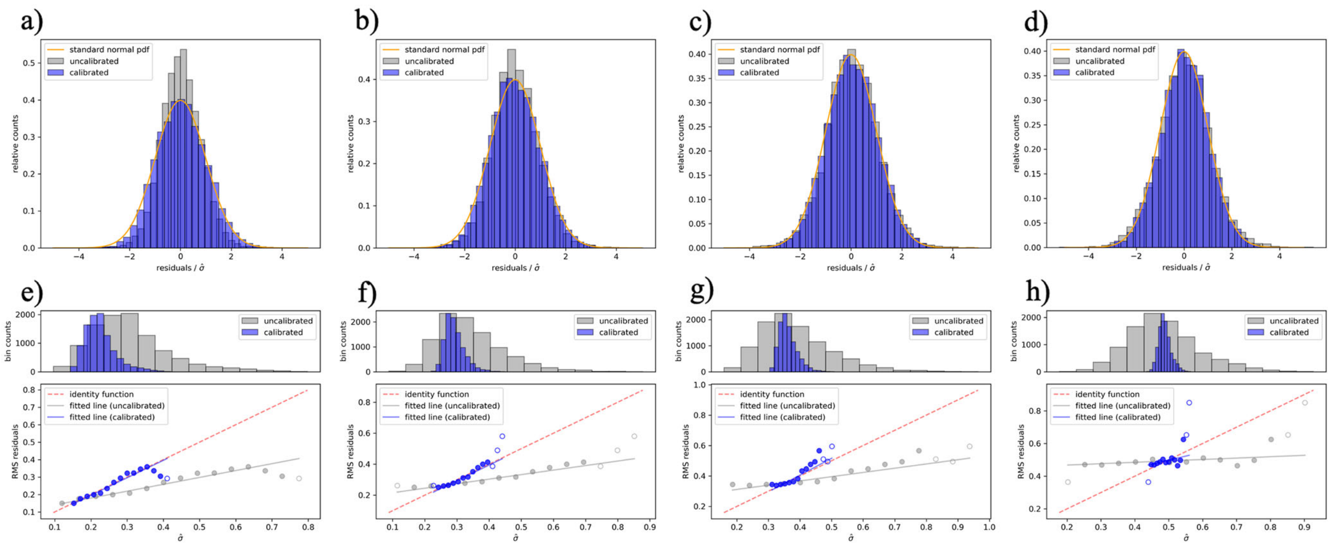

Fig. 4 Calibrated and uncalibrated r-statistic and RvE plots for random forest using synthetic data with noise added. Distributions of r-statistic values and RMS residual vs. $\hat{\sigma}$ plots for random forest, using the synthetic dataset with varying amounts of noise added. Gaussian noise with mean zero and standard deviation equal to 0.1 (panels **a** and **e**), 0.2 (panels **b** and **f**), 0.3 (panels **c** and **g**), and 0.5 (panels **d** and **h**) times the standard deviation of the training set with no noise added. Both uncalibrated and calibrated $\hat{\sigma}$ are shown. Markers which are not filled in represent bins with fewer than 30 points. Statistics for each plot are summarized in Table 1.

(9 materials datasets, 1 synthetic dataset) used in this work, we believe our calibration method offers the materials ML community a useful tool to obtain accurate model uncertainties. We believe that accurate uncertainty quantification is one of the key challenges facing not only the materials research community but the broader community of scientists using ML methods.

Overall, we have demonstrated that across multiple models and datasets, our calibrated bootstrap method of estimating uncertainty is highly accurate. Particularly noteworthy are its exceedingly accurate estimates for random forest across all observed datasets, as demonstrated by the r-statistic and RMS residual vs. $\hat{\sigma}$ plots, and its superior performance for GPR predictions when compared to the Bayesian UQ method typically used in that setting. When our results are taken together, they suggest that the calibrated bootstrap method could be of significant utility for applied scientists in need of better UQ. However, further work is needed to establish its limitations, particularly for predictions on data that is far from the domain of the training data. In addition, the specific approach here is just one of a large family of related methods that can be generated by considering other ensembles besides bootstrap, other uncertainty estimators besides standard deviation (e.g., using confidence intervals), and other calibration approaches, and further exploration of this family of methods could yield additional useful approaches.

METHODS

Models

We used four types of machine learning models to evaluate our method: random forest, Gaussian process regression (GPR), linear ridge regression, and neural networks. Random forest, GPR, and linear ridge regression were implemented using scikit-learn[33]. The neural networks were implemented in Keras using the TensorFlow backend[34,35]. For the scikit-learn models, all values not specified and any explicitly referenced as default values are set to default values used as of January 2021 in scikit-learn version 0.23.2. In particular, all bootstrap ensembles were trained by training each individual model to a sample of n training data points, sampled with replacement from the training set (of size n). For random forest, we used 500 decision trees, a max depth of 30, and default values for all other settings. For GPR, we used as the kernel the sum of the ConstantKernel, the Matern kernel with length scale set to 2.0 and the default value 1.5 of the smoothness parameter nu, and the WhiteKernel with the noise level set to 1. We set alpha, the value added to the diagonal of the kernel matrix to prevent numerical issues, to 10^{-4}, and the n_restarts_optimizer parameter to 30. An ensemble of Keras neural networks was applied to the diffusion dataset only. For this case, the individual neural networks consisted of three layers: an input layer of 20 nodes, a hidden layer with 10 nodes, and an output layer with a single node. The network was fully connected, used rectified linear activation for each layer, and was optimized using the Adam optimizer using mean squared error as the loss function. An ensemble of these neural networks was built using the BaggingRegressor model contained in scikit-learn, where each estimator in the BaggingRegressor was made using the KerasRegressor method in Keras, which provides a means to have scikit-learn-like functionality applied to Keras models. When constructing the bootstrap ensemble of GPR models, we used 200 models to keep the total computation time tractable. For linear ridge regression, we used scikit-learn default values, and used 500 models to construct the bootstrap ensembles. For neural networks, only 25 models were used to keep the computation time tractable.

Datasets

We used ten total datasets to evaluate our methods, but the bulk of our analysis was concentrated on just three of these datasets. Here, we provide an overview of the three datasets which we evaluated in the most detail. First, we used a set of synthetic data based on a method proposed by Friedman[27]. There were five features x_0 through x_4 for each point, each drawn uniformly at random from the interval [0,0.5]. Then, y-values were generated for each point using the function:

$$y = 30\sin(4\pi x_0 x_1) + 20(x_2 - 0.5)^2 + 10x_3 + 5x_4 \tag{3}$$

We used the above process to generate both a training set of 500 points and a test set of 10,000 points.

Next, we used two physical datasets from the materials science community. The first dataset (referred in this work as the diffusion dataset) is a computed database of impurity diffusion activation energies for 15 pure metal hosts and 408 host-impurity pairs[26]. The second dataset (referred in this work as the perovskite dataset) is a computed database of perovskite oxide thermodynamic phase stabilities[28].

In addition to the above datasets, we performed additional select tests on seven more physical datasets from the materials science community. These datasets consist of experimental steel yield strengths (https://citrination.com/datasets/153092/), experimental thermal conductivities (https://www.citrination.com), calculated maximum piezoelectric displacements[36], the calculated saturation magnetization of Heusler compounds (https://citrination.com/datasets/150561/, http://heusleralloys.mint.ua.edu/), calculated bulk moduli of assorted materials[37], calculated electronic bandgaps of double perovskite oxides[38], and experimental superconducting critical temperatures of a large assortment of materials[39]. These datasets were chosen because they are representative computed and experimental materials property databases of a range of physical properties relevant to technological applications from many subfields of materials science. They are publicly available via repositories such as Citrination and Matminer.

Train/test splitting

For all datasets, we use a nested cross-validation approach to obtain the value of $\hat{\sigma}_{\text{cal}}$ and assess the performance of our calibrated UQ models. First, each dataset (synthetic or physical) was split into numerous train and test subsets (we call this the level 1 split). The test data are held out until the end and used to evaluate the performance of the calibrated UQ models. The training data are then subsequently split into training and validation subsets (we call this the level 2 split) and is used to assess the uncalibrated UQ models and obtain the calibration parameters for that training dataset. For the synthetic dataset, we trained models on the entire training set (i.e., there was no level 1 split as test data could be readily generated), and then made predictions, along with uncertainty estimates for those predictions on the entire test set. To obtain calibration scaling factors (see below), we used five-fold cross-validation repeated four times on the training set, for a total of 20 level 2 train/validation splits. For both materials datasets, we generated multiple level 1 80%/20% train/test splits through five-fold cross-validation, and used the predictions accumulated over those splits to evaluate our method. For the diffusion dataset, we used 25 level 1 train/test splits, while for the perovskite dataset, because of its larger size, we used 10 level 1 train/test splits. The different numbers of level 1 train/test splits were chosen to yield reasonable statistics in and reasonable time, which led to smaller numbers of splits for the larger datasets. Within each of the training sets, we again calculated the calibration scaling factors using five-fold cross-validation repeated four times, for a total of 20 level 2 train/validation splits. For interpretability on the plots (described below), we scaled all $\hat{\sigma}$ values and residuals so that they were in units of one standard deviation of the entire training set. After the calibration factors were obtained through cross-validation for each training set, models were trained on the entire training set to make test-set predictions.

r-statistic plots

To create the r-statistic plot for a given model, uncertainty-estimate method, and dataset, we calculate the ratio of the residual and $\hat{\sigma}$ value for each prediction. We plotted these ratios in a histogram with 30 bins, so that the width of the bins was 1/30 of the range from the smallest (most negative) ratio value to the largest (most positive) ratio value. For comparison, we plotted the probability density function of a standard normal distribution and overlaid it on the histogram. In addition, for all r-statistic plots included in the paper, we have included Q–Q plots displaying the same data in the Supplementary Information. In the Q–Q plots, a red line is added to indicate the theoretical quantiles of a standard normal distribution.

RMS residual vs. $\hat{\sigma}$ plots

To create the RMS residual vs. $\hat{\sigma}$ plots for a given model, uncertainty-estimate method, and dataset, we used the residuals and $\hat{\sigma}$ values for all predictions, scaled by the standard deviation of the dataset as described above. For most plots, we used 15 bins for the $\hat{\sigma}$ values, so that the width of each bin was 1/15 of the range from the smallest $\hat{\sigma}$ value to the largest $\hat{\sigma}$ value. However, to ensure that estimates were spread over multiple bins in the range where most uncertainty estimates fell, we decreased this bin size if necessary to ensure that the lowest 90% of the $\hat{\sigma}$ values were spread across at least five bins.

With the $\hat{\sigma}$ values organized into bins, we calculated the root mean square of the residuals corresponding to the $\hat{\sigma}$ values in each bin. We made a scatter plot with one point for each bin, where the horizontal axis represents the $\hat{\sigma}$ value and the vertical axis represents the root mean square of the corresponding residual values. We fit a line to this scatter plot using least-squares linear regression with scikit-learn[33], with the fit weighted by the number of points in each bin. For comparison, we also overlaid a line with a slope of one and a y-intercept of zero. We also included above each of these plots a histogram of the number of data points in each bin to enable assessment of sampling quality.

Calculating calibration factors

We calculated calibration factors to correct $\hat{\sigma}_{uc}$ using a method similar to one applied by Hirschfeld et al.[1], and described briefly in the introduction. Given the set of $\hat{\sigma}$ values and residuals for the cross-validation predictions, we labeled each uncalibrated $\hat{\sigma}$ value as $\hat{\sigma}_{uc}(x)$, and sought to find the corresponding calibrated uncertainty estimates $\hat{\sigma}_{\text{cal}}(x)$ that accurately predict the standard error of predictions. To do so, we assumed that the prediction standard error was linearly related to $\hat{\sigma}_{uc}(x)$, such that for some a and b, we have $\hat{\sigma}_{\text{cal}}(x) = a\hat{\sigma}_{uc}(x) + b$. To find appropriate values of a and b, we found the values that minimized the sum of the negative log-likelihoods that each residual $R(x)$ from cross-validation predictions was drawn from a normal distribution with a mean of 0 and a standard deviation of $a\hat{\sigma}_{uc} + b$. That is, letting D_{cv} be the set of cross-validation data, we solved the optimization problem:

$$a, b = \operatorname{argmin}_{a',b'} \sum_{x,y \in D_{cv}} \ln 2\pi + \ln(a'\hat{\sigma}_{uc}(x) + b')^2 + \frac{R(x)^2}{(a'\hat{\sigma}_{uc}(x) + b')^2} \tag{4}$$

To solve the above problem, we used the Nelder-Mead optimization algorithm as implemented in scipy[40]. Once these optimal values of a and b were obtained in the above manner, we used them to calibrate the $\hat{\sigma}_{uc}$ values for test-set predictions by scaling as $a\hat{\sigma}_{uc}(x) + b$.

Convergence data

For each model and dataset, we made plots to determine the number of bootstrap models necessary for the calibration-factor calculations to converge to a consistent value (see Supplementary Figs. 61–76). In general, the convergence behavior will be dataset- and model-dependent. In addition, for a given model and dataset type, the convergence behavior may also be affected by the chosen model hyperparameters, though our own preliminary tests show the effect of hyperparameters is very small. To calculate calibration factors to assess convergence behavior, we did five-fold cross-validation repeated four times with each entire dataset, and calculated calibration factors through the log-likelihood optimization method described above. For random forest and ridge regression models with all three datasets, we calculated factors using 50, 100, 200, 500, and 1000 bootstrap models. We repeated this calculation 10 times and plotted the mean and standard deviation of these results on a scatter plot (see Supplementary Figs. 61–64, 67–70, and 73–76). Because of computational constraints, we did these calculations for the GPR model only with 50, 100, and 200 bootstrap models, and repeated each calculation five times. Based on the resulting convergence plots, random forest and ridge regression models appeared to converge by 500 models, and GPR appeared to approximately converge by 100 models. Therefore, we expect that our results from using 500 models for random forest and ridge regression and 200 models for GPR are reasonable. Our neural network tests used only 25 models which are likely too few for robust convergence but were enough to show qualitatively that our approach works well for this type of model.

Adding noise

To evaluate how our calibration method responds when dealing with noisy data, we added varying amounts of Gaussian noise to both the training and test sets of our synthetic dataset. We used seven different amounts of noise, all drawn from a normal distribution with a mean of zero, but standard deviations of 0.1, 0.2, 0.3, 0.4, 0.5, 1.0, and 2.0 times the original standard deviation of the training set. We then used the calibration method described above and evaluated our results with the r-statistic and RMS residual vs. $\hat{\sigma}$ plots.

DATA AVAILABILITY

We have made the synthetic, diffusion, and perovskite datasets we used publicly available on GitHub (https://github.com/uw-cmg/ML-error). They can be found in the SI folder in this repository. We have not included the additional datasets we studied as they were not studied as extensively here and are all readily available through the provided references. We have also included csv files with the data from all convergence plots, as well as all test-set residuals, $\hat{\sigma}_{uc}$ values, $\hat{\sigma}_{cal}$ values, and calibration factors we obtained and used to create the r-statistic and RMS residual vs. $\hat{\sigma}$ plots.

CODE AVAILABILITY

We have made the python code used to perform all the calculations and generate all figures publicly available on GitHub in the same repository as the data described above (https://github.com/uw-cmg/ML-error). We have also added the methods in this paper to the Materials Simulation Toolkit for Machine Learning (MAST-ML) toolkit for easy application by future users[41]. The MAST-ML toolkit can be found at https://github.com/uw-cmg/MAST-ML.

Received: 25 August 2021; Accepted: 21 April 2022;
Published online: 20 May 2022

REFERENCES

1. Hirschfeld, L., Swanson, K., Yang, K., Barzilay, R. & Coley, C. W. Uncertainty quantification using neural networks for molecular property prediction. *J. Chem. Inf. Model.* **60**, 3770–3780 (2020).
2. Janet, J. P., Duan, C., Yang, T., Nandy, A. & Kulik, H. J. A quantitative uncertainty metric controls error in neural network-driven chemical discovery. *Chem. Sci.* **10**, 7913–7922 (2019).
3. Liu, R. & Wallqvist, A. Molecular similarity-based domain applicability metric efficiently identifies out-of-domain compounds. *J. Chem. Inf. Model.* **59**, 181–189 (2019).
4. Tran, K. et al. Methods for comparing uncertainty quantifications for material property predictions. *Mach. Learn. Sci. Technol.* **1**, 025006 (2020).
5. Tian, Y. et al. Role of uncertainty estimation in accelerating materials development via active learning. *J. Appl. Phys.* **128**, 014103 (2020).
6. Schwalbe-Koda, D., Tan, A. R. & Gómez-Bombarelli, R. Differentiable sampling of molecular geometries with uncertainty-based adversarial attacks. *Nat. Commun.* **12**, 1–12 (2021).
7. Mueller, T., Kusne, A. G. & Ramprasad, R. in *Reviews in Computational Chemistry*, 186–273 (John Wiley & Sons, 2016).
8. Dietterich, T. G. in *Lecture Notes in Computer Scienc,e* vol.1857, 1–15 (Springer Verlag, 2000).
9. Breiman, L. Random forests. *Mach. Learn* **45**, 5–32 (2001).
10. Wager, S., Hastie, T. & Efron, B. Confidence intervals for random forests: The Jackknife and the Infinitesimal Jackknife. *J. Mach. Learn. Res.* **15**, 1625–1651 (2014).
11. Efron, B. & Gong, G. A leisurely look at the bootstrap, the jackknife, and cross-validation. *Am. Stat.* **37**, 36–48 (1983).
12. Efron, B. & Tibshirani, R. Improvements on cross-validation: the 632+ bootstrap method. *J. Am. Stat. Assoc.* **92**, 548–560 (1997).
13. Kohavi, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. in *Proc. Fourteenth International Joint Conference on Artificial Intelligence* Vol. 14, 1137–1143 (1995).
14. Molinaro, A. M., Simon, R. & Pfeiffer, R. M. Prediction error estimation: a comparison of resampling methods. *Bioinformatics* **21**, 3301–3307 (2005).
15. Wu, C. F. J. Jackknife, bootstrap and other resampling methods in regression analysis. *Ann. Stat.* **14**, 1261–1295 (1986).
16. Sexton, J. & Laake, P. Standard errors for bagged and random forest estimators. *Comput. Stat. Data Anal.* **53**, 801–811 (2009).
17. Efron, B. Jackknife-after-bootstrap standard errors and influence functions. *J. R. Stat. Soc. Ser. B* **54**, 83–111 (1992).
18. Lu, B. & Hardin, J. A unified framework for random forest prediction error estimation. *J. Mach. Learn. Res.* **22**, 1–41 (2021).
19. Morgan, D. & Jacobs, R. Opportunities and challenges for machine learning in materials science. *Annu. Rev. Mater. Res.* **50**, 71–103 (2020).
20. Kuleshov, V., Fenner, N. & Ermon, S. Accurate uncertainties for deep learning using calibrated regression. in *35th International Conference on Machine Learing, ICML 2018*, vol. 6, 4369–4377 (International Machine Learning Society (IMLS), 2018).
21. Platt, J. C. Probabilistic outputs for support vector machines and comparisons to regularized likelihood. *Methods Adv. Large Margin Classif.* **10**, 61–74 (1999).
22. Levi, D., Gispan, L., Giladi, N. & Fetaya, E. Evaluating and calibrating uncertainty prediction in regression tasks. *ArXiv Prepr.* **1905**, 11659 (2019).
23. Busk, J. et al. Calibrated uncertainty for molecular property prediction using ensembles of message passing neural networks. *Mach. Learn. Sci. Technol.* **3**, 015012 (2022).
24. Musil, F., Willatt, M. J., Langovoy, M. A. & Ceriotti, M. Fast and accurate uncertainty estimation in chemical machine learning. *J. Chem. Theory Comput.* **15**, 906–915 (2019).
25. Ling, J., Hutchinson, M., Antono, E., Paradiso, S. & Meredig, B. High-dimensional materials and process optimization using data-driven experimental design with well-calibrated uncertainty estimates. *Integr. Mater. Manuf. Innov.* **6**, 207–217 (2017).
26. Lu, H. J. et al. Error assessment and optimal cross-validation approaches in machine learning applied to impurity diffusion. *Comput. Mater. Sci.* **169**, 109075 (2019).
27. Friedman, J. H. Multivariate adaptive regression splines. *Ann. Stat.* **19**, 1–67 (1991).
28. Li, W., Jacobs, R. & Morgan, D. Predicting the thermodynamic stability of perovskite oxides using machine learning models. *Comput. Mater. Sci.* **150**, 454–463 (2018).
29. James, G., Witten, D., Hastie, T. & Tibshirani, R. *An Introduction to Statistical Learning wth application in R*. (Springer, 2013).
30. Yuan, R. et al. Accelerated discovery of large electrostrains in BaTiO3-based piezoelectrics using active learning. *Adv. Mater.* **30**, 1–8 (2018).
31. Wen, C. et al. Machine learning assisted design of high entropy alloys with desired property. *Acta Mater.* **170**, 109–117 (2019).
32. Jones, D. R., Schonlau, M. & Welch, W. J. Efficient global optimization of expensive black-box functions. *J. Glob. Optim.* **13**, 455–492 (1998).
33. Pedregosa, F. et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res* **12**, 2825–2830 (2011).
34. Chollet, F. Keras. https://keras.io/getting_started/faq/#how-should-i-cite-keras. (2015).
35. Abadi, M. et al. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)* 265–284 https://doi.org/10.1038/nn.3331 (2016).
36. de Jong, M., Chen, W., Geerlings, H., Asta, M. & Persson, K. A. A database to enable discovery and design of piezoelectric materials. *Sci. Data* **2**, 150053 (2015).
37. De Jong, M. et al. Charting the complete elastic properties of inorganic crystalline compounds. *Sci. Data* **2**, 150009 (2015).
38. Pilania, G. et al. Machine learning bandgaps of double perovskites. *Sci. Rep.* **6**, 19375 (2016).
39. Stanev, V. et al. Machine learning modeling of superconducting critical temperature. *npj Comput. Mater* **4**, 1–14 (2018).
40. Virtanen, P. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **17**, 261–272 (2020).
41. Jacobs, R. et al. The Materials Simulation Toolkit for Machine learning (MAST-ML): an automated open source toolkit to accelerate data-driven materials research. *Comput. Mater. Sci.* **176**, 109544 (2020).

ACKNOWLEDGEMENTS

The National Science Foundation provided financial support for G.P. (Award # 1545481), S.D., A.P., J.P.E., and X.Y. (Award # 1636950 and 1636910), and R.J. and D.M. (Award # 1931298). Financial support for A.G. and G.G. was provided by the University of Wisconsin Harvey D. Spangler Professorship. We thank Max Hutchinson, Matthias Rupp, and Logan Ward for many helpful discussions.

AUTHOR CONTRIBUTIONS

G.P., R.J., and D.M. developed the UQ method presented. G.P., S.D., A.P., J.P.E., X.Y., A.G., G.G., Z.L., and R.J. wrote code and analyzed output data. G.P. wrote the first draft of the manuscript. G.P., R.J., and D.M. edited and contributed to later drafts of the manuscript.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41524-022-00794-8.

Correspondence and requests for materials should be addressed to Dane Morgan.