A Implementation Details

A.1 Details of the Base Model

For the NequIP base, we used a latent dimension of 64 and included all irreducible representations of even and odd parity up to $l = 2$.
We used a radial network with a radial cutoff of 4 Å, a trainable Bessel basis of size 8 and two hidden layers of dimension 64 with SiLU nonlinearities.
Gated SiLU- and tanh-based nonlinearities were used for the even and odd features respectively.
A base with 5 interaction blocks was used.
As in the original paper, we used two projection layers. The first one reduces the latent dimension to 32 and the second one reduces it further to 1.
The model was constructed from the official NequIP package built on top of the e3nn library (Geiger et al. 2022).

A.2 Details of the Stochastic Model

To calculate the standard deviations for the forces and energies from the invariant features produced by the NequIP base, two separate fully connected feed-forward neural networks were used. These neural networks have an input layer of dimension 64, two hidden layers of dimension 32 and 16 respectively and an output layer of dimension 1.
The hidden layers have SiLU activation functions and the output layer has the exponential function as the activation function.
The neural network for the force standard deviations maps each invariant feature $\boldsymbol{v}_i$ directly to the standard deviations $\sigma_{\boldsymbol{F}_i}$.
The neural network for the energy standard deviations maps each invariant feature $\boldsymbol{v}_i$ to a scalar latent variable u_i and the standard deviation is then calculated as $\sigma_E = \frac{1}{n}\sum_{i=1}^{n} u_i$.
We do not normalize the force and energy labels and instead rescale the predicted means for both the forces and energies by the root mean square of the force components evaluated on the training data set. We do not rescale the standard deviations.

A.3 Details of the Sampling Algorithm

For all experiments, all models were sampled from a single Markov chain. We always set α to 0.1, β to 0.999 and ϵ to 10^{-3}.
The choice of α and β is oriented at the corresponding values of these hyperparameters in the original AMSGrad algorithm (Reddi, Kale, and Kumar 2018). The stability constant ϵ has to be chosen more carefully. Because

$$\Delta\boldsymbol{\theta}_t = \gamma \boldsymbol{M}_t^{-1} \odot \boldsymbol{v}_t \propto \alpha\sqrt{\frac{2\gamma}{\boldsymbol{M}_t|D|}} \odot \boldsymbol{\mathcal{N}}_t\left(0, I\right),$$

it is advisable to set ϵ large enough that $\alpha\sqrt{\frac{2\gamma}{\epsilon|D|}} << 1$ holds.
On the RMD17 and coupled cluster-level dataset, a batch size of 32 was used and the learning rate γ was exponentially decayed from 10^{-2} to 10^{-5} during the first 10^6 training steps. After step 10^6 the first model was sampled. Afterward, we utilized a cyclical learning rate schedule very similar to the one proposed in (Zhang et al. 2020) $\gamma_i = \frac{\gamma_0}{2}\left(\cos\left(\pi + \frac{i \cdot \pi}{K}\right) + 1\right)$ with $\gamma_0 = 0.001$ and cycle length $K = 50000$ to sample the subsequent models from the same Markov chain. A single model was sampled after each cycle. The mass vector $\boldsymbol{M}$ was kept constant after the first $9 \cdot 10^5$ training steps.

The same sampling procedure was used for the PEDOT dataset except a batch size of 10 was used, the initial learning rate decay and subsequent sampling occurred during the first $5 \cdot 10^5$ training steps and the mass vector $\boldsymbol{M}$ was kept constant after the first 10^5 training steps.

A.4 Details of the Dropout Neural Networks

For the dropout architecture, a single dropout layer was added after the NequIP base in the stochastic model.
The resulting model was optimized using the AMSGrad optimizer with the same batch size and learning rate schedule used during the initial convergence phase of the Markov chain sampling algorithm.
The mean log-likelihoods were used as an objective function.
We used a dropout rate of 1/64 so that on average one activation was dropped per atomic energy contribution.
We found it necessary to use this fairly small dropout rate because higher rates significantly negatively impacted the accuracy of the resulting model, which is consistent with what was found in (Wen and Tadmor 2020).

A.5 Details of the Deep Ensemble

For the deep ensemble, 8 stochastics NequIP models were initialized with different weights by manually setting different pytorch seeds. The model architecture itself was identical to the one used in the Bayesian neural network model. Each model was trained with a batch size of 30 with the AMSGrad optimizer.
The mean log-likelihoods were used as an objective function.
An initial learning rate of 0.01 was chosen which was exponentially decayed to 10^{-5} over $5 \cdot 10^5$ training steps. Every 1000

training steps, the model was evaluated on the validation set. For each of the 8 models, the model weights with the lowest force-MAE on the validation set during the training of that model were used as the final weights of the specific model.

A.6 Evaluation of the Calibration Error and the Error Densities

In the evaluation of the calibration errors and observed error densities, we utilize a bin width $\delta = 0.01$.
To visualize the observed distributions of the components of the errors $(\boldsymbol{F}_i - \boldsymbol{F}_{pred\,i})$ on a test dataset, we set the observed error density as $\rho_{observed}(x) = \frac{f_x}{\delta}$ where f_x is the fraction of the prediction errors of force components falling into the interval $[x, x+\delta)$.
The predicted error density was calculated as

$$\rho_{predicted}(x) = \frac{1}{3 \cdot molsize \cdot |D|} \sum_{i=1}^{3} \sum_{j=1}^{molsize} \sum_{l=1}^{|D|} \rho_{pred}(\boldsymbol{F}_{lji} - \boldsymbol{F}_{pred,\,lji} = x)$$

$$= \frac{1}{3 \cdot molsize \cdot |D|} \sum_{i=1}^{3} \sum_{j=1}^{molsize} \sum_{l=1}^{|D|} \frac{1}{\sigma_{pred,\,lji}\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2_{pred,\,lji}}}$$

where l enumerates the molecular configurations of the test dataset, j enumerates the atoms in the molecule of size $molsize$ and i runs over the individual force components. $\sigma_{pred,\,lji}$ is the predicted standard deviation of corresponding force component.

A.7 Calculation of Densities

To smooth the predicted distribution of several Monte Carlo samples or ensemble models we calculate the final distribution by fitting a normal distribution to the predicted means and variances. The total variance of several Monte Carlo samples or Ensemble models for force components was calculated as

$$\sigma^2_{F_i} = Variance\left(\hat{F}_{i,j}\right) + \frac{1}{k}\sum_{j=1}^{k} \sigma^2_{F_{i,j}},$$

where j enumerates the individual Monte Carlo samples/ ensemble models, $\hat{F}_{i,j}$ is the predicted expectation value for the i-th force component of that particular model and $\sigma_{F_{i,j}}$ the corresponding standard deviation. The variance is calculated over the Monte Carlo samples/ ensemble models. The mean of the predicted distribution was simply calculated as $\hat{F}_i = \frac{1}{k}\sum_{j=1}^{k} \hat{F}_{i,j}$.
The calculation of the final energy distribution was done completely analogously.

B The PEDOT Dataset

The *ab initio* molecular dynamics on PEDOT was carried out using NWChem 6.8 (Aprà et al. 2020), using hybrid DFT: B3LYP (Becke 1993; Perdew, Ernzerhof, and Burke 1996) 6-31G* (Rassolov et al. 1998) with D3(BJ) dispersion corrections (Grimme et al. 2016) for the geometry relaxations. Previous work had indicated that diffuse functions did not significantly change the results. Damping, direct inversion in the iterative subspace (DIIS) and level shifting were turned off, and the quadratic convergence algorithm was used. The DRIVER module was used for structural relaxations, which is a quasi-newton optimization. Its default values were used, with a maximum gradient of 0.00045 and root mean square gradient of 0.00030, and a maximum Cartesian step of 0.00180 and root mean square Cartesian gradient of 0.00120. The molecular dynamics itself was carried out using the stochastic velocity rescaling (SVR) thermostat (Bussi, Donadio, and Parrinello 2007), with a relaxation time parameter (τ) of 25 fs and a timestep of 0.5 fs.

The training set is comprised of 50 randomly sampled configurations of the first 1500 timesteps from both the length 8 and length 12 PEDOTs. Equivalently, the validation set is comprised of 15 randomly sampled configurations of the timesteps 1501-1600 from both the length 8 and length 12 PEDOTs. The timesteps 1601-2000 served as the test sets for the respective lengths. The test set for the length 16 PEDOT contained all 2000 configurations. For each of the PEDOT molecules, we defined the potential energy of the ground state geometry as zero.

C Histogram of Force Predictions

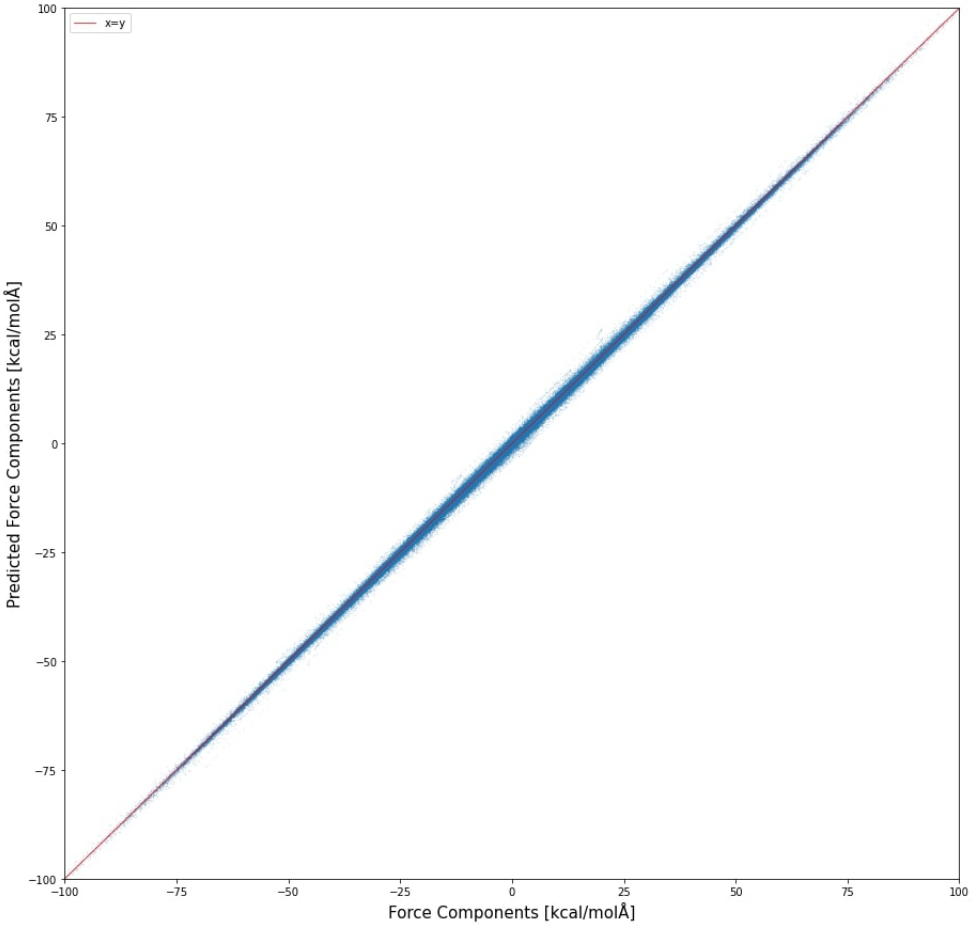

Figure 1: Observed and predicted force components in kcal/(mol·Å) with k=8 Monte Carlo samples on the length 16 PEDOT dataset.

D Comparison to the Original NequIP Model on the RMD17 Dataset

Table 1: Results of the NequIP architecture used as a base model and a single Monte Carlo sample of the Bayesian posterior of the stochastic model on the RMD17 dataset. Both models were generated with identical training, validation and test datasets. The NequIP models were trained according to the procedure specified in (Batzner et al. 2022)

Molecule		Base Model		Proposed Model (k=1)			
		MAE	RMSE	MAE	RMSE	NECE	MLL
Aspirin	Energy	0.060	0.087	0.047	0.068	1.63	0.77
	Forces	0.197	0.308	0.154	0.244	1.03	-1.76
Ethanol	Energy	0.013	0.023	0.010	0.018	6.51	1.72
	Forces	0.078	0.139	0.067	0.117	2.69	-2.16
Uracil	Energy	0.010	0.017	0.014	0.021	4.21	2.07
	Forces	0.068	0.115	0.089	0.145	0.96	0.20
Malonaldehyde	Energy	0.020	0.036	0.016	0.029	4.35	1.04
	Forces	0.131	0.227	0.104	0.178	2.60	-8.68
Salicylic acid	Energy	0.020	0.046	0.025	0.042	4.33	0.95
	Forces	0.098	0.182	0.117	0.194	0.79	-0.30
Naphthalene	Energy	0.010	0.013	0.010	0.013	5.55	2.89
	Forces	0.035	0.055	0.046	0.074	1.25	1.31
Touluene	Energy	0.009	0.012	0.011	0.015	3.32	2.73
	Forces	0.044	0.070	0.055	0.087	1.35	0.92
Benzene	Energy	0.004	0.005	0.004	0.005	6.34	3.88
	Forces	0.008	0.012	0.010	0.016	0.04	3.06

E Additional Comparisons for the Dropout Model and Proposed BNN Model on the RMD17 Dataset

Table 2: Extended results of the dropout model and the proposed BNN model on the RMD17 dataset. Both models were generated with identical training, validation and test datasets. 8 Monte Carlo samples were used for each model.

Molecule		Dropout			Proposed Model		
		MAE	RMSE	MLL	MAE	RMSE	MLL
Aspirin	Energy	0.072	0.096	0.88	0.045	0.067	1.41
	Forces	0.215	0.340	-1.85	0.144	0.229	0.22
Ethanol	Energy	0.012	0.021	2.61	0.010	0.017	2.99
	Forces	0.078	0.147	0.74	0.064	0.115	1.03
Uracil	Energy	0.012	0.020	2.63	0.013	0.020	2.63
	Forces	0.082	0.141	0.71	0.085	0.138	0.83
Malonaldehyde	Energy	0.021	0.038	1.82	0.016	0.027	2.51
	Forces	0.145	0.259	-0.18	0.099	0.170	0.49
Salicylic acid	Energy	0.022	0.042	1.87	0.022	0.038	2.13
	Forces	0.108	0.199	0.14	0.109	0.182	0.63
Naphthalene	Energy	0.009	0.013	2.55	0.009	0.012	2.74
	Forces	0.044	0.069	1.19	0.043	0.069	1.63
Touluene	Energy	0.011	0.015	2.63	0.010	0.014	2.79
	Forces	0.052	0.084	1.11	0.051	0.082	1.43
Benzene	Energy	0.005	0.006	3.20	0.004	0.005	3.53
	Forces	0.020	0.032	1.50	0.010	0.016	3.06

F Training Times of the Experiments

All training times reported were evaluated on an A100 GPU.

The total training time to generate all 16 Monte Carlo samples for the PEDOT dataset was 138.5 GPU hours. The training times on the RMD17 dataset can be found in Table 3.

Table 3: Total training times on the RMD17 dataset in GPU hours

Dataset	Original NequIP	Dropout Model	Proposed BNN Model
Aspirin	47.3	59.6	74.1
Ethanol	40.7	57.8	82.0
Uracil	113.3	59.8	73.1
Malonaldehyde	44.7	58.2	78.6
Salicylic Acid	41.6	57.4	74.8
Naphthalene	39.7	57.6	79.3
Toluene	43.2	56.9	78.1
Benzene	40.6	57.7	78.9

The total training time to generate all 8 Monte Carlo samples on the coupled cluster-level dataset was 69.2 GPU hours. The training time for training all 8 models of the deep ensemble was 205.4 GPU hours.

It should be noted, that the training runs were optimized for performance at the cost of runtime. Generally, the last half of the training run only results in small improvements for all models and sometimes results in no improvement.
Hence, the training times should not be used to rank the individual algorithms by speed since faster training times might have been possible at little or no cost in performance. Additionally, it is most likely possible to speed up the training times of the BNNs substantially, by first optimizing them classically to a local maximum of the posterior and then generating Monte Carlo samples via the cyclical procedure discussed in Appendix A.3.

References

Aprà, E.; Bylaska, E. J.; de Jong, W. A.; Govind, N.; Kowalski, K.; Straatsma, T. P.; Valiev, M.; van Dam, H. J. J.; Alexeev, Y.; Anchell, J.; Anisimov, V.; Aquino, F. W.; Atta-Fynn, R.; Autschbach, J.; Bauman, N. P.; Becca, J. C.; Bernholdt, D. E.; Bhaskaran-Nair, K.; Bogatko, S.; Borowski, P.; Boschen, J.; Brabec, J.; Bruner, A.; Cauët, E.; Chen, Y.; Chuev, G. N.; Cramer, C. J.; Daily, J.; Deegan, M. J. O.; Dunning, T. H.; Dupuis, M.; Dyall, K. G.; Fann, G. I.; Fischer, S. A.; Fonari, A.; Früchtl, H.; Gagliardi, L.; Garza, J.; Gawande, N.; Ghosh, S.; Glaesemann, K.; Götz, A. W.; Hammond, J.; Helms, V.; Hermes, E. D.; Hirao, K.; Hirata, S.; Jacquelin, M.; Jensen, L.; Johnson, B. G.; Jónsson, H.; Kendall, R. A.; Klemm, M.; Kobayashi, R.; Konkov, V.; Krishnamoorthy, S.; Krishnan, M.; Lin, Z.; Lins, R. D.; Littlefield, R. J.; Logsdail, A. J.; Lopata, K.; Ma, W.; Marenich, A. V.; Martin del Campo, J.; Mejia-Rodriguez, D.; Moore, J. E.; Mullin, J. M.; Nakajima, T.; Nascimento, D. R.; Nichols, J. A.; Nichols, P. J.; Nieplocha, J.; Otero-de-la Roza, A.; Palmer, B.; Panyala, A.; Pirojsirikul, T.; Peng, B.; Peverati, R.; Pittner, J.; Pollack, L.; Richard, R. M.; Sadayappan, P.; Schatz, G. C.; Shelton, W. A.; Silverstein, D. W.; Smith, D. M. A.; Soares, T. A.; Song, D.; Swart, M.; Taylor, H. L.; Thomas, G. S.; Tipparaju, V.; Truhlar, D. G.; Tsemekhman, K.; Van Voorhis, T.; Vázquez-Mayagoitia, .; Verma, P.; Villa, O.; Vishnu, A.; Vogiatzis, K. D.; Wang, D.; Weare, J. H.; Williamson, M. J.; Windus, T. L.; Woliński, K.; Wong, A. T.; Wu, Q.; Yang, C.; Yu, Q.; Zacharias, M.; Zhang, Z.; Zhao, Y.; and Harrison, R. J. 2020. NWChem: Past, present, and future. *The Journal of Chemical Physics*, 152(18): 184102.

Batzner, S.; Musaelian, A.; Sun, L.; Geiger, M.; Mailoa, J. P.; Kornbluth, M.; Molinari, N.; Smidt, T. E.; and Kozinsky, B. 2022. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13.

Becke, A. D. 1993. Density-functional thermochemistry. III. The role of exact exchange. *The Journal of Chemical Physics*, 98(7): 5648–5652.

Bussi, G.; Donadio, D.; and Parrinello, M. 2007. Canonical sampling through velocity rescaling. *The Journal of Chemical Physics*, 126(1): 014101.

Geiger, M.; Smidt, T.; M., A.; Miller, B. K.; Boomsma, W.; Dice, B.; Lapchevskyi, K.; Weiler, M.; Tyszkiewicz, M.; Uhrin, M.; Batzner, S.; Madisetti, D.; Frellsen, J.; Jung, N.; Sanborn, S.; jkh; Wen, M.; Rackers, J.; Rød, M.; and Bailey, M. 2022. e3nn/e3nn: 2022-12-12.

Grimme, S.; Hansen, A.; Brandenburg, J. G.; and Bannwarth, C. 2016. Dispersion-Corrected Mean-Field Electronic Structure Methods. *Chemical Reviews*, 116(9): 5105–5154. PMID: 27077966.

Perdew, J. P.; Ernzerhof, M.; and Burke, K. 1996. Rationale for mixing exact exchange with density functional approximations. *The Journal of Chemical Physics*, 105(22): 9982–9985.

Rassolov, V. A.; Pople, J. A.; Ratner, M. A.; and Windus, T. L. 1998. 6-31G* basis set for atoms K through Zn. *The Journal of Chemical Physics*, 109(4): 1223–1229.

Reddi, S. J.; Kale, S.; and Kumar, S. 2018. On the Convergence of Adam and Beyond. In *Proceedings of the 6th International Conference on Learning Representations*.

Wen, M.; and Tadmor, E. B. 2020. Uncertainty quantification in molecular simulations with dropout neural network potentials. *npj Computational Materials*, 6: 124.

Zhang, R.; Li, C.; Zhang, J.; Chen, C.; and Wilson, A. G. 2020. Cyclical Stochastic Gradient MCMC for Bayesian Deep Learning. In *International Conference on Learning Representations*.