Supplementary Information

Electronic, redox, and optical property prediction of organic π-conjugated molecules through a hierarchy of machine learning approaches

Vinayak Bhat[1], Parker Sornberger[1], Balaji Sesha Sarath Pokuri[2], Rebekah Duke[1], Baskar Ganapathysubramanian[2], and Chad Risko[1]

[1]Department of Chemistry & Center for Applied Energy Research, University of Kentucky, Lexington, Kentucky 40506, USA

[2]Department of Mechanical Engineering, & Translational AI center, Iowa State University, Ames, Iowa, 50010, USA

Table of Contents

Methods

Here we provide a detailed description of DFT and ML techniques. We note that some details may be redundant with the main text, but we reiterate those for completeness.

Dataset

The curated dataset used in this study is derived from the OCELOT (Organic Crystals in Electronic and Light-Oriented Technologies) database of DFT computed properties for organic, π-conjugated molecules and crystal structures.[1] A detailed description of the methods to generate the high-throughput data is provided elsewhere.[1] In brief, the π-conjugated molecules were obtained from the crystal structures in the OCELOT database using the OCELOT API.[1] Each molecule is fragmented to obtain the largest, contiguous π-conjugated fragment that is then used for the subsequent DFT calculations (see Figure S1). The DFT structure optimizations, single-point energies, and TDDFT evaluations for the low-lying excited states are performed with (ionization potential) IP-tuned LC-ωHPBE functionals derived for each distinct molecule and the Def2SVP basis set.[2-4] Entries that do not contain all the DFT/TDDFT values or have erroneous values are removed. All calculations are performed in Gaussian 16 Rev. A.03.[5]

ML models and training

ML model training was performed in PyTorch version 1.10 and used Cuda 11.4 for GPU acceleration.[6, 7] A five-fold cross-validation method was implemented instead of a fixed train-test data split for training the models as the dataset, though it contains 25k molecules and 200k energy entries, is small. Moreover, this method provides insights into the trained models' generalizability over the dataset's diversely sampled chemical space. All models, except models with evidential deep learning, were subject to five-fold cross-validation. The performance metrics reported here are the averaged results of five-fold cross-validation and the respective standard deviations. The hyperparameters for each model are tuned with Optuna version 2.10, where the metric R^2 is maximized.[8] The hyperparameters for all models were obtained using only one random 80:20 split of the dataset. The mean squared error (MSE) loss function was used for training all models except the evidential deep learning models. The two-dimensional molecular descriptors and extended connectivity fingerprints (ECFP) that were used as the input features to some models are generated with RDKit 2021.3.5.[9, 10] The two-dimensional descriptors are normalized by first dividing each feature by its maximum absolute value and then fitting each feature to the normal distribution. A list of descriptors can be found in the comma separated-values (CSV) file of SI.

The first-generation models were trained with Scikit-Learn version 0.24.2 with training accelerated by Scikit-learn-intelex version 2021.2.[11] The classical ML algorithms used were ridge regression (RR), support vector machine (SVM), and kernel ridge regression (KRR). Two model sets were generated – one with only molecular descriptors as input features and the other with molecular descriptors and ECFP, where the length of the bit vector of ECFP was tuned along with the other hyperparameters of the model.

Tuning a support vector regression model requires finding the optimal weights w such that they minimize the slack variables ζ and ζ^*, which represent the errors of $f(x_i)$ above and below the target y_i respectively (see equations 1-4). Here, we tuned the kernel f along with C and ϵ. During parameter tuning, we chose

between the linear function kernel, polynomial function kernel, radial basis function kernel, and the sigmoid function kernel.

$$\min_{w,\zeta,\zeta^*}\left(\frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\left(\zeta_i + \zeta_i^*\right)\right)\#(1)$$

$$w^T f(x_i) - y_i \leq \epsilon + \zeta_i\,\#(2)$$

$$y_i - w^T f(x_i) \leq \epsilon + \zeta_{i\,t}^*\#(3)$$

$$\zeta_i,\zeta_i^* \geq 0, i = 1, \ldots, N\,\#(4)$$

For both ridge regressions, the cost C is minimized by tuning the weights, w. Unlike standard linear regression, a regularization term λ is added to the sum of the squared errors. The regularization term was varied between 0.1 and 2 when the ridge regression model hyperparameters were tuned.

Kernel ridge regression keeps the regularization term, λ, but applies a kernel function, f, to the input features (see equations 5-6). During parameter tuning, the kernels were varied using the same selection as for the support vector machine kernels but also included the Laplacian kernel.

$$C = \frac{1}{2}\sum_{i}^{N}\left(y_i - w^T x_i\right)^2 + \frac{1}{2}\lambda||w||^2\#(5)$$

$$C = \sum_{i=1}^{N}\left(y_i - w^T f(x_i)\right)^2 + \lambda||w||^2\#(6)$$

For the second-generation feed-forward networks (FFN), the number of layers in the network, the number of nodes per layer, and the rate of node dropouts were tuned. While tuning the models, we used the ReLU activation function with the Adam optimizer, and the learning rate was held constant at 0.001.[12, 13] Similar to the first-generation models, two model sets were generated – one that used only the molecular descriptors as input and one that used both molecular descriptors and ECFP bit-vectors with their length tuned.

In general, each network is a sequence of fully connected layers $h_1 \ldots h_n$ where h_1 accepts the chemical features, x, as input. These are either the RDKit-created features or those concatenated with the ECFP bit-vectors. The final layer h_n output the estimation of a molecule's electronic property (see equations 7-8). The number of layers was tuned via Optuna, and the models have between one and five layers, depending on the properties they predict and the features that provide their input.

$$h_1 = g_1\left(W_1^T x + b_1\right)\#(7)$$

$$h_n = g_n\left(W_1^T h_{n-1} + b_n\right)\#(8)$$

The third-generation models were created with message-passing neural networks (MPNN) for quantum chemistry.[14] The MPNN utilizes a graph-based representation of molecules where nodes represent atoms and edges represent bonds. The nodes and edges are associated with features like the type of atom and the type of bond on which the MPNN operates to provide a learned representation of the molecule. The learning process for MPNNs involves T message-passing steps. During each step $t < T$, the features h_v^t

associated with a node v are updated using an update function U_t. The information m_t to update the feature is gathered by the message function M_t from features h_w^t of atoms w in the neighborhood of v and associated bonds e_{vw} as described by equations 9 and 10.

$$m_t = \sum_{w \in N(v)} M_t\left(h_v^t, h_w^t, e_{vw}\right) \#(9)$$

$$h_v^{t+1} = U_t\left(h_v^t, m_v^{t+1}\right) \#(10)$$

To fetch the learned representation after T message-passing steps, the set2set model, as described by Gilmer et al., was used.[14] The representation from the MPNN was then passed to 2-layer FFN for molecular property prediction. The molecular graphs for MPNNs were created from SMILES and embedded with atom and bond features using the deep graph library 0.7.2 (DGL) and DGL-Lifesci v0.2.8 Python packages.[15, 16] The atom and bond features used for generating the MPNN input are listed in Table 1 and Table 2, respectively. The tuned hyperparameters for these models were the number of message-passing steps (1-10), the number of set2set steps (1-10), the depth of the set2set layer (1-10), the output feature size of MPNN (32-256), hidden features for edge network (32,256) and the FFN dropout rate (0-1). An Adam optimizer with a learning rate of 0.0001 was implemented. The early stopping algorithm with patience of 20 was used to prevent overfitting.

The fourth-generation models used the same MPNN network as the third-generation. However, the output features from MPNN were concatenated with molecular or DFT descriptors before being passed to the FFN. The hyperparameter tuning process was the same as that of the third-generation models.

The evidential uncertainty for fourth-generation models was evaluated by factoring the code to include an evidential deep learning layer.[17] The evidential deep learning assumes that the prediction (y) of a model is from a Gaussian distribution (N) with unknown mean and variance (μ, σ^2). Accordingly, the mean and variance are represented as –

$$\mu \sim N\left(\gamma, \sigma^2 \upsilon^{-1}\right) \#(11)$$

$$\sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \#(12)$$

where, Γ is the gamma function, and $\gamma, \upsilon, \alpha, \beta$ are parameters. The posterior distribution follows a normal inverse gamma distribution from which the prediction ($E[\mu]$) and epistemic uncertainty ($Var[\mu]$) are computed from the following equations:

$$E[\mu] = \gamma \#(13)$$

$$Var[\mu] = \frac{\beta}{\upsilon(\alpha - 1)} \#(14)$$

The loss function $L(x)$ for training the evidential deep learning model includes a negative likelihood loss $L^{NLL}(x)$ that is responsible for maximizing the model prediction and an evidential loss $L^{EL}(x)$ which minimizes the evidence of errors.

$$L^{EL}(x) = |y - \gamma| \cdot (2\upsilon + \alpha) \#(15)$$

$$L(x) = L^{NLL}(x) + \lambda L^{EL}(x) \#(16)$$

The hyperparameter λ in the loss function was set to 0.2 for training the models with uncertainty quantification.[17] The errors are recalibrated with a Python-based uncertainty toolbox package by minimizing the miscalibration area.[18] The recalibration of uncertainty uses a black-box optimizer to find a standard deviation scalar factor that produces the best recalibration. The hyperparameters of the model MPNN and FFN are the same as those without uncertainty quantification.

Hyperparameter tuning

As mentioned before, the hyperparameters for each model are tuned with Optuna version 2.10, where the metric R^2 is maximized. We used the median pruner to prune the trials with default parameters of no warmup steps and five startup trials. We performed 20 trials, excluding the pruned trials, to obtain the final hyperparameters for five-fold cross-validation. We do not find any trends in the optimized hyperparameters. All the best-trained ML models for every property from each generation are publicly available with the hyperparameters in the *params.json* file. The final parameters for some of the models are also provided in Tables S6-S11.

Tables

Table S1. Statistics of the *OCELOT chromophore v1* dataset. All values are in eV.

Property	Minimum	Maximum	Mean	Standard deviation
HOMO	-11.109	-4.428	-7.312	0.672
LUMO	-4.221	2.306	-0.622	0.696
HL	3.249	11.372	6.689	1.012
VIE	4.422	11.136	7.527	0.635
AIE	4.268	10.892	7.319	0.624
CR1	0.001	0.835	0.208	0.096
CR2	0.002	0.872	0.212	0.1
HR	0.051	1	0.421	0.184
VEA	-4.203	2.203	-0.445	0.681
AEA	-4.386	1.813	-0.692	0.664
AR1	0.013	0.956	0.247	0.096
AR2	0.025	0.758	0.238	0.087
ER	0.073	1	0.485	0.175
S0S1	1.003	6.31	3.569	0.643
S0T1	0.009	4.442	2.462	0.548

Table S2. The R^2 values for ML model trained on a single property, and the unified ML model trained to predict multiple properties.

Model	Epochs	VEA	VIE	S0S1	S0T1	HR
Single	500	0.91	0.82	0.72	0.81	0.37
Unified	500	0.88	0.80	-	-	-
Unified	500	0.84	0.76	0..67	0.74	-
Unified	500	0.84	0.75	0.66	0.73	0.27
Unified	1000	0.84	0.76	0.68	0.74	0.28

Table S3. Performance metrics computed for first-generation ML models. The MAE is reported in eV for all models. The values are averaged over three independently trained models. The input features for these models are the molecular features and ECFP. The values are averaged over five-fold cross-validation models.

Property	RR		SVM		KRR	
	R^2	MAE	R^2	MAE	R^2	MAE
HOMO	0.49±0.007	0.355±0.005	0.51±0.007	0.345±0.005	0.51±0.017	0.349±0.004
LUMO	0.60±0.007	0.338±0.004	0.64±0.006	0.315±0.002	0.64±0.013	0.320±0.005
H-L	0.39±0.011	0.594±0.007	0.41±0.007	0.579±0.004	0.42±0.012	0.578±0.008
VIE	0.70±0.007	0.262±0.003	0.74±0.005	0.24±0.003	0.73±0.005	0.249±0.003
AIE	0.71±0.006	0.256±0.001	0.75±0.005	0.232±0.003	0.74±0.009	0.242±0.002
CR1	0.23±0.012	0.062±0.001	0.23±0.010	0.063±0.001	0.26±0.013	0.06±0.001
CR2	0.27±0.014	0.062±0.001	0.26±0.016	0.062±0.001	0.31±0.014	0.060±0.001
HR	0.27±0.007	0.119±0.001	0.30±0.005	0.111±0.001	0.31±0.008	0.115±0.001
VEA	0.72±0.009	0.276±0.001	0.78±0.007	0.243±0.003	0.77±0.008	0.249±0.004
AEA	0.73±0.008	0.266±0.002	0.78±0.003	0.234±0.001	0.78±0.006	0.24±0.003
AR1	0.27±0.013	0.063±0.001	0.34±0.013	0.057±0.001	0.35±0.010	0.059±0.001
AR2	0.29±0.005	0.055±0.001	0.36±0.014	0.051±0.001	0.35±0.014	0.052±0.001
ER	0.30±0.014	0.114±0.001	0.28±0.010	0.119±0.001	0.38±0.006	0.107±0.001
S0S1	0.50±0.011	0.345±0.003	0.56±0.013	0.321±0.004	0.57±0.008	0.321±0.003
S0T1	0.53±0.017	0.284±0.004	0.59±0.012	0.260±0.005	0.60±0.005	0.264±0.003

Table S4. Metrics for quantifying uncertainty estimates from the fourth-generation ML model with evidential learning. The metrics miscalibration area, sharpness, and negative log-likelihood (NLL) before and after recalibration are shown.

Property	Miscalibration area		Sharpness		NLL	
	Before	After	Before	After	Before	After
HOMO	0.417	0.022	3.788	0.394	2.017	0.877
LUMO	0.102	0.014	0.441	0.297	0.387	0.462
H-L	0.484	0.046	54.475	0.861	4.366	2.264
VIE	0.129	0.021	0.191	0.295	0.823	0.111
AIE	0.131	0.013	0.148	0.225	0.618	0.003
CR1	0.467	0.017	0.002	0.057	529.693	-1.028
CR2	0.444	0.023	0.007	0.092	226.474	-0.774
HR	0.401	0.026	0.024	0.171	87.775	0.367
VEA	0.375	0.038	0.063	0.430	38.515	0.079
AEA	0.372	0.034	0.073	0.454	33.854	-0.041
AR1	0.464	0.023	0.004	0.082	559.042	-0.765
AR2	0.475	0.031	0.003	0.089	1716.917	-0.621
ER	0.428	0.031	0.015	0.145	174.737	0.273
S0S1	0.012	0.011	0.379	0.373	0.449	0.458
S0T1	0.345	0.028	0.146	0.678	18.477	0.094

Table S5. Performance metrics computed on the random split test dataset for the fourth-generation ML models with the evidential learning layer. The MAE is reported in eV for all models.

Property	R^2	MAE
HOMO	0.58	0.304
LUMO	0.78	0.248
H-L	0.49	0.517
VIE	0.85	0.174
AIE	0.87	0.164
CR1	0.39	0.051
CR2	0.46	0.05
HR	0.46	0.096
VEA	0.93	0.146
AEA	0.94	0.125
AR1	0.47	0.052
AR2	0.48	0.044
ER	0.51	0.089
S0S1	0.74	0.257
S0T1	0.86	0.148

Table S6. The optimized hyperparameters used for the first-generation support vector machine models with ECFP2.

Property	ECFP2 length	Regularization	Epsilon	Kernel
HOMO	3807	1.25	0.172	RBF
LUMO	3789	1.72	0.143	RBF
HL	2817	1.86	0.587	Polynomial
VIE	3764	1.84	0.177	RBF
AIE	3429	1.99	0.0897	Polynomial
CR1	683	1.50	0.0699	RBF
CR2	3787	1.08	0.0568	Polynomial
HR	1735	0.941	0.129	Polynomial
VEA	2384	1.85	0.0500	RBF
AEA	1292	1.96	0.0931	Polynomial
AR1	1587	1.89	0.0357	RBF
AR2	1135	1.59	0.0421	Polynomial
ER	136	1.99	0.0229	RBF
S0S1	2621	1.96	0.103	Polynomial
S0T1	3698	1.83	0.0195	Polynomial

Table S7. The optimized parameters for first-generation support vector machine models without ECFP2.

Property	Regularization	Epsilon	Kernel
HOMO	1.99	0.335	Polynomial
LUMO	1.99	0.102	Polynomial
HL	1.99	0.511	RBF
VIE	1.99	0.101	Polynomial
AIE	1.99	0.0317	Polynomial
CR1	0.906	0.0628	Linear
CR2	0.329	0.0827	RBF
HR	1.11	0.0512	Polynomial
VEA	1.99	0.0302	Polynomial
AEA	1,99	0.0882	Polynomial
AR1	1.27	0.0131	RBF
AR2	0.963	0.0309	RBF
ER	1.12	0.171	Linear
S0S1	1.99	0.0845	Polynomial
S0T1	1.99	0.0859	Polynomial

Table S8. The optimized hyperparameters used for the second-generation models without ECFP2.

Property	Dropout	Hidden nodes	Layers
HOMO	0.45	1405	1
LUMO	0.54	300	1
HL	0.50	337	2
VIE	0.62	3774	1
AIE	0.15	2740	1
CR1	0.47	3915	1
CR2	0.71	1590	1
HR	0.73	3331	1
VEA	0.09	3758	2
AEA	0.09	1626	3
AR1	0.51	3774	1
AR2	0.40	1655	1
ER	0.32	3926	2
S0S1	0.44	3897	1
S0T1	0.17	2470	2

Table S9. The optimized hyperparameters used for the second-generation models with ECFP2.

Property	ECFP2 length	Dropout	Hidden nodes	Layers
HOMO	696	0.13	2080	3
LUMO	1661	0.27	3842	1
HL	1296	0.71	1547	1
VIE	616	0.78	2980	1
AIE	903	0.58	2141	1
CR1	1994	0.72	1170	1
CR2	408	0.90	3453	1
HR	1607	0.52	1970	1
VEA	1150	0.38	2620	1
AEA	1696	0.34	2999	1
AR1	1153	0.65	2133	1
AR2	1756	0.33	1560	5
ER	1060	0.56	3886	1
S0S1	1084	0.43	1522	1
S0T1	1359	0.36	3033	1

Table S10. The optimized hyperparameters used for the third-generation models.

Property	FFN Droupout	Edge network size	MPNN output size	Number of set2set layer	Number of message-passing steps	Number of set2set steps
HOMO	0.92	83	240	8	5	10
LUMO	0.44	153	39	5	10	5
HL	0.96	139	45	4	9	5
VIE	0.57	102	173	9	5	1
AIE	0.45	128	221	6	8	5
CR1	0.70	65	195	10	10	4
CR2	0.12	176	75	6	10	7
HR	0.09	32	124	2	2	10
VEA	0.17	235	200	1	5	9
AEA	0.11	208	184	7	6	9
AR1	1.00	163	161	5	8	6
AR2	0.29	123	114	5	4	3
ER	0.43	125	249	2	9	10
S0S1	0.25	223	163	3	9	9
S0T1	0.75	195	234	1	7	4

Table S11. The optimized hyperparameters used for the fourth-generation models

Property	FFN Droupout	Edge network size	MPNN output size	Number of set2set layer	Number of message-passing steps	Number of set2set steps
HOMO	0.74	164	39	3	9	4
LUMO	0.38	47	69	5	10	1
HL	0.82	227	42	2	10	1
VIE	0.56	122	134	7	10	5
AIE	0.85	206	67	10	5	6
CR1	0.98	244	76	9	9	7
CR2	0.90	93	67	2	7	9
HR	0.08	127	168	4	9	1
VEA	0.72	243	131	1	6	5
AEA	0.69	78	227	5	8	7
AR1	0.81	82	241	5	9	6
AR2	0.40	36	224	3	6	7
ER	0.08	181	114	4	6	3
S0S1	0.31	190	75	8	8	8
S0T1	0.54	161	165	6	9	6

Figures

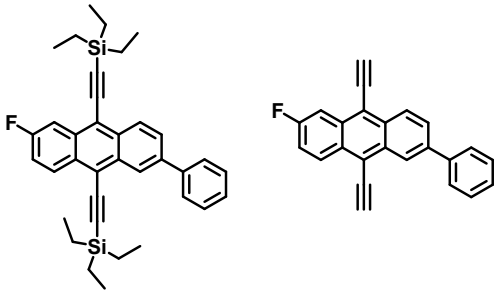

Figure S1. Schematic showing a molecule entry in the OCELOT database (left) and the corresponding fragment (right) obtained using OCELOT API. The *OCELOT chromophore v1* dataset contains DFT computed electronic, redox and optical properties of the fragments.

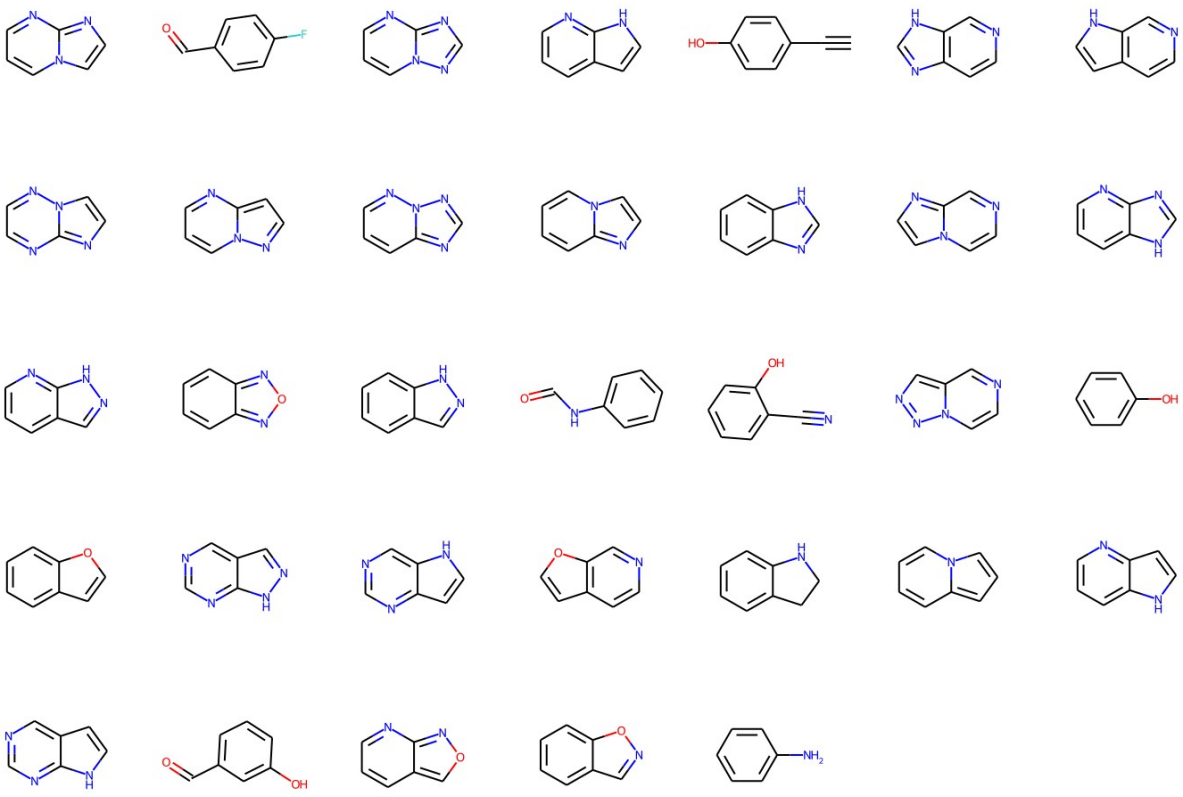

Figure S2. The 33 molecules that are present in both QM9 and OCELOT chromophore v1 dataset.

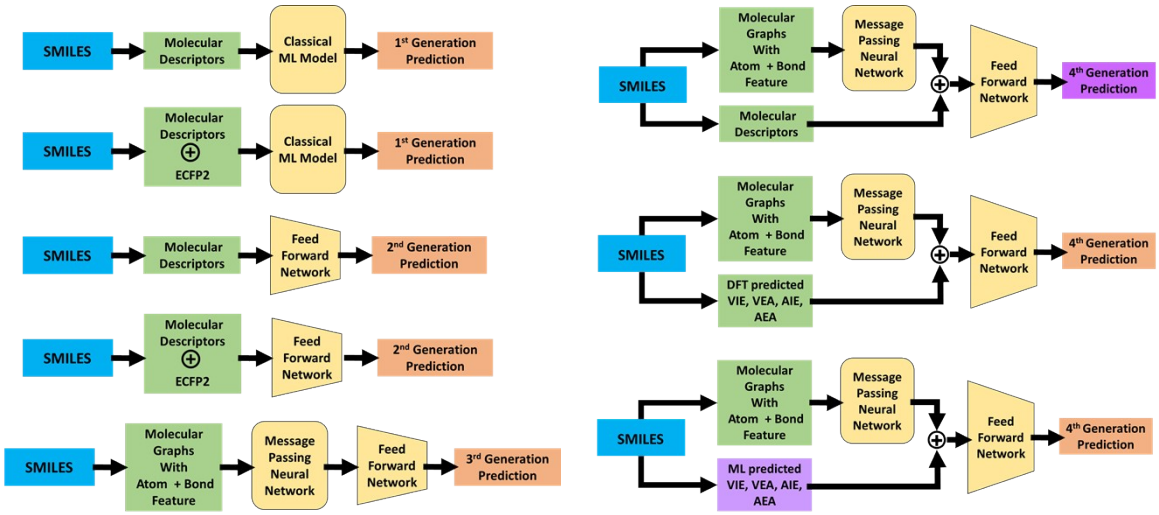

Figure S3: Schematic representation of all ML pipelines explored in the investigation. The input is a SMILES representation (blue) of a molecule from which the input features (green) for the ML models (yellow) are generated for predicting the electronic property (orange/purple). Output from pretrained fourth-generation models (purple) is used as input features for the ML pipeline on the bottom right.

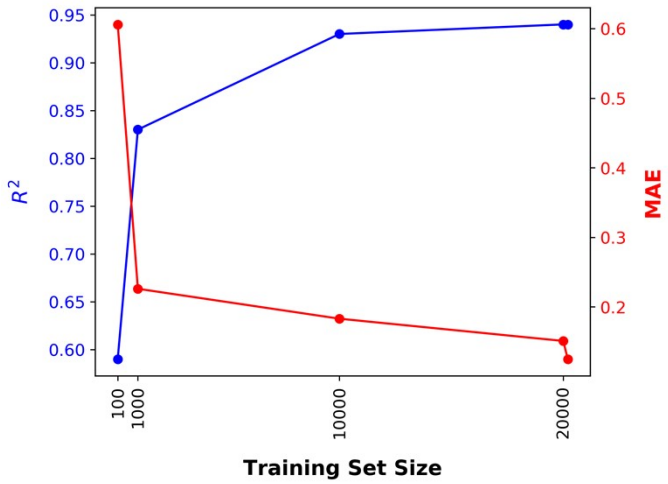

Figure S4. Learning rate for the fourth-generation ML model with molecular descriptors as the concatenated feature for prediction of AEA.

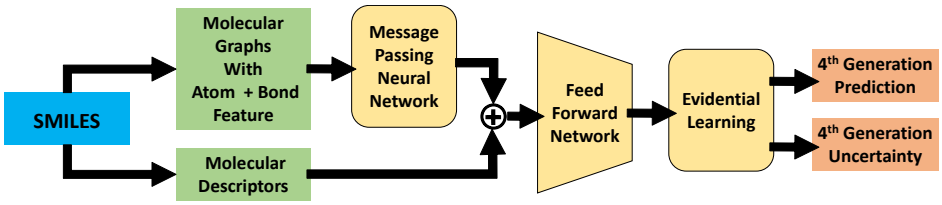

Figure S5. Schematic representation of the architecture of the fourth-generation model with evidential uncertainty layer to predict the uncertainty for a prediction.

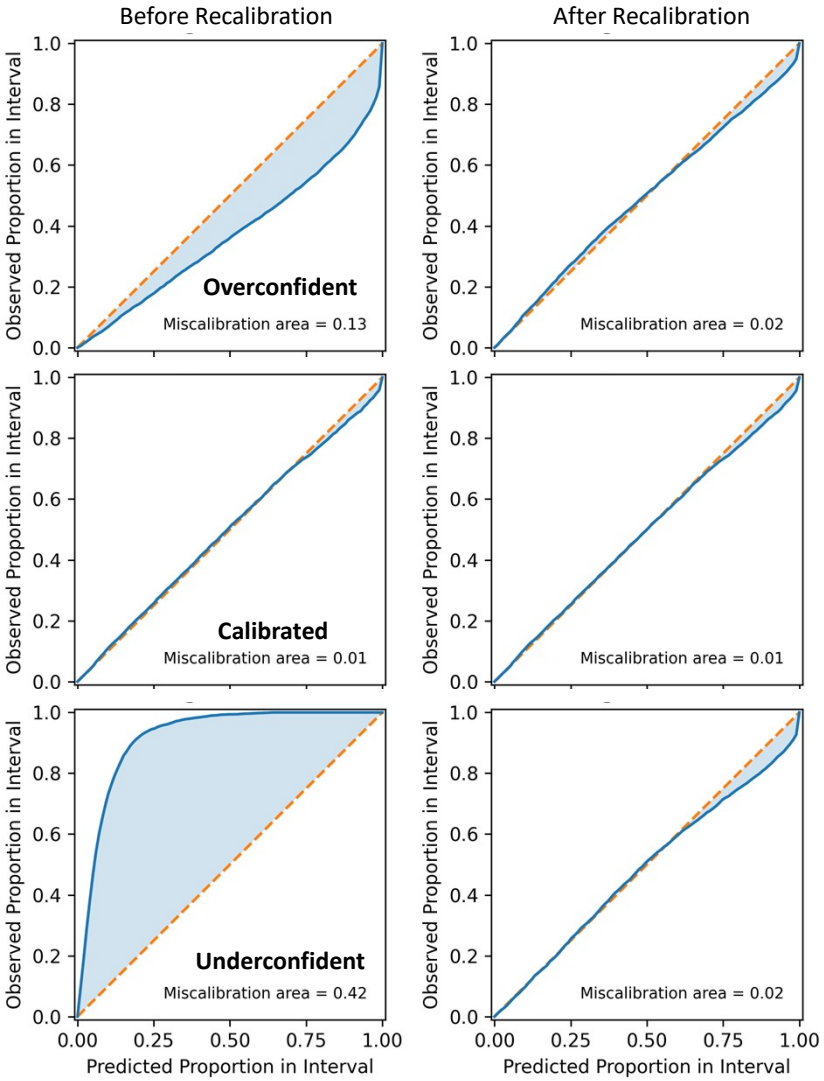

Figure S6. Select calibration curves for uncertainty estimates from the fourth-generation ML model with evidential learning for (top) VIE, (middle) S0S1, and HOMO (bottom). The plots on the left are for uncertainty estimates from the ML model and recalibrated estimates are on right.

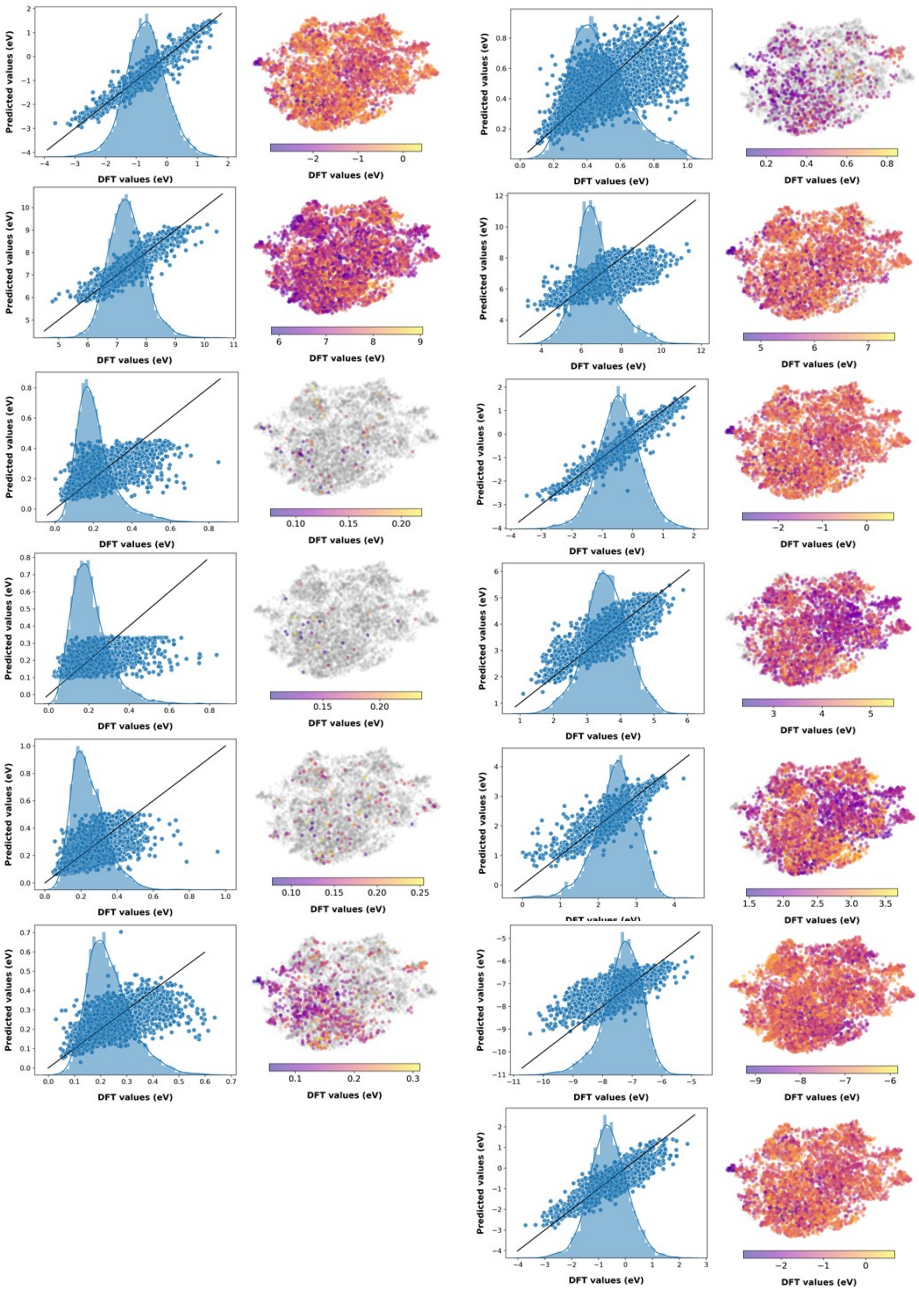

Figure S7. Predictions from the fourth-generation ML model with evidential learning on the test dataset for properties (column 1 top to bottom) AEA, AIE, CR2, CR1, AR1, AR2, and (column 2 top to bottom) ER, H-L, VEA, S0S1, S0T1, HOMO, LUMO. The histograms on the left plot represent the distribution of the corresponding DFT evaluated property in the test dataset. Scatter plots on the right represent the chemical space of the test dataset. The data points where the uncertainty is greater than 10% of the DFT values are gray.

References

1. Q. Ai, V. Bhat, S. M. Ryno, K. Jarolimek, P. Sornberger, A. Smith, M. M. Haley, J. E. Anthony and C. Risko, *The Journal of Chemical Physics*, 2021, **154**, 174705.
2. T. M. Henderson, A. F. Izmaylov, G. Scalmani and G. E. Scuseria, *Journal of Chemical Physics*, 2009, **131**, 044108-044108.
3. F. Weigend and R. Ahlrichs, *Physical chemistry chemical physics : PCCP*, 2005, **7**, 3297-3305.
4. R. Baer, E. Livshits and U. Salzner, *Annual Review of Physical Chemistry*, 2010, **61**, 85-109.
5. G. W. Frisch, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, D. J. M. J. Hratch and Trucks, *Gaussian, Inc., Wallingford, CT*, 2016, DOI: 111.
6. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein and L. Antiga, *Advances in neural information processing systems*, 2019, **32**.
7. J. Nickolls, I. Buck, M. Garland and K. Skadron, *Queue*, 2008, **6**, 40-53.
8. A. Takuya, S. Shotaro, Y. Toshihiko, O. Takeru and K. Masanori, *Journal*, 2019, DOI: 10.1145/3292500.3330701, 2623–2631.
9. G. Landrum, *Components*, 2011.
10. D. Rogers and M. Hahn, *Journal of Chemical Information and Modeling*, 2010, **50**, 742-754.
11. P. Fabian, V. Gaël, G. Alexandre, M. Vincent, T. Bertrand, G. Olivier, B. Mathieu, P. Peter, W. Ron, D. Vincent, V. Jake, P. Alexandre, C. David, B. Matthieu, P. Matthieu and D. Édouard, *J. Mach. Learn. Res.*, 2011, **12**, 2825–2830.
12. A. F. Agarap, *arXiv preprint arXiv:1803.08375*, 2018.
13. D. P. Kingma and J. Ba, *arXiv preprint arXiv:1412.6980*, 2014.
14. J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, *ArXiv*, 2017, **abs/1704.01212**.
15. M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu and Y. Gai, *arXiv preprint arXiv:1909.01315*, 2019.
16. M. Li, J. Zhou, J. Hu, W. Fan, Y. Zhang, Y. Gu and G. Karypis, *ACS Omega*, 2021, **6**, 27233-27238.
17. A. P. Soleimany, A. Amini, S. Goldman, D. Rus, S. N. Bhatia and C. W. Coley, *ACS Central Science*, 2021, **7**, 1356-1367.
18. Y. Chung, I. Char, H. Guo, J. Schneider and W. Neiswanger, *arXiv preprint arXiv:2109.10254*, 2021.