Electronic Supplementary Material (ESI) for Digital Discovery.
This journal is © The Royal Society of Chemistry 2024

Supporting Information for

De Novo Molecule Design Towards Biased Properties via a Deep Generative Framework and Iterative Transfer Learning

Kianoosh Sattari[1], Dawei Li[1], Bhupalee Kalita[4], Yunchao Xie[1], Fatemeh Barmaleki Lighvan[5], Olexandr Isayev[4], and Jian Lin[1, 2, 3*]

[1]Department of Mechanical and Aerospace Engineering,
[2]Department of Electrical Engineering and Computer Science,
[3]Department of Physics and Astronomy,
University of Missouri, Columbia, Missouri 65211, United States
[4]Department of Chemistry, Carnegie Mellon University, Pittsburgh, PA 15213, United States
[5]Department of Biological Sciences, Southern Illinois University Edwardsville, Edwardsville, IL 62026, United States.
[*]E-mail: LinJian@missouri.edu (J. L.)

Supplementary Notes

Supplementary Note 1: Molecule representations by SMILES

Molecules can be represented as undirected graphs. Each atom is considered as a node, and bonds are considered edges connecting the nodes. Introduced by David Weininger in 1988,[1] *SMILES* (Simplified Molecular Input Line Entry System) have been used as a representation approach in computational chemistry and cheminformatics.[2] *SMILES* is a string-based representation method. It is based on a molecular graph theory that defines molecular structures with predefined grammar rules. By following the specific rules, *SMILES* represents the topology of a molecule as a standard molecular graph.[3] It is true that *SMILES* only includes 2D molecular information, but an accurate prediction in the properties of molecules in equilibrium do not need all conformational degrees of freedom of the molecules.[4] In the *SMILES* representation, atomic symbols represent heavy atoms (e.g. C, N, F, P, S and O), "=" and "#" represent bond types (double and triple, respectively), numbers represent rings, and parentheses represent branches within a molecular structure.[1] To reduce the complexity, hydrogen atoms are removed since they can be deduced from the chemistry valence rules.[5]

There are two sources of non-uniqueness in *SMILES*. First, there is an ambiguity about which atom to start the *SMILES*. Second, the choice of whether to include charge information in the resonance structure makes the *SMILES* representations not unique. The canonical *SMILES* following the standard rules defines the atoms and bonds of molecules in a defined order. Thus, there is a unique canonical *SMILES*. It should be noted that the canonical *SMILES* is considered neither a universal nor a global identifier since there are various code rules for generating *SMILES* in each system. We used an open-source cheminformatics suite, RDKit[6] for both the input dataset and the generated samples. It represents unique *SMILES* for a given molecule.

The major challenge of using the *SMILES* representation is that a large fraction of strings generated by a probabilistic model do not represent valid molecules.[2] The generated sequences are either syntactically invalid where the strings do not follow the *SMILES* grammar rule, or semantically invalid where they do not follow the chemistry rule. For instance, the sequence “CCCC(OCC” has an open parenthesis but not a closed one. The sequence “c1ccccc” has a starting point for a ring but no a closing point. They are examples of semantically invalid sequences. The string “CO(C)C”, on the other hand, does not follow the chemistry rule since number of the explicit valence state for oxygen is greater than the one permitted. It is semantically invalid.

Researchers have proposed modified *SMILES* to solve the mentioned validity problems. They include *DeepSMILES*[7] and *SELFIES*.[2] In the *SMILES* syntax, branches are represented by balanced pairs of parentheses, an open parenthesis followed by atoms inside the branch and a close parenthesis to end the branch. For example, Isobutyric acid with two branches is represented as “CC(C)C(=O)O”. Moreover, rings are indicated by a pair of digits with the atoms between the two digits. For example, “c1ccccc1” represents a benzene ring consisting of 6 carbons. Thus, for the rings and branches, *SMILES* uses two symbols that must occur in pairs. *DeepSMILES*, invented by

O'Boyle and Dalke,[7] solves the unbalanced parentheses problem in *SMILES*. It defines the branches by only the close parentheses. Also, it uses only one digit to show the ring. In future, we plan to investigate *DeepSMILES*. Krenn et al. very recently solved the problem at a fundamental level by introducing *SELFIES* (SELF-referencIng Embedded Strings). It is a string-based representation method like *SMILES* but has 100% robustness.[2] In *SELFIES*, all the combinations of the strings are valid. However, the derivation rules are complicated and take lots of effort to be designed for a specific dataset. Also, applying such rules on the final generated sequences does not guarantee that the generator learns the rules and can generate only valid samples. Thus, SELFIES despite being 100% valid, might not be efficient for generative models capable of on-target molecule generation.

The choice of using SMILES-based encoding, particularly the one-hot encoding, was made based on specific considerations aligned with the objectives of our study. While grammar-based VAEs[8] have indeed shown remarkable performance in certain cases, we opted for SMILES-based encoding for the following reasons:

1) Interpretability and Simplicity: The one-hot encoding of SMILES allows for straightforward interpretability. Each character in the SMILES string corresponds to a specific atom or bond, making it easier to trace the encoding process and understand the generated representations.

2) Generality Across Molecular Structures: One-hot encoding of SMILES is agnostic to the complexity of molecular structures and can handle a wide range of chemical compounds, including those with diverse and intricate arrangements. It can also be used to represent very large molecules with many heavy atoms.

3) Computational Efficiency: One-hot encoding is computationally efficient, making it suitable for large-scale datasets and generative tasks where rapid processing is essential.

4) Easy Validity Check: Using RDKit package, it is very easy to check the validity of the generated SMILES.

Supplementary Note 2: Design and training of the reinforced regressional and conditional GAN (RRCGAN)

Architecture of the encoder. The encoder is a convolutional neural network (CNN) with an architecture shown in Fig. S2. It outputs fixed-dimensional latent vectors (6×6×2) that have the most statistically important information from the input discrete one-hot encoded matrices. As shown in Fig. S2, the encoder has two parallel networks that were fed with the same one-hot encoded matrices. Each layer has 4 sequential convolutional blocks that gradually reduce the size of the output of the previous layers that start from 40×27×1 and end with 6×6×1. Each convolutional block consists of a convolutional layer, a leaky ReLU activation function (AF), and a batch normalization layer. The four convolutional blocks are the followed by a convolutional layer and a Tanh AF to output a 6×6×1 vector with continuous number between -1 and 1. The two output of the parallel networks are then concatenated, resulting in the final latent matrices with 6×6×2 dimensions. We hypothesized that one of the two parallel networks in the encoder architecture relates to atoms information and the another relates to bonds information.

Architecture of the decoder. The architecture of the decoder was modified from Google Inception V2, shown in Fig. S3. The decoder converts the latent vectors back to the original SMILES strings (input to the encoder). The advanced architecture of Inception V2 allows for increasing the depth and width of the network to convert the continuous vectors back to original discrete SMILES representation. The original Inception V2 model has been used for classification tasks. Here, the decoder is used for a similar task, as it should come up with a probability for each possible 27 characters for every 40 different positions.

Architecture of the regressor. The structure of the regressor is shown in Fig. S4. It was modified from the Google Inception V2 model. The inception modules are activated by a leaky rectified linear unit (RELU). Some of the modules are followed by an extra max-pooling layer. The output from each module is flattened and then enters a RELU activated dense layer. Eventually, after an extra dense layer, the final output layer with 1 node can output normalized heat capacity.

Architecture of the generator. The architecture of the generator is shown in Fig. S5. To generate a latent vector with a desired heat capacity, the heat capacity is concatenated to a randomly generated noise vector z with a dimension of 128×1, which is then fed into the generator. The first five modules of the network consist of a deconvolutional layer which slides the reshaped input with a stride of 1. The activation function is a RELU. Two modules follow the first five modules. Their number of filters is reduced from 512 to 256 and then to 128. Finally, there is a deconvolutional layer to ensure that only one structure is generated at each iteration. It means that the generator synthesizes one latent vector (6×6×2) for each target value it receives. The tanh activation function is applied to the final layer for outputting continuous numbers ranging from -1 to 1.

Architecture of the discriminator. The architecture of the discriminator, shown in Fig. S6, has two functions. The first function is to distinguish the synthesized latent vectors from the real ones, and the second one is to determine if a generated molecule corresponds to a desired heat capacity. By appending the information of the heat capacity, a single vector rather than a high-dimension tensor is a desired format of the data as an input to the discriminator. Therefore, instead of directly feeding a real or a synthesized latent vector to the discriminator, a latent vector of a structure is first concatenated with the corresponding heat capacity. And then the concatenated vector is fed as the input to the discriminator. The discriminator is trained to distinguish the real latent vectors from the synthesized ones. The discriminator has only one intermediate dense layer with 64 nodes, followed by a RELU activation layer. The output layer is a single-node dense layer activated by the sigmoid function, which forces 0 or 1 output, indicating fake or real ones, respectively.

Training of RRCGAN. The RRCGAN was designed and trained by Google's TensorFlow API. Adam was selected as the optimizer for the encoder, decoder, generator, discriminator, and regressor. The training was performed on the high-performance computing infrastructure provided by Research Computing Support Services and in part by the National Science Foundation under grant number CNS-1429294 at the University of Missouri, Columbia.

Supplementary Note 3. Evaluation metrics

The discrepancies between the DFT calculated $\Delta E_{H\text{-}L}$ of the generated molecules and the desired $\Delta E_{H\text{-}L}$ and predicted $\Delta E_{H\text{-}L}$ by the regressor, respectively, are evaluated by standard statistical metrics including the coefficient of determination (R^2), mean absolute error (MAE), root mean squared error (RMSE) mean squared error (MSE), and relative error (RE). These metrics are provided as follows.

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2} \tag{S1}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y}_i\right| \tag{S2}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2} \tag{S3}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2 \tag{S4}$$

$$RE = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|y_i - \hat{y}_i\right|}{y_i} \tag{S5}$$

where y is the desired $\Delta E_{H\text{-}L}$ or predicted $\Delta E_{H\text{-}L}$ by the regressor, ŷ is the DFT calculated $\Delta E_{H\text{-}L}$ of the generated molecule, and ȳ is the average for all the samples. N is the total number of evaluated molecules.

Supplementary Note 4. Predicting the gap values from structural features

To establish a quantitative relationship, we trained an XGBoost model which takes 18 structural features as input to predict the output $\Delta E_{H\text{-}L}$. These features include molecular weight (MW), number of heavy atoms, MW of heavy atoms, number of hydrogen acceptors, number of Hydrogen donors, number of hetero atoms, number of rotatable bonds, the sum of valence electrons in heavy atoms, number of aromatic rings, number of saturated rings, number of aliphatic rings, number of radical electrons, number of aliphatic carbocycles, number of aliphatic heterocycles, number of aromatic carbocycles, number of aromatic heterocycles, number of saturated carbocycles, and number of saturated heterocycles. The hyperparameters of the well-trained model are shown in Table S4. R^2 of the targeted $\Delta E_{H\text{-}L}$ of the generated samples vs. their predicted values is 0.94 for training and 0.89 for testing, respectively. Fig. S15 shows the rankings of the input features in determining $\Delta E_{H\text{-}L}$. The result shows that the type (aromatic, saturated, and aliphatic) and the number of rings as well as the number of rotatable bonds and the number of hydrogen-bond acceptors are the most important features. Following those, molecular weight of the heavy atoms and the number of valence electrons are ranked as the subsequently important features. This observation agrees well with the results shown in Fig. 4. For instance, all the rings used in the first row of generated molecules in Fig. 4a with low $\Delta E_{H\text{-}L}$ values are unsaturated rings that have some double bonds.

The following 4 structural features of the molecules were used as the input to the XGBoost model for predicting the gap values.

a) Number of Aliphatic Rings*:* The number of aliphatic rings in a molecule refers to the count of cyclic carbon structures that are not part of an aromatic system. The aliphatic rings can vary in size, e.g., three-membered cyclopropane, five-membered cyclopentane, and six-membered cyclohexane rings.

b) Number of Valence Electrons of the Heavy Atoms: The valence electrons of atoms are the electrons involved in chemical bonding. Heavy atoms typically refer to atoms other than hydrogen (H) in a molecule. The number of valence electrons of the heavy atoms in a molecule refers to the sum of the valence electrons contributed by each heavy atom. The valence electrons determine the atom's ability to form bonds with other atoms, and the count of these electrons affects the chemical properties and reactivity of the molecule.

c) Number of Radical Electrons: A radical electron, also known as an unpaired electron, is an electron that exists in an orbital without pairing with other electrons. These unpaired electrons are typically highly reactive and can participate in chemical reactions. The number of radical electrons in a molecule refers to the count of unpaired electrons present within its structure. Presence of the radical electrons can impact the molecule's stability and reactivity.

d) Number of Heteroatoms: Heteroatoms are atoms other than carbon (C) and hydrogen (H) in a molecule. We considered molecules with elements nitrogen (N), oxygen (O), sulfur (S), phosphorus (P), and fluorine (F). The number of heteroatoms in a molecule refers to the count of these non-carbon, non-hydrogen atoms present. $S_{AB} = \frac{c}{a + b - c}$

Supplementary Note 5. Dimension reduction of latent spaces.

First, projection of the latent features is performed using principal component analysis (PCA). PCA derives components formed as a linear combination of the original variables that explain the most variance of the data. The results show that the PCA components of the latent features of the training and generated structures follow close distributions (Fig. S18a-b). We can observe well-defined boundaries of molecules in high and low values of heat capacity for mapped latent features. Second, we performed a non-linear mapping, named Spectral Embedding (SE). SE uses Laplacian Eigenmaps to find a low dimensional representation of the latent features using a spectral decomposition of the graph Laplacian. We can test the hypothesis that the latent features can indeed explain how the model learns the structure-property relationship for catching the physical and chemical laws. Figure S18c and S18d show the spectral embedding values for the training and testing samples, respectively. For the training samples, the boundaries of the four ranges of heat capacity are clear. For the testing molecules, the very high and low values are clustered, but the middle values are mixed.

Supplementary Note 6. Data collection and curation.

From the published PubChemQC database (http://pccdb.org/), we manually downloaded ~135K molecules. Because there are restrictions on downloading the molecules (up to 2000 molecules per file) and we have limited resources to download all the 3 million molecules. Using RDKit, canonical SMILES were extracted to represent the molecules. We picked 40 as the maximum number of characters in the extracted canonical SMILES and excluded any molecules with > 40 characters. We kept molecules with C, O, N, S, P, and F heavy atoms and removed any molecules with other heavy atoms, e.g, Si, resulting in ~132K molecules with their corresponding HOMO-LUMO gap values for model development.

Supplementary Tables:

Table S1. Comparison of the regressor accuracy.

Model	**Descriptors**	**Test MAE (eV, ↓)**	**Test R^2 (↑)**
Base (mean)	-	1.67	0.00
QMCVNet (Ref.[9])	3D geometry descriptors	0.48	0.65
Our predictor	SMILES	0.33	**0.95**
NNs with (Ref.[10])	PaDEL, CDK, and modified distance descriptors	**0.21**	0.91

Table S2. Hyperparameters of RRCGAN.

Models	**Hyperparameters**	**Values**
Encoder-Decoder	Batch size	32
	Epochs	1000
	Adam learning rate	1×10^{-5}
	Adam regularization term (β_1)	0.9
Regressor	Batch size	64
	Epochs	150
	Adam learning rate	5×10^{-7}
	Adam regularization term (β_1)	0.9
Discriminator	Batch size	256
	Epochs	150

	Adam learning rate	2×10^{-5}
	Adam regularization term (β_1)	0.5
Generator	Batch size	256
	Epochs	150
	Adam learning rate	2×10^{-5}
	Adam regularization term (β_1)	0.5

Table S3. Comparison of the existing generative models for targeted molecule design.

Model	**Architecture**	**Targeted property**	**Accuracy evaluation on new generated molecules?**	**One-to-one comparison of targeted vs. reached values?**	**Extrapolate the training dataset?**	**Experimentally validated?**	**Ref.**
RRCGAN	GAN (CNN) + Predictor (CNN)+ AE(CNN) + TL	QC property: ΔE_{H-L}	Yes	Yes	Yes	No	Ours
ReLeaSE	GAN(RNN) + predictor + RL	MT	No	No	Yes	No	[11]
Targeted generation model	GAN(RNN) + predictor + RL	logP: calculated from structure	No	No	Yes	No	[10]
Specialized RNN	RNN + TL	QC property: ΔE_{H-L}	No	No	No	No	[12]
ORGAN	GAN + RL	Druglikeness: calculated from structure	No	No	No	No	[13]
ORGANIC	GAN + RL	MT	No	No	Yes	No	[14]

Note: CNN: convolutional neural network, RNN: recurrent neural network, GAN: generative adversarial network, AE: autoencoder, RL: reinforcement learning, TL: transfer learning, QC: quantum chemical, MT: melting temperature.

Table S4. Statistics of 132626 samples from PubChemQC database.

Min. ΔE_{H-L}	**Ave. ΔE_{H-L}**	**Max. ΔE_{H-L}**	**# of samples > 10**	**% of samples > 10**	**# of samples > 10.5**
1.05	5.94	10.99	461	0.35 %	23

Note: The unit for all the tabulated numbers is (eV).

Table S5. Hyperparameters of the XGBoost model.

Models	**Hyperparameters**	**Values**
XGBoost	max_depth	3
	n_estimators	340
	min_child_weight	31
	learning_rate	0.13
	gamma	0.01
	subsample	0.8

Supplementary Figures:

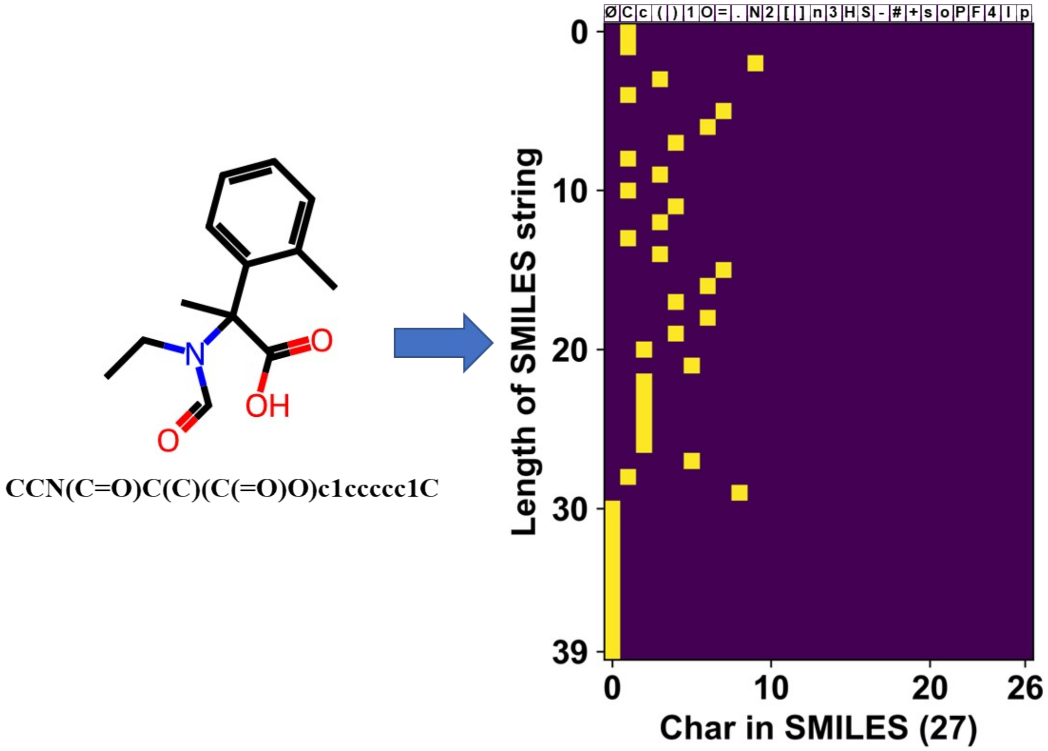

Figure S1. One-hot encoding a representative molecule with SMILES "CCN(C=O)C(C)(C(=O)O)c1ccccc1C" to a 40×27 matrix. The yellow pixels show value of "1" and the rest are "0".

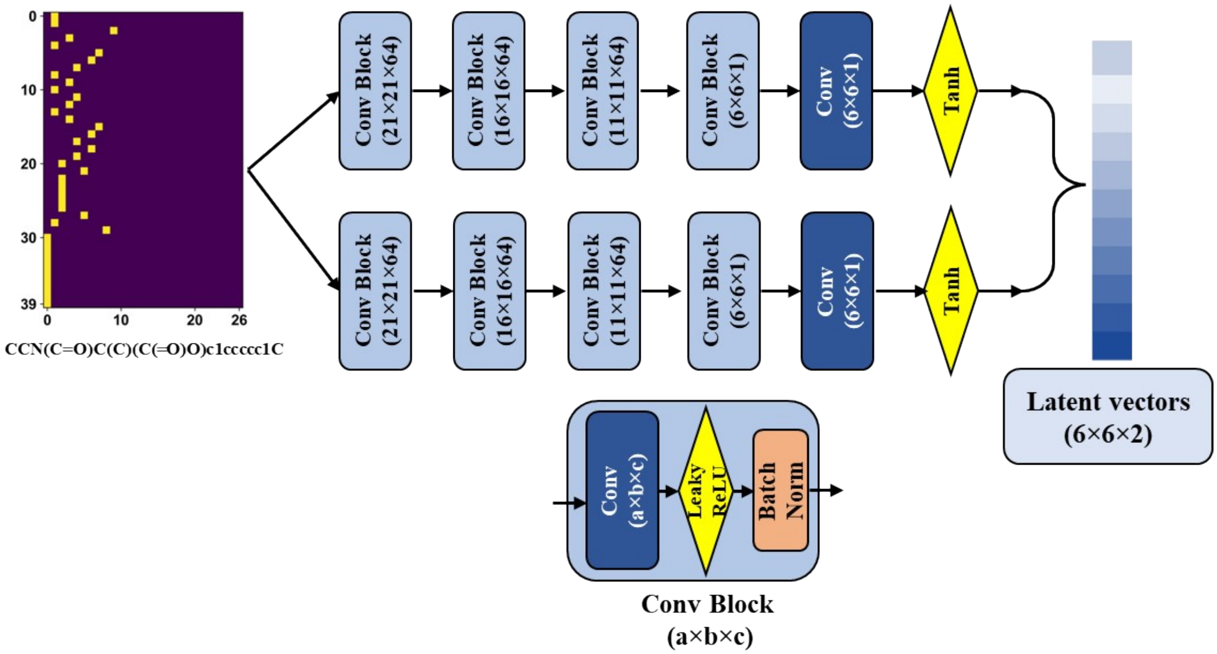

Figure S2. Schematic of the encoder. It takes one-hot encoded SMILES strings (40×27) as input and outputs latent vectors with dimension of (6×6×2).

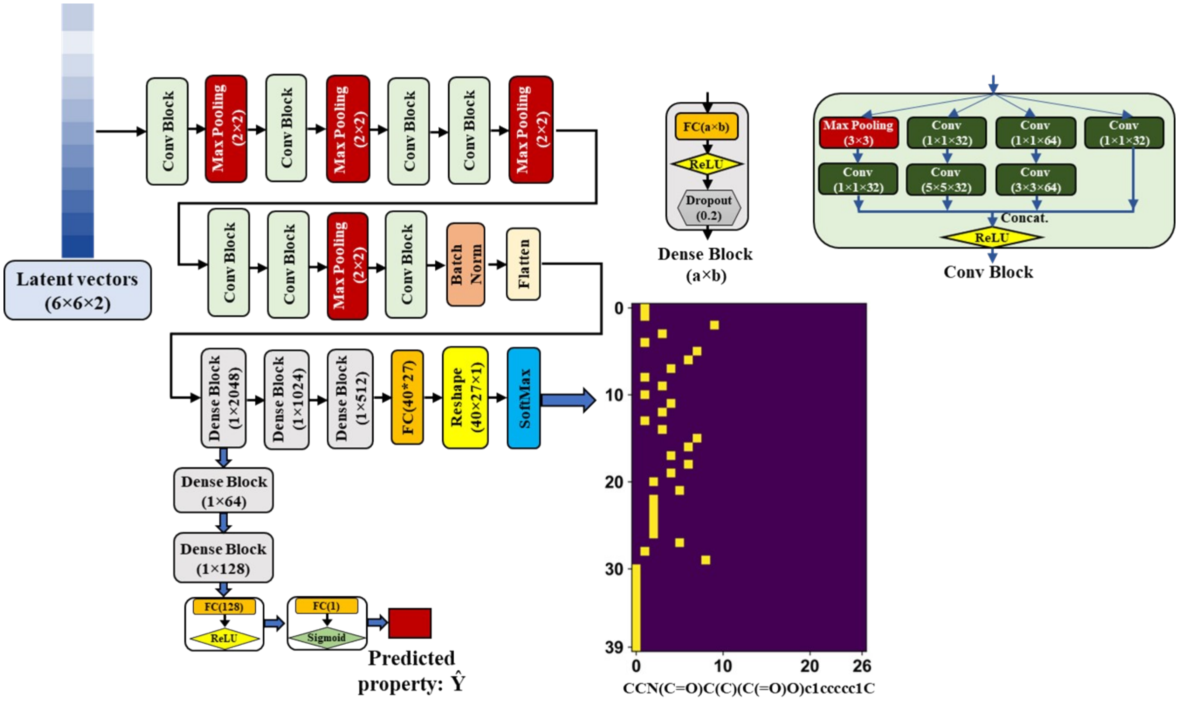

Figure S3. A schematic of the decoder. It takes the latent vectors (6×6×2) as input and outputs one-hot encoded SMILES (40×27).

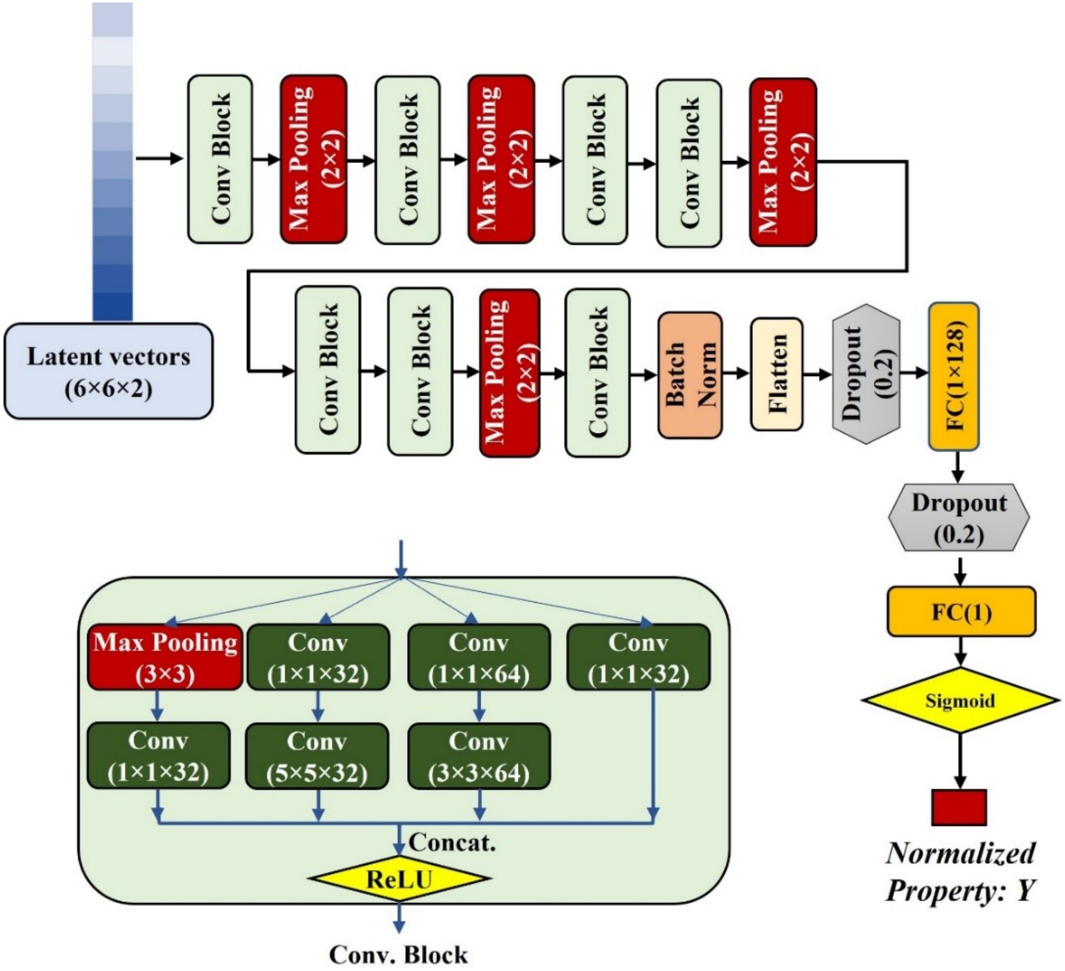

Figure S4. Architecture of the regressor. It takes the latent vectors (6×6×2) as input and outputs the $\Delta E_{H\text{-}L}$ values.

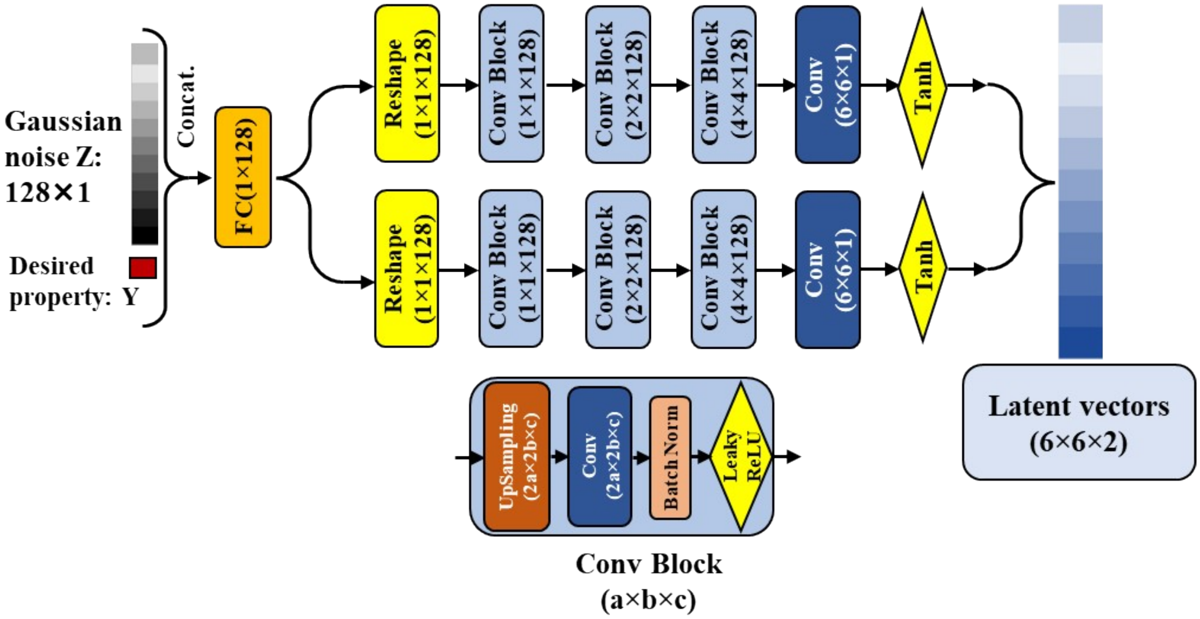

Figure S5. Architecture of the generator. It takes a random noise Z and a desired $\Delta E_{H\text{-}L}$ as inputs and generates the latent vectors of the molecules in response to the targeted $\Delta E_{H\text{-}L}$.

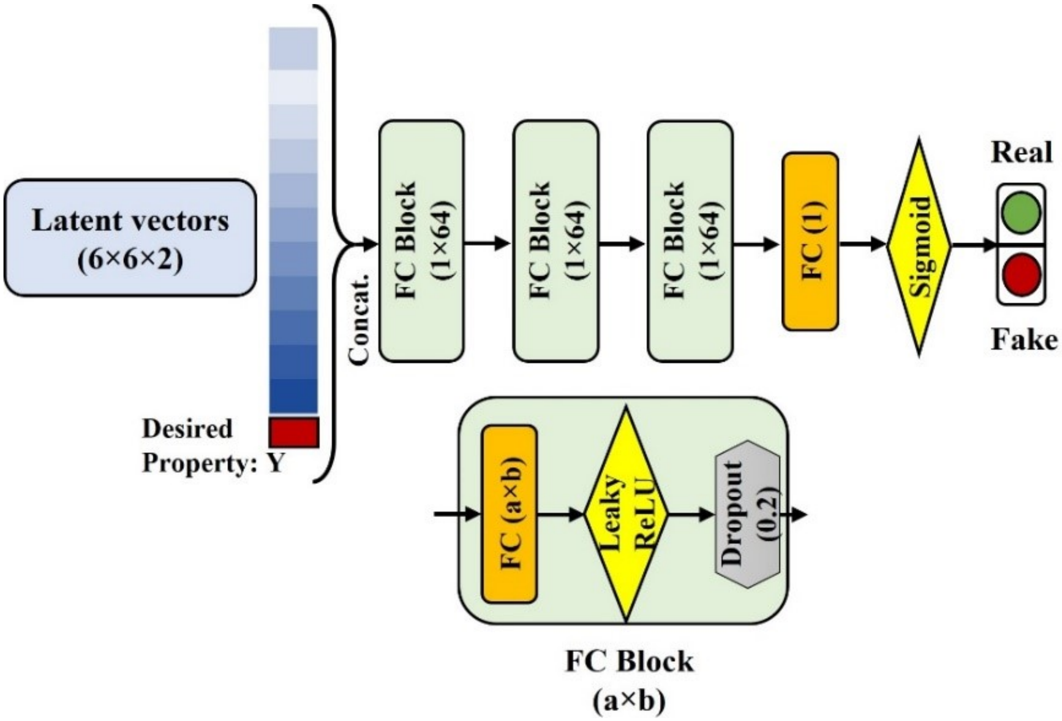

Figure S6. Architecture of the discriminator. It takes a real or fake (synthesized) latent feature which is concatenated with the corresponding $\Delta E_{H\text{-}L}$ as inputs to output a probability of being real (closer to 0 means fake and to "1" means real).

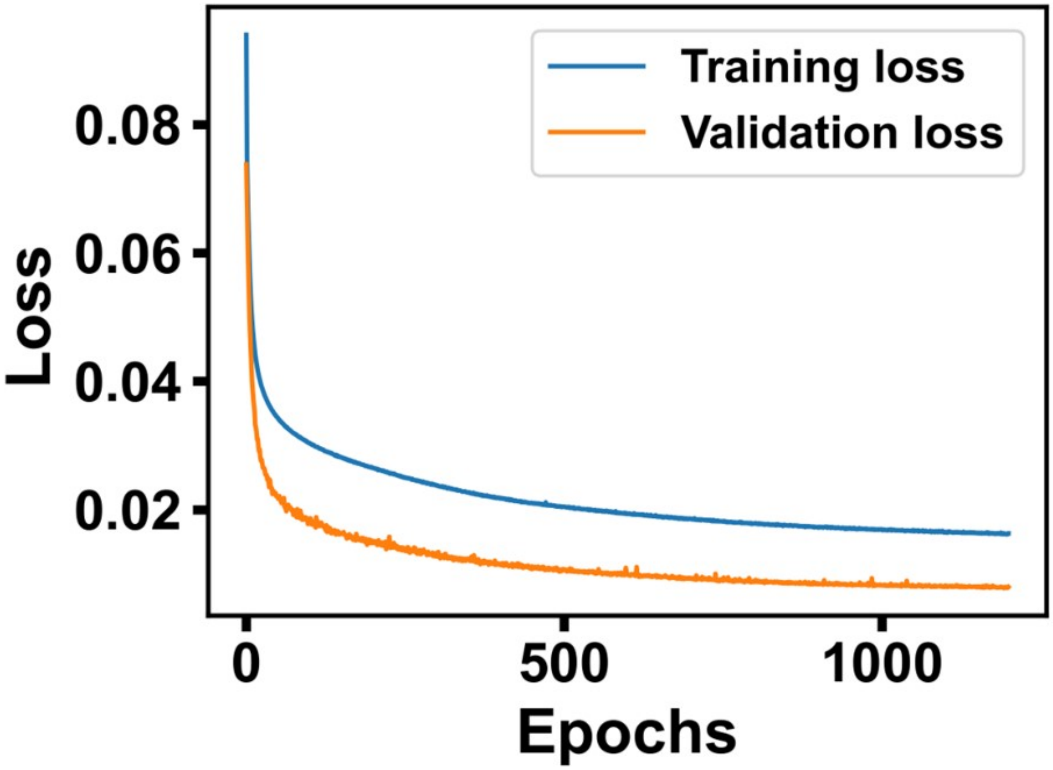

Figure S7. Loss evolution of the autoencoder during the training process. It shows that after 1000 epochs training, the loss is stabilized, indicating a success of the training.

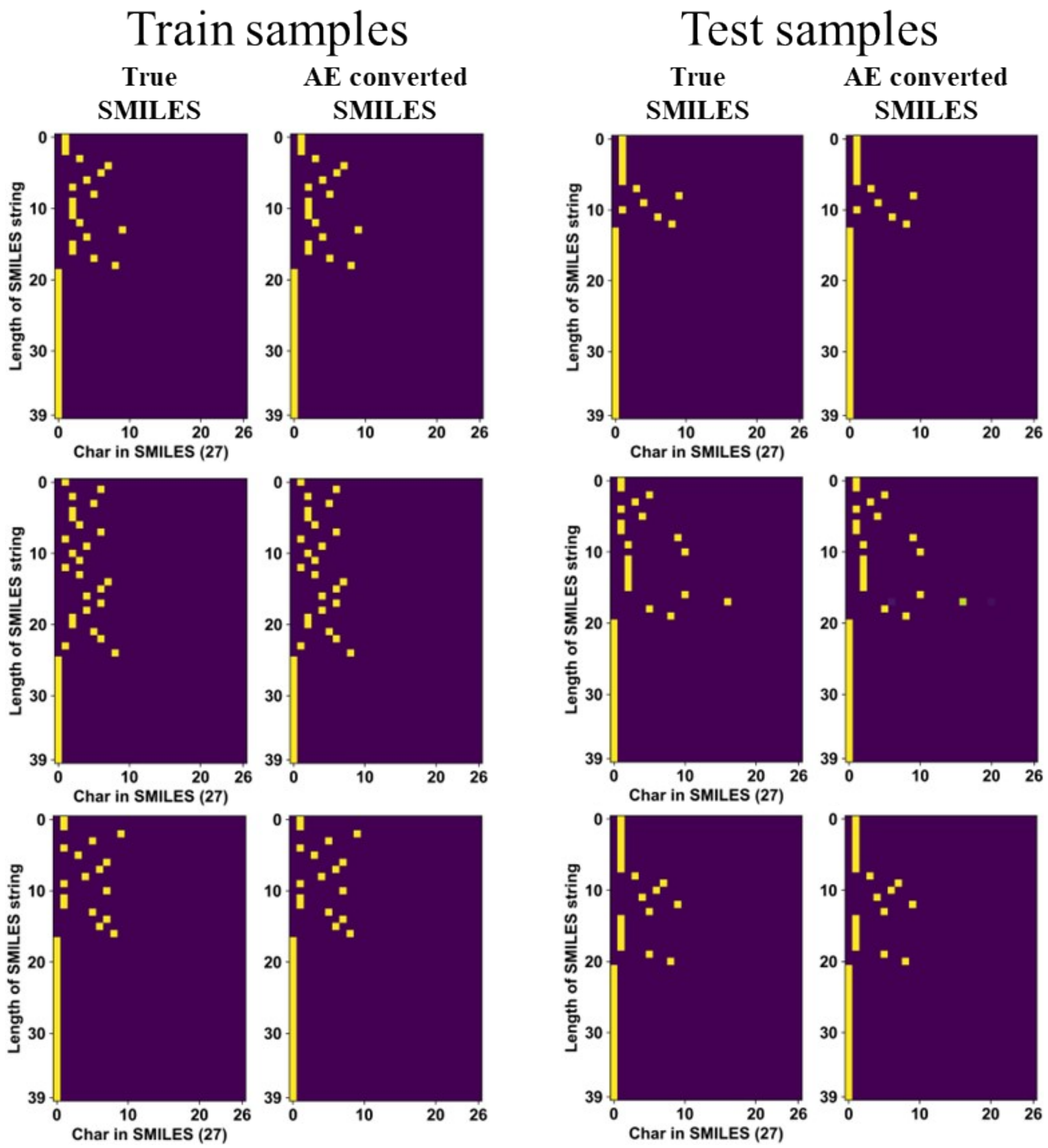

Figure S8. Comparison of true versus converted one-hot encoded SMILES using the AE.. Ideally, they should be the same. Left figures show 3 training molecules, and the right figure are from 3 testing molecules. The lighter the color is, the closer probability of that string to 1 is.

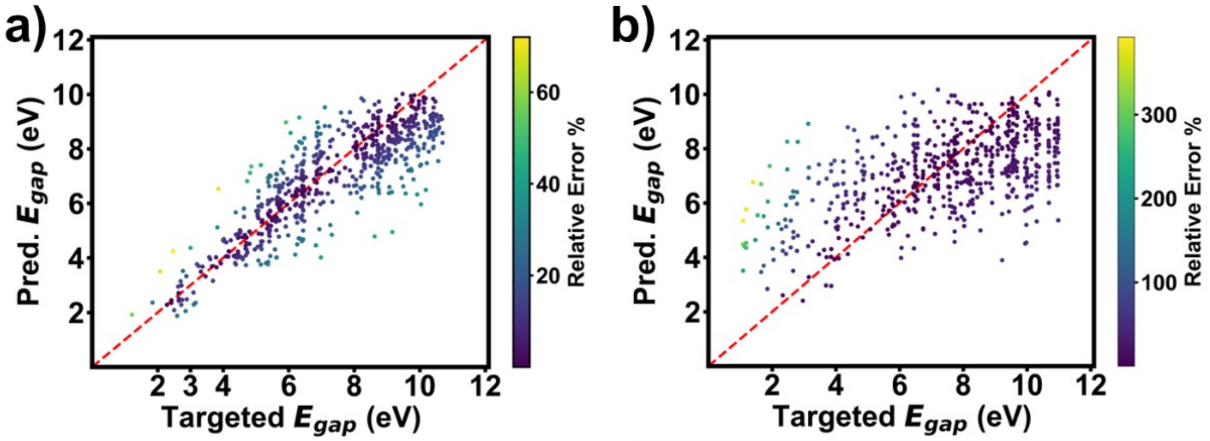

Figure S9. Comparison of the targeted and predicted gap values for two randomly generated batches for a) (6×6×2) with an R^2 of 0.70 and MAE of 0.91, and **b)** (8×8×2) with an R^2 of 0.30 and MAE of 1.7.

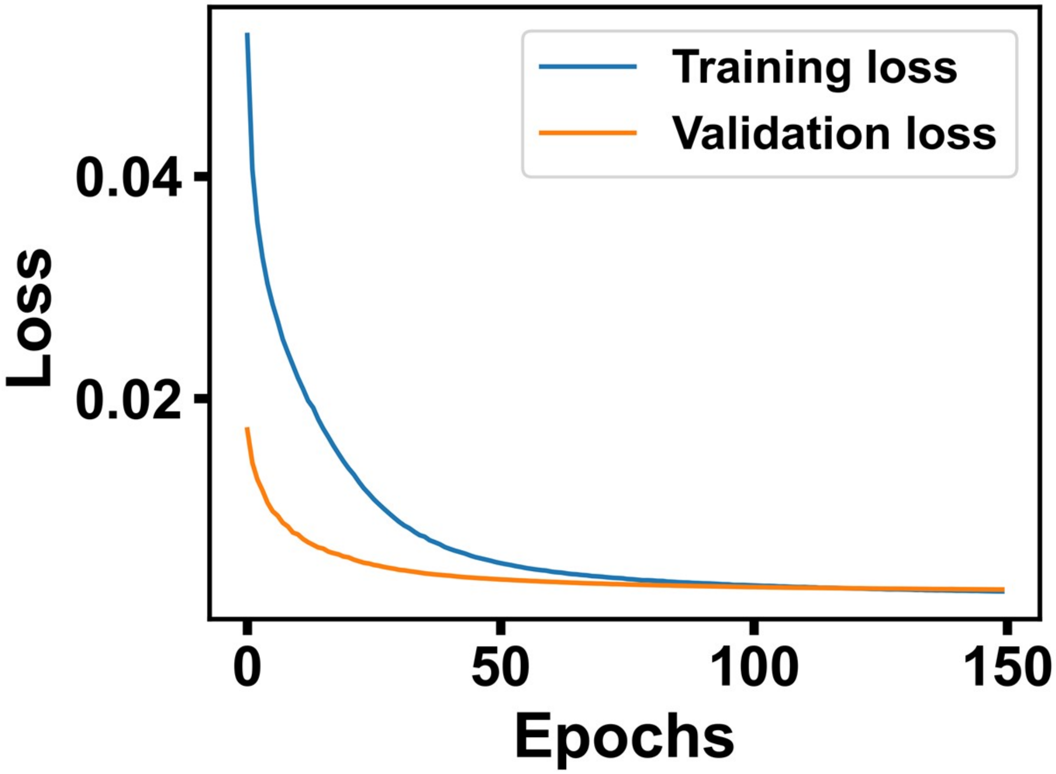

Figure S10. Loss evolution of the regressor during the training process. It shows that after 100 epochs training, the loss is stabilized, indicating a success of the training.

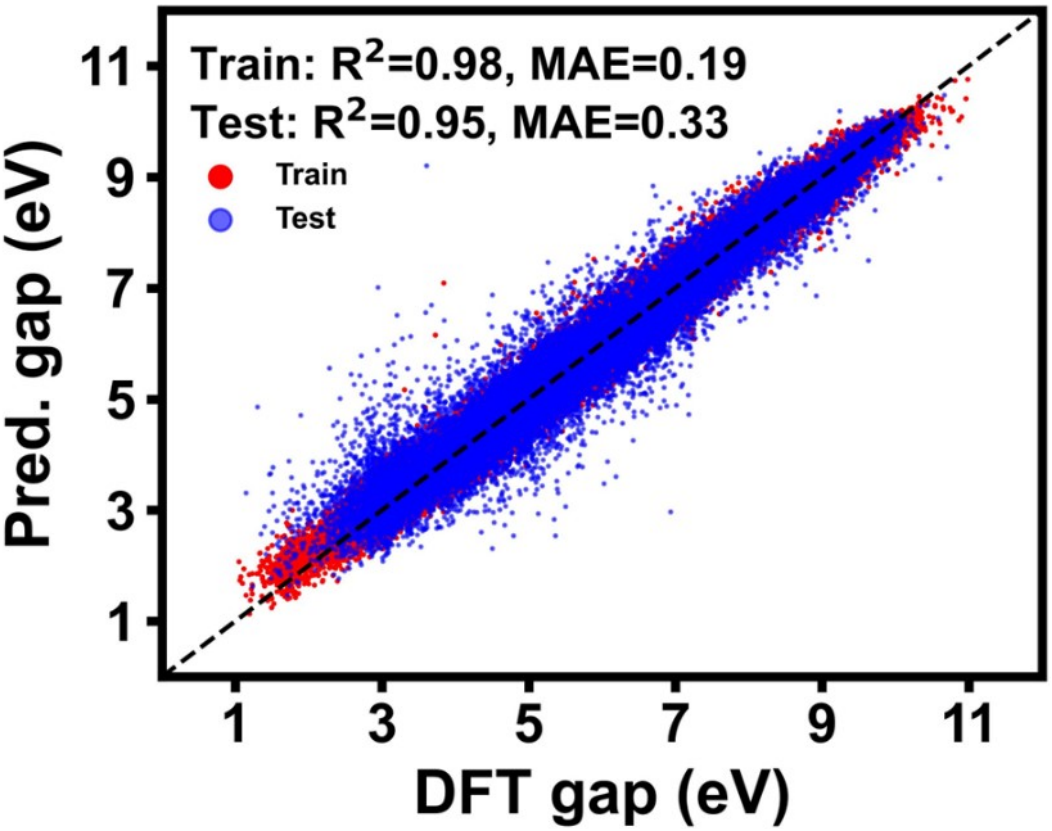

Figure S11. Distribution of the regressor predicted $\Delta E_{H\text{-}L}$ (in eV) and the true values of the molecules from the PubChemQC database.

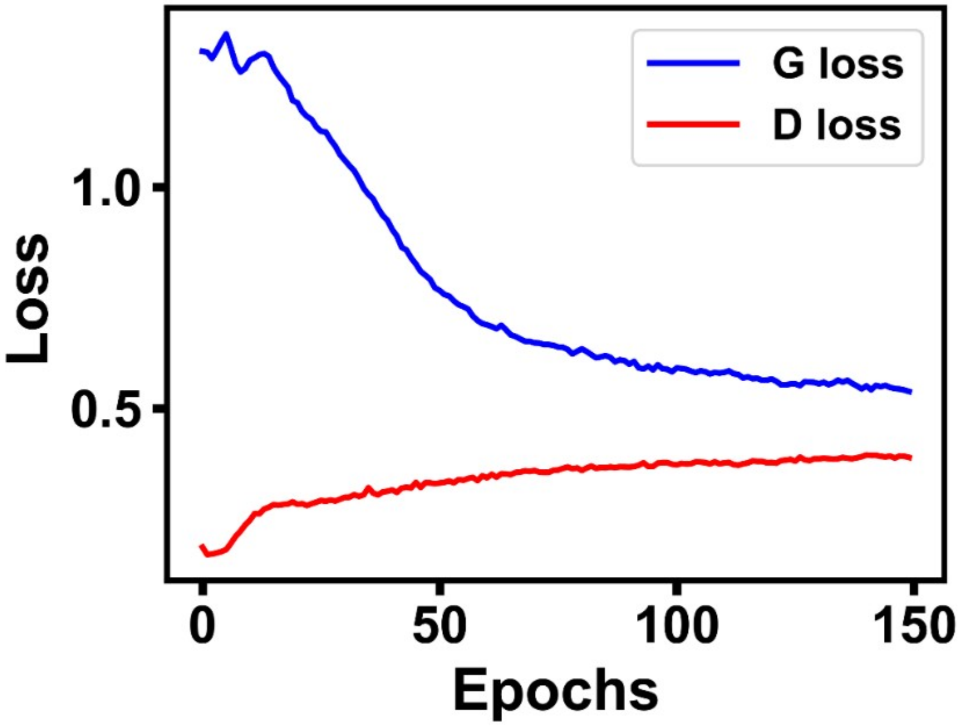

Figure S12. Loss evolution of the generator and discriminator during the training process. After 150 epochs, the losses of generator and discriminator were stabilized, showing a success of training.

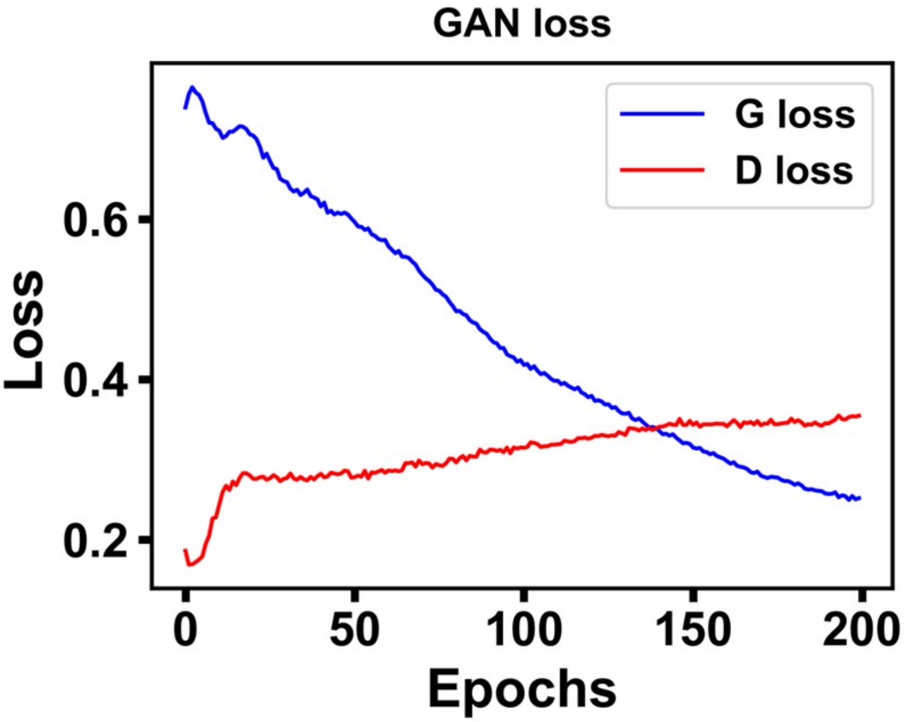

Figure S13. Loss evolution of the generator and discriminator during the training process without reinforcement section. Even with 200 epochs, the losses of the generator and discriminator did not converge.

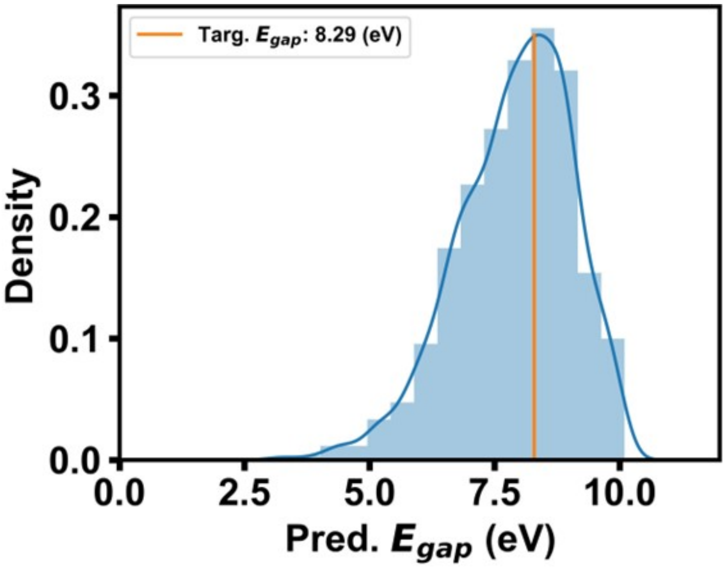

Figure S14. Distribution of the predicted values for ~2500 generated molecules corresponding to a targeted $\Delta E_{H\text{-}L}$ value of 8.29 eV. Among the generated molecules, 85% of them have a predicted $\Delta E_{H\text{-}L}$ value within 20% RE of the targeted one.

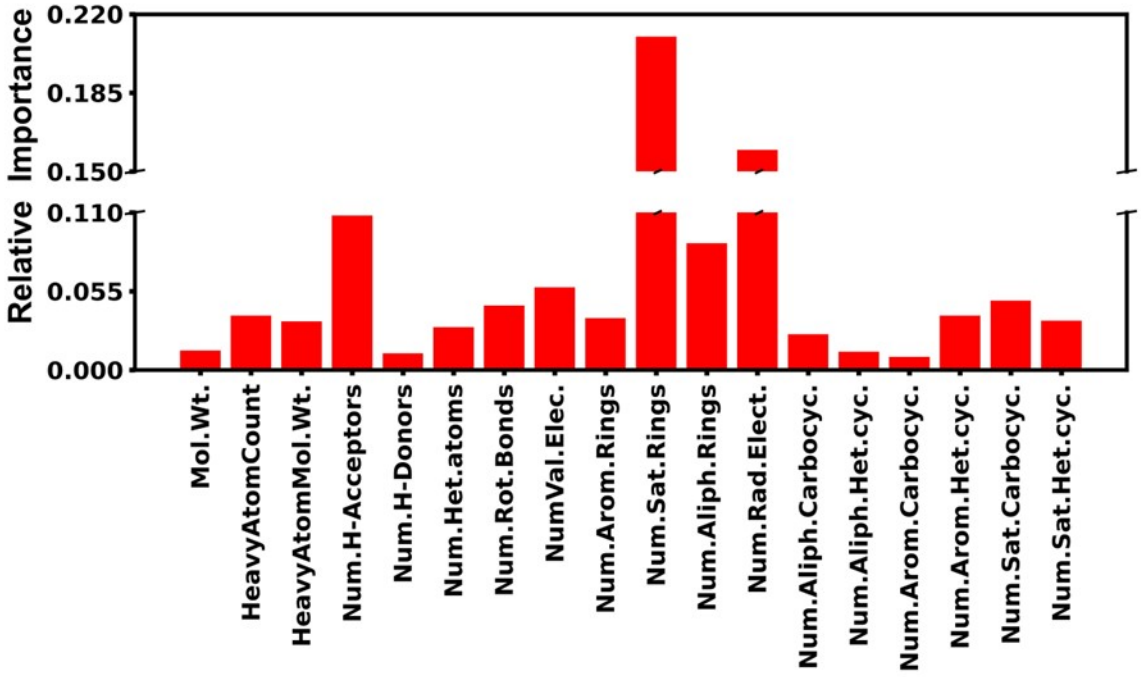

Figure S15. Feature importance values of the 18 input features extracted from the well-trained XGBoost model. The R^2 score for predicting the heat capacity using XGBoost model was 0.95 for training and 0.91 for testing. The hyperparameters of the XGBoost model are listed in Table S4. XGBoost is a prediction model that consists of assembling some weak decision trees.

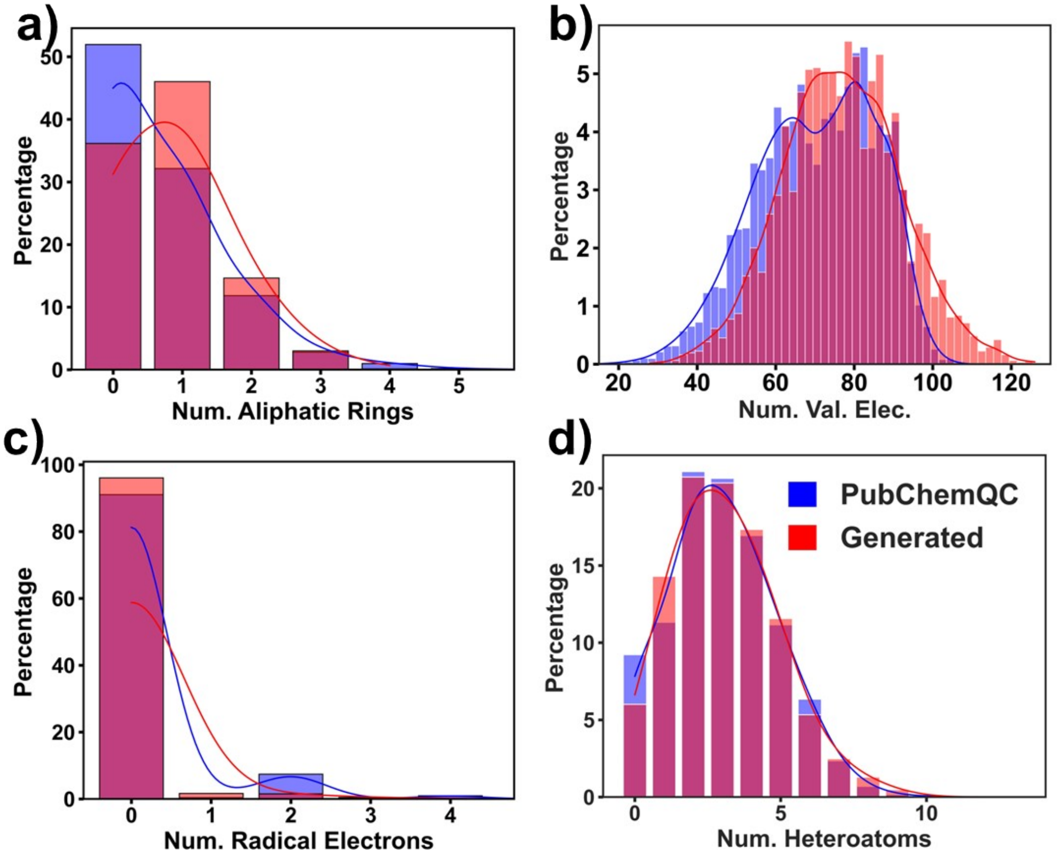

Figure S16. Density distribution of the four selected features for the generated versus training samples: (**a**) number of aliphatic rings; (**b**) number of valence electrons of the heavy atoms; (**c**) number of radical electrons; (**d**) number of heteroatoms.

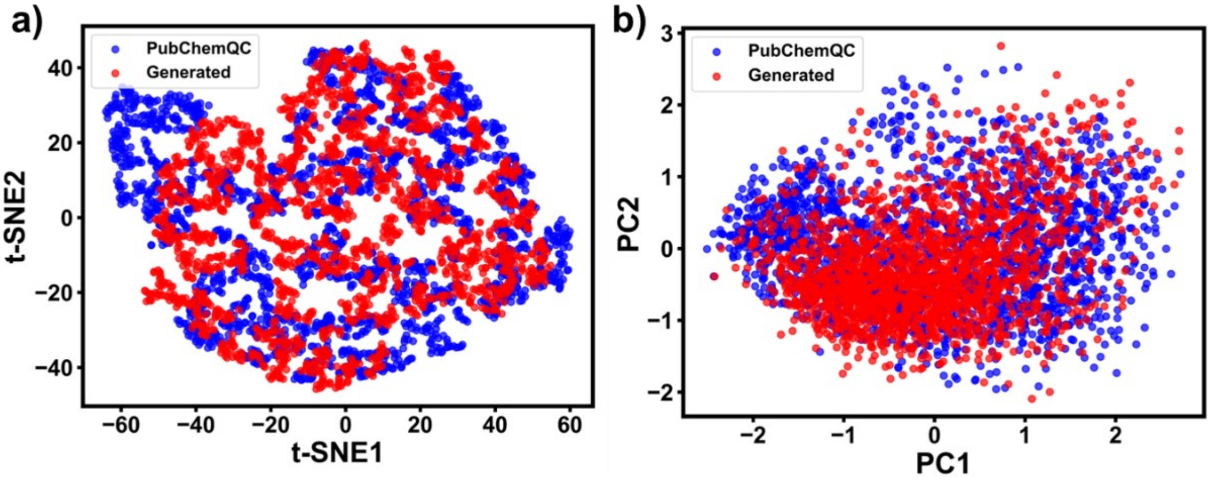

Figure S17. Dimension reduction of the latent features output from the encoder and the generator. The mapped space from (**a**) non-linear t-SNE and (**b**) PCA.

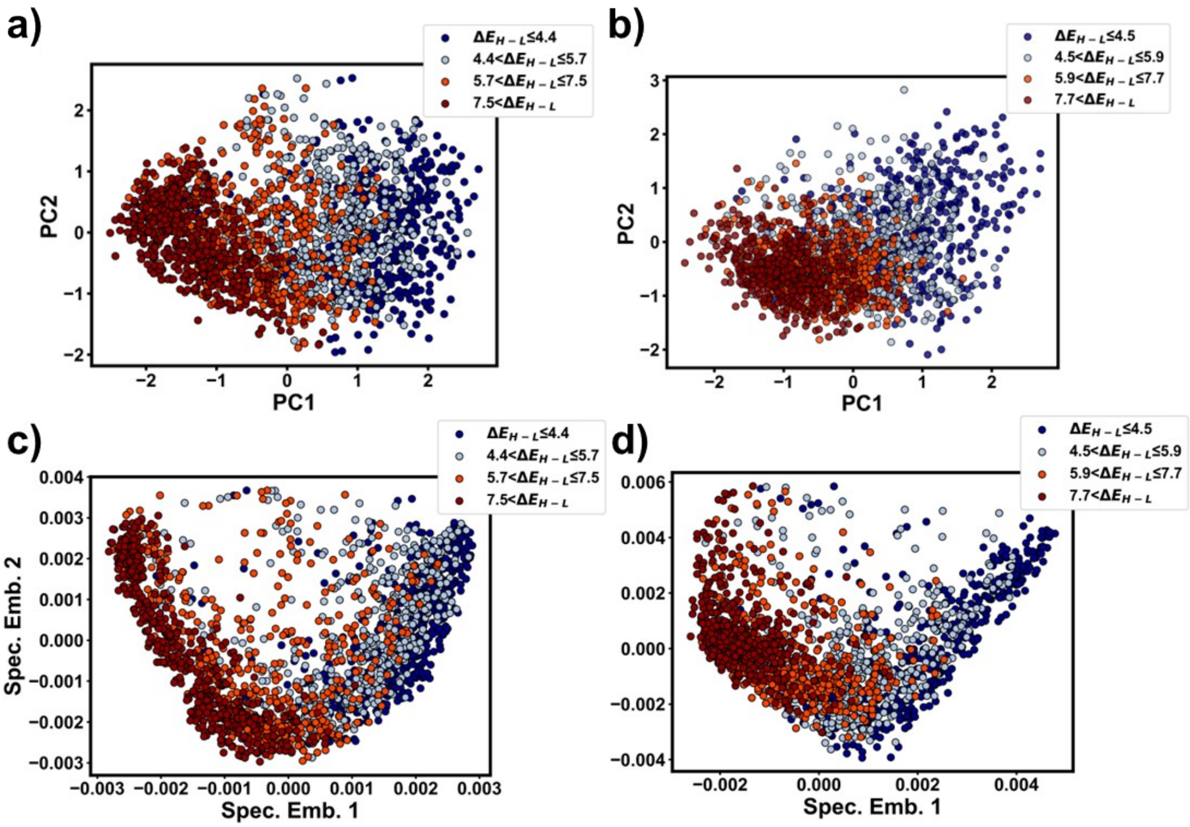

Figure S18. Dimension reduction of the latent features output from the encoder and the generator. PCA of the training (**a**) and generated molecules (**b**). Non-linear spectral embedding of the training (**c**) and generated molecules (**d**).

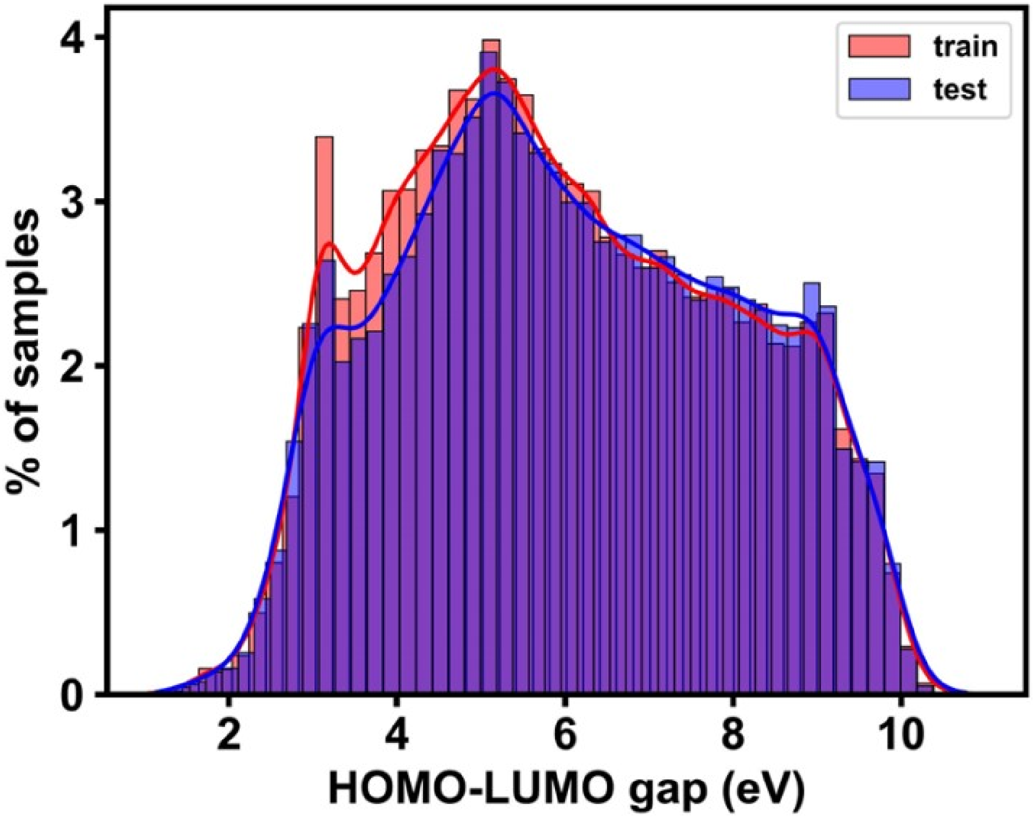

Figure S19. Distribution of ΔE_{H-L} for training (in red) and testing (in blue) in the PubChemQC database.

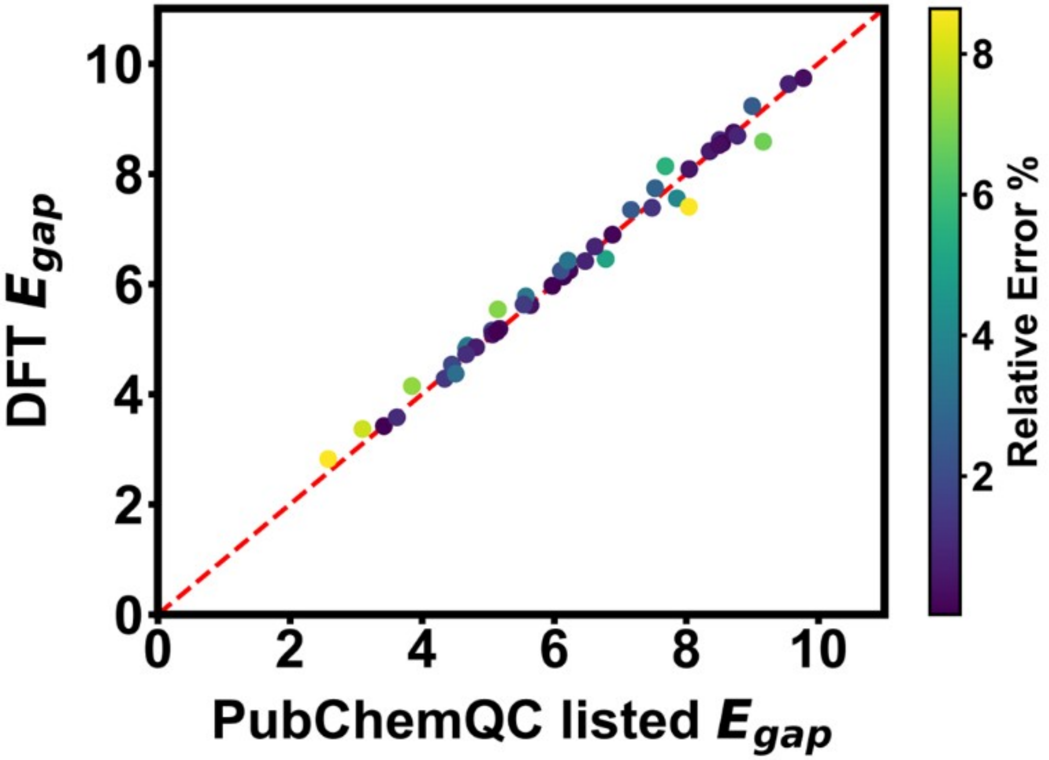

Figure S20. Comparison of the DFT calculated and the listed $\Delta E_{H\text{-}L}$ values of 46 randomly selected molecules from the PubChemQC database. They show an MAE of 0.14 (eV).

Supplementary References

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

2 Krenn, M., Häse, F., Nigam, A., Friederich, P. & Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Machine Learning: Science and Technology* **1**, 045024 (2020).

3 Bjerrum, E. J. SMILES enumeration as data augmentation for neural network modeling of molecules. Preprint at (2017).

4 Rupp, M., Tkatchenko, A., Müller, K.-R. & von Lilienfeld, O. A. Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning. *Physical Review Letters* **108**, 058301 (2012).

5 Sanchez-Lengeling, B. & Aspuru-Guzik, A. Inverse molecular design using machine learning: Generative models for matter engineering. *Science* **361**, 360-365 (2018).

6 Landrum, G. *Open-source Cheminformatics Software*, (2006); https://www.rdkit.org

7 O'Boyle, N. & Dalke, A. DeepSMILES: an adaptation of SMILES for use in machine-learning of chemical structures. (2018).

8 Kusner, M. J., Paige, B. & Hernández-Lobato, J. M. Grammar variational autoencoder. In Proc. *International conference on machine learning.* 1945-1954 (PMLR).

9 Maser, M. R. & Reisman, S. E. 3D Computer Vision Models Predict DFT-Level HOMO-LUMO Gap Energies from Force-Field-Optimized Geometries. (2021).

10 Pereira, F. *et al.* Machine Learning Methods to Predict Density Functional Theory B3LYP Energies of HOMO and LUMO Orbitals. *Journal of Chemical Information and Modeling* **57**, 11-21 (2017).

11 Popova, M., Isayev, O. & Tropsha, A. Deep reinforcement learning for de novo drug design. *Science Advances* **4**, eaap7885 (2018).

12 Yuan, Q., Santana-Bonilla, A., Zwijnenburg, M. A. & Jelfs, K. E. Molecular generation targeting desired electronic properties via deep generative models. *Nanoscale* **12**, 6744-6758 (2020).

13 Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C. & Aspuru-Guzik, A. Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models. Preprint at https://arxiv.org/abs/1705.10843 (2017).

14 Sanchez-Lengeling, B., Outeiral, C., Guimaraes, G. L. & Aspuru-Guzik, A. Optimizing distributions over molecular space. An objective-reinforced generative adversarial network for inverse-design chemistry (ORGANIC). Preprint at https://doi.org/10.26434/chemrxiv.5309668.v3 (2017).