Science Advances
AAAS

advances.sciencemag.org/cgi/content/full/6/1/eaax9324/DC1

Supplementary Materials for

Inverse design of porous materials using artificial neural networks

Baekjun Kim, Sangwon Lee, Jihan Kim*

*Corresponding author. Email: jihankim@kaist.ac.kr

Published 3 January 2020, *Sci. Adv.* **6**, eaax9324 (2020)
DOI: 10.1126/sciadv.aax9324

This PDF file includes:

Supplementary Materials and Methods

Section S1. Computational methods

Section S1-1. Details of ZeoGAN

The architecture of ZeoGAN is based on ESGAN (*29*) but it has some advanced characteristics: 1) extension to the material space, 2) more stable learning using WGAN with gradient penalty, and 3) user-desired property generation.

ZeoGAN takes three types of grid data (i.e. oxygen, silicon, and methane potential energy) and their shared lattice parameters as its input. If the memory is sufficient, these representations can be applied to other materials with a wider variety of atom types, given that we can increment the types of grid data for each additional atom type.

Data pre-processing

In the methane potential grids, a cut off energy value of 5000 K was imposed because the energy value goes to infinity when the gas molecule is overlapped at the framework atom positions (making the energy range too large). For all of the input zeolite structures, the methane potential energy range was set to [-4000, 5000] Kelvins and the lattice length ranges were set to [0, 150] Angstroms.

With the methane potential ranging from -4000 to 5000 K, we noted that energies near -4000 K contained more important information regarding methane adsorption capacity compared to 5000 K. As such, we set -4000 K to be 1 and 5000 K to be 0 in our normalization procedure as 1 symbolizes propagation from current layer to the next layer in the neural network. The lattice lengths were normalized into the range [0, 1].

Finally, to deal with the translation and rotation invariances of the periodic crystals, data augmentation method (*29*) was applied in the all of the input grids.

Section S1-2. Training of ZeoGAN

To train the ZeoGAN, we used Wasserstein GAN (WGAN) proposed by Arjovsky et al. (*27*). While normal GANs are trained to minimize the Jensen-Shannon (JS) divergence between data distribution P_r and generator distribution P_g, the WGAN is trained to minimize the Earth-Mover distance (EMD) (or Wasserstein-1) between P_r and P_g. One tractable form of EMD is given as follows

$$W(P_r, P_g) = \sup_{f \in \mathcal{L}_1} \mathbb{E}_{\mathbf{x} \sim P_r}[f(\mathbf{x})] - \mathbb{E}_{\mathbf{x}' \sim P_g}[f(\mathbf{x}')] \tag{4}$$

where $\mathcal{L}_1$ is the set of 1-Lipschitz function, f is the critic function, P_r is the data distribution, P_g is the generator distribution, $\mathbf{x}$ is a grid from data distribution and $\mathbf{x}'$ is a grid from generator distribution. The critic is the counterpart of the discriminator in normal GANs but it is called as "the critic" in WGAN variants because the critic is not trained to discriminate the real samples and the generated samples. If we replace f to a parameterized function, the supremum in EMD can be considered as maximization problem as below

$$W(P_r, P_g) = \max_{w \in \mathcal{W}} \mathbb{E}_{\mathbf{x} \sim P_r}[f_w(\mathbf{x})] - \mathbb{E}_{\mathbf{x}' \sim P_g}[f_w(\mathbf{x}')] \tag{5}$$

$$\mathcal{W} = \{w | f_w \in \mathcal{L}_1\} \tag{6}$$

where f_w is parameterized critic function and w is the parameter of the parameterized critic function.

Because the aim of the critic function is to estimate the EMD between P_r and P_g, the loss function for the critic, $L_{\text{cri,WGAN}}$, is defined as below

$$L_{\text{cri,WGAN}} = -\mathbb{E}_{\mathbf{x}\sim P_r}[f_w(\mathbf{x})] + \mathbb{E}_{\mathbf{x}'\sim P_g}[f_w(\mathbf{x}')] \tag{7}$$

And the minimization problem given as the following

$$\min_{w\in\mathcal{W}} L_{\text{cri,WGAN}} \tag{8}$$

For optimal critic parameter w, the critic can estimate the EMD. On the other hand, the aim of the generator is to imitate data distribution by generating samples similar to the real data. So the generator needs to be trained to minimize the EMD between data and generator distribution. If we replace $\mathbf{x}'$ with the generator function $g_\theta(\mathbf{z})$, we can get a form of the EMD as a function of generator parameter θ as follows

$$\begin{aligned} W(P_r, P_g) &= \max_{w\in\mathcal{W}} \mathbb{E}_{\mathbf{x}\sim P_r}[f_w(\mathbf{x})] - \mathbb{E}_{\mathbf{x}'\sim P_g}[f_w(\mathbf{x}')] \\ &= \max_{w\in\mathcal{W}} \mathbb{E}_{\mathbf{x}\sim P_r}[f_w(\mathbf{x})] - \mathbb{E}_{\mathbf{z}\sim P_\mathbf{z}}\left[f_w\left(g_\theta(\mathbf{z})\right)\right] \end{aligned} \tag{9}$$

where $\mathbf{z}$ is noise source of the generator, and $P_\mathbf{z}$ is the noise distribution. Because learning of generator is performed on fixed critic and the first term of RHS of the equation has no parameter θ, the loss function for generator $L_{\text{gen,WGAN}}$ becomes

$$L_{\text{gen,WGAN}} = -\mathbb{E}_{\mathbf{z}\sim P_\mathbf{z}}\left[f_w\left(g_\theta(\mathbf{z})\right)\right] \tag{10}$$

By solving the minimization problem

$$\min_\theta L_{\text{gen,WGAN}} \tag{11}$$

the generator can minimize the EMD between P_r and P_g and as such, the generator can create samples similar to the real data.

In the original WGAN paper, Lipschitz constraint on the critic is attained by weight clipping. Gulrajani et al. (*28*) have suggested improved technique for training the WGAN with the gradient penalty (WGAN-GP). In the WGAN-GP, the 1-Lipschitz constraint on the critic is attained by the gradient penalty. The loss function of the critic of WGAN-GP is given as follows

$$L_{\text{cri,WGAN-GP}} = L_{\text{cri,WGAN}} + \alpha\mathbb{E}_{\hat{\mathbf{x}}\sim P_{\hat{\mathbf{x}}}}[(\|\nabla_{\hat{\mathbf{x}}} f_w(\hat{\mathbf{x}})\| - 1)^2] \tag{12}$$

$$\hat{\mathbf{x}} = t\mathbf{x} + (1 - \mathrm{t})\mathbf{x}' \text{ for } t\sim U[0,1] \tag{13}$$

where α is the gradient penalty coefficient, $P_{\hat{\mathbf{x}}}$ is the distribution of $\hat{\mathbf{x}}$, and U is the uniform distribution. $P_{\hat{\mathbf{x}}}$ is defined implicitly from the definition of $\hat{\mathbf{x}}$. The WGAN-GP uses the fact that the optimal critic function satisfying 1-Lipschitz continuity has unit gradient norm almost everywhere under $P_{\hat{\mathbf{x}}}$.Because the 1-Lipschitz constraint is attained by gradient penalty, the original minimization problem becomes unconstrained minimization problem on w as below

$$\min_{w\in\mathcal{W}} L_{\text{cri,WGAN}} \longrightarrow \min_w L_{\text{cri,WGAN-GP}} \tag{14}$$

So far we explained only losses about the "grid" part of the zeolite shape. ZeoGAN generates grid values and lattice constants jointly unlike normal GANs. In order to generate the lattice constants, an auxiliary network, R_ϕ that generates the lattice constants was attached to first hidden layer of the generator (See fig. S1B). ϕ is the parameter of the auxiliary network and we named the network as "lattice generator". Different from the case of the grids, the lattice generator is not trained using GAN objectives. To train the lattice generator, first a model is trained to approximate conditional probability $P(\mathbf{c}|\mathbf{x})$ then the lattice generator is trained by minimizing the negative log-likelihood $-\log \mathbb{E}_{\mathbf{z}\sim P_\mathbf{z}}[P(\mathbf{c}'|\mathbf{x}')]$ for θ and ϕ. Here, $\mathbf{c}$ is the lattice constants of real zeolites and $\mathbf{c}'$ is the generated lattice constants. To infer the lattice constants from the grids**,** an auxiliary network, Q_ψ, is attached to last hidden layer of critic network (See fig. S1A). ψ is the parameter of the auxiliary network and we named the network as "lattice regressor" because it infers the lattice constants from the activations of last hidden layer of critic. Essentially, the activations of last hidden layer is calculated from the grid $\mathbf{x}$. So the whole forward path from $\mathbf{x}$ to output layer of Q_ψ can be considered as a regression model for the learning of conditional probability $P(\mathbf{c}|\mathbf{x})$.

The loss function for the training of lattice regressor, $L_{\text{cri,lattice}}$, is given as follows

$$L_{\text{cri,lattice}} = \mathbb{E}_{(\mathbf{c},\mathbf{x})\sim P_r}[H(\mathbf{c}, \tilde{\mathbf{c}})] \tag{15}$$

$$H(\mathbf{x}, \mathbf{y}) = \frac{1}{N}\sum_{i=1}^{N} x_i \log y_i + (1 - x_i)\log(1 - y_i) \tag{16}$$

where $\tilde{\mathbf{c}}$ is inferred lattice constants from $\mathbf{x}$ (See fig. S1a). The loss for the training of the lattice generator, $L_{\text{gen,lattice}}$, is given as

$$L_{\text{gen,lattice}} = \mathbb{E}_{(\mathbf{c}\prime,\mathbf{x}\prime)\sim P_g}[H(\tilde{\mathbf{c}}', \mathbf{c}')] \tag{17}$$

where $\tilde{\mathbf{c}}'$ is inferred lattice constant from $\mathbf{x}'$ (See fig. S1a). The optimization of above losses is performed with other losses in the same way as training of WGAN-GP. So the lattice regressor and the lattice generator are trained gradually with the critic and the generator.

The additional loss for feature-matching (*33*) is included to improve the convergence of ZeoGAN. We define the feature of a tensor $\mathbf{x}$ as the following

$$\Phi_\mathbf{x} = \log\left(\frac{1}{N}\sum_{i=1}^{N} e^{-\beta x_i}\right) \tag{18}$$

where $\Phi_\mathbf{x}$ is the feature calculated from $\mathbf{x}$. In the case of energy grid, the feature is the Helmholtz free energy of target gas in the framework (*29*). However, in the case of "material" grid, the feature has no analogous physical meaning.

The feature-matching loss for a tensor $\mathbf{x}$, $L_{\text{FM},\mathbf{x}}$, is defined as

$$L_{\text{FM},\mathbf{x}} = \left|\mathbb{E}_{\mathbf{x}\sim P_r}[\Phi_\mathbf{x}] - \mathbb{E}_{\mathbf{x}\prime\sim P_g}[\Phi_{\mathbf{x}\prime}]\right| + \left|\mathbb{S}_{\mathbf{x}\sim P_r}[\Phi_\mathbf{x}] - \mathbb{S}_{\mathbf{x}\prime\sim P_g}[\Phi_{\mathbf{x}\prime}]\right| \tag{19}$$

$$\mathbb{S}[X] = \sqrt{\text{Var}(X)} \tag{20}$$

The grids used in ZeoGAN consists of three channels like below

$$\mathbf{x} = \left[\mathbf{x}_{\text{energy}}, \mathbf{x}_{\text{oxygen}}, \mathbf{x}_{\text{silicon}}\right] \tag{21}$$

In ZeoGAN, the feature-matching is applied separately to the each channel of the zeolite grid

$$L_{\mathrm{FM}} = L_{\mathrm{FM},\mathbf{x}_{\mathrm{energy}}} + L_{\mathrm{FM},\mathbf{x}_{\mathrm{oxygen}}} + L_{\mathrm{FM},\mathbf{x}_{\mathrm{silicon}}} \tag{22}$$

To the end, the loss for critic function, L_{cri}, is given as

$$L_{\mathrm{cri}} = L_{\mathrm{cri,WGAN-GP}} + L_{\mathrm{cri,lattice}} \tag{23}$$

$$\min_{w,\psi} L_{cri}$$

And the loss for generator, L_{gen}, is given as

$$L_{\mathrm{gen}} = L_{\mathrm{gen,WGAN}} + \gamma L_{\mathrm{gen,lattice}} + L_{\mathrm{FM}} \tag{24}$$

$$\min_{\theta,\phi} L_{gen}$$

where γ is scaling factor.

We implemented the neural network model using TensorFlow and training takes about 20 days on a single GeForce GTX 1080Ti until 500k step. Details on ZeoGAN architectures and hyperparameters are included in fig. S1 and table S1.

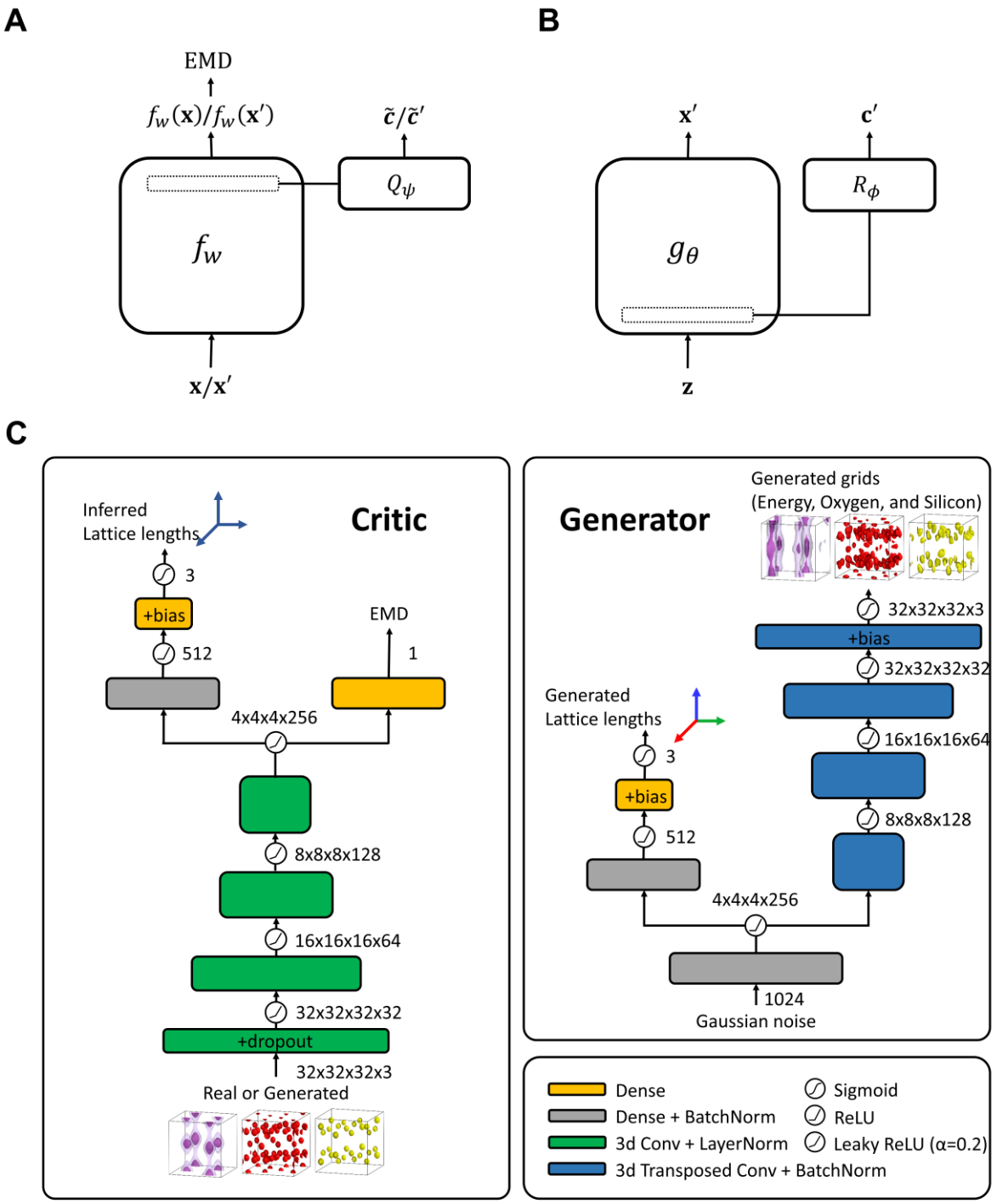

Fig. S1. Architecture of ZeoGAN. (**A**) The critic network and auxiliary lattice inference network Q_ψ (lattice regressor). The dashed box indicates last hidden layer of the critic network. (**B**) The generator network and auxiliary sub-network R_ϕ generating lattice constant (lattice generator). The dashed box indicates first hidden layer of the generator. (**C**) More implementation details about the critic and the generator.

Table S1. ZeoGAN hyperparameters.

Hyperparameter types		Values
Batch size		32
Initial weights	Mean	0.00
	Standard deviation	0.02
Convolution kernel size		5 x 5 x 5
Adam optimizer (*37*)	Learning rate	0.0001
	β_1	0.5
	β_2	0.9
Dropout rate		0.5
Lattice scaling factor		0.1
Feature matching system temperature (Kelvin)	Energy	298
	Material	0.06
Gradient penalty hyperparameter α		10.0
Critic updates per generator update		5

Section S1-3. Training for user-desired properties

To enforce the generator to create materials with user-desired property, additional loss for the generator, L_{UD}, is given as follows

$$L_{\mathrm{UD}} = \mathbb{E}_{Y' \sim P_g}[g(\log Y')] \tag{25}$$

$$g(x) = \max(x - x_u, 0)^2 + \min(x - x_l, 0)^2 \tag{26}$$

where Y is the target property, subscript u indicates the upper bound, subscript l indicates the lower bound and $g(x)$ is the penalty function.

The total loss of the ZeoGAN generating zeolites of user-desired property is given as

$$L_{\mathrm{gen,UD}} = L_{\mathrm{gen}} + \beta L_{\mathrm{UD}} \tag{27}$$

where β is the scaling factor for L_{UD} (β=100 in this work). The training of ZeoGAN with the constraint for property bias was conducted with pre-trained parameter of the normal ZeoGAN. The EMD and L_{UD} over training steps are shown in fig. S2. As shown in the figure, the EMD is not affected by existence of the constraint on property and L_{UD} is dropped sharply at the beginning of the training (See fig. S2B).

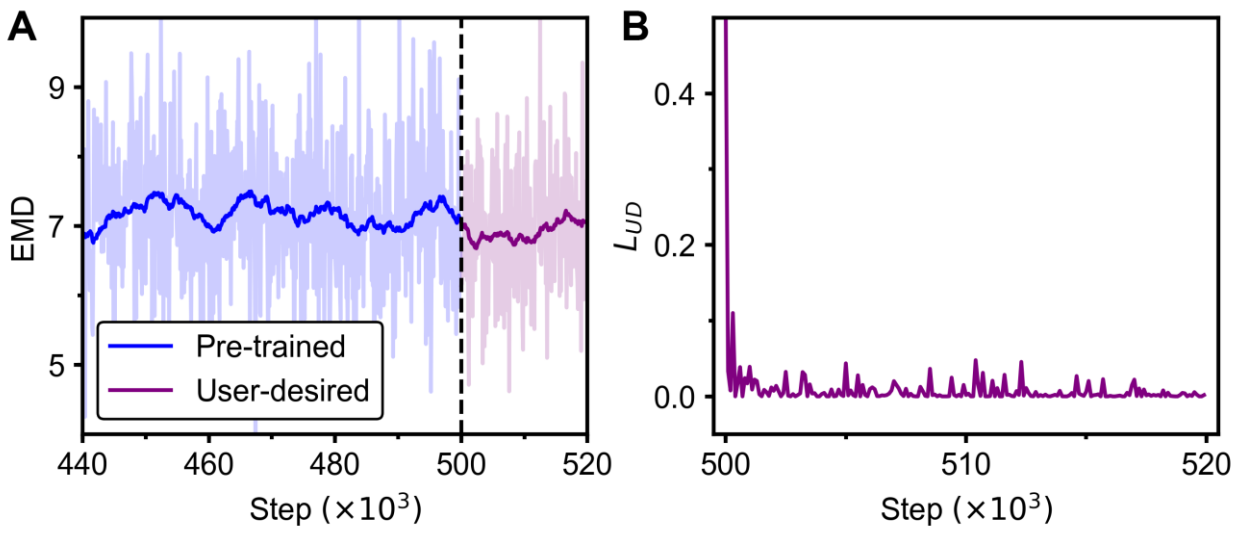

Fig. S2. Training results after adding user-desired loss. (**A**) EMD over training steps. (**B**) L_{UD} over training steps after addition of the loss at 500k step.

Section S1-4. Zeolite cleanup

Assignment of atomic positions

A volume-based atomic position assignment were used for assignment of Si and O to avoid undesired local maximums at the noisy region of the ANN generated shapes. The volume-based algorithm is sufficient for material shapes because a specific volume (a collection of grid points) always surrounds each atomic positions with containing high Gaussian function values.
The volume-based approach consists of three parts: 1) detection of the Gaussian volumes, 2) separation of the Gaussian volumes and 3) assignment of atomic positions. At the first step, the surrounding volumes were detected using a simple flood-fill algorithm with a volume detection limit of 0.6. After this detection, the flood-fill algorithm conducted again with a limit of 0.7 to separate several atoms in the detected volume. For each volumes after separation, the atomic position was determined by weighted mean of grid positions and the Gaussian values.

Connectivity of zeolite structures

All of atoms in the four-connected pure-silica zeolite structures have their proper bond counts. The silicon atoms are connected to the nearest four oxygen neighbors and the oxygen atoms connected the nearest two silicon neighbors.

In this work, the connectivity of the pure-silica zeolite structure was defined as the ratio of the number of atoms that have their proper bond counts. To check the connectivity of the ANN generated structures, the Si-O bond calculation was conducted within the sufficient bond threshold 2.5 Å. The zeolite cleanup algorithm is described below and fig. S3 explains the details of the algorithm with illustrations to facilitate understanding.

Connectivity Repairing Algorithm

Algorithm 1. Connectivity Repairing Algorithm.
All simulations in this work used the default values n_{cycle} = 10000, f_{update} = 100, θ_{random} = 0.3, w_{max} = 10.

Require: n_{cycle}, the number of cycles. f_{update}, the update frequency of cycle ($n_{total} = n_{cycle} \times f_{update}$). θ_{random}, the probability of allowing random moves. w_{max}, the maximum number of waiting on the absence of improvement.

for $i, \ldots, n_{cycle}$ **do**
  for $t, \ldots, f_{update}$ **do**
    if $Connectivity(S) > Connectivity(S_{best,cycle})$ **then**
      $S_{best,cycle} \leftarrow S$
      if $Connectivity(S) = 1$ **then** break
    end if
    $S_{next} \leftarrow$ a randomly selected move from **Fig. S3**
    $\theta \leftarrow$ a random number [0, 1]
    if $\theta < \theta_{random}$ **then**
      $S \leftarrow S_{next}$
    else if $Connectivity(S_{next}) \geq Connectivity(S)$ **then**
      $S \leftarrow S_{next}$
    end if
  end for
  if $Connectivity(S_{best,cycle}) > Connectivity(S_{best})$ **then**
    $w \leftarrow 0$
    $S_{best} \leftarrow S_{best,cycle}$
  else
    $w \leftarrow w + 1$
  end if
  if $w > w_{max}$ **then**
    Stop searching (No improvement) and return S_{best}
  end if
  if $Connectivity(S_{best}) = 1$ **then**
    Return S_{best}
  end if
  $S \leftarrow S_{best}$
end for

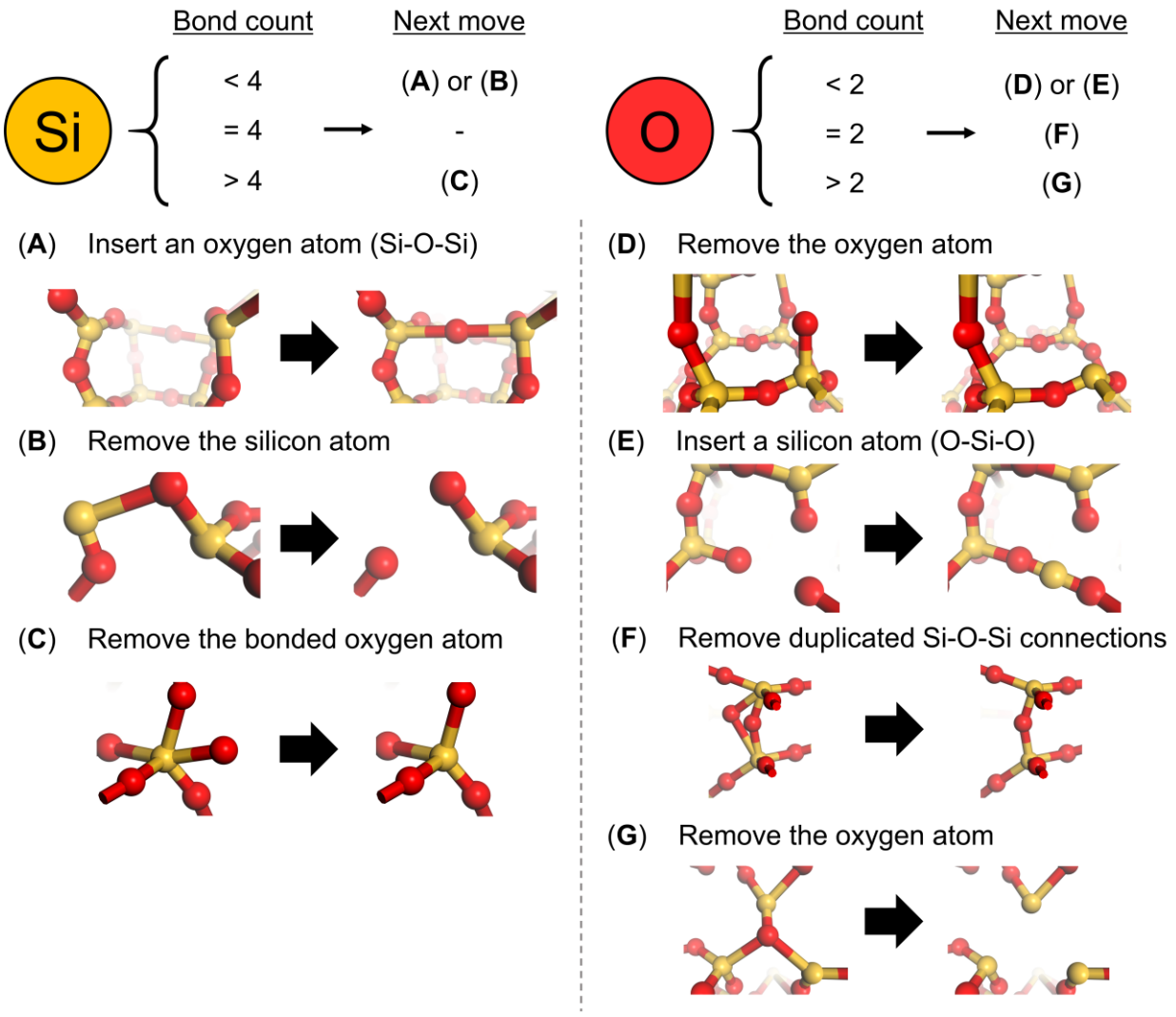

Fig. S3. Allowed next structure moves for the connectivity repairing algorithm. One of these moves is randomly selected for the next iteration in our connectivity repairing algorithm. The Si-O bond lengths are always less than 2.5 Å. (**A**) In the case of where the silicon atom is unsaturated (bond count is less than their proper bond count), an oxygen atom can be inserted at the midpoint between another unsaturated Si. (fig. S3 **B**, **D** and **G**) Removal of atoms are also required when the atom has inaccurate bond counts. (**C**) If a silicon atom has overfull bonds, one of its bonded atom can be removed in the next structure. (**E**) A silicon atom can be inserted between the unsaturated oxygen atoms. (**F**) The duplicated Si-O-Si connections are rejected.

Structure relaxation

The fully connected cleaned-up zeolites are relaxed by sequential minimization steps. In the first step, geometry optimizations were performed with the Universal Force Field (UFF) (*38*) with fixed lattice parameters After this pre-relaxation, the constant pressure energy minimization were conducted with the GULP software (*39*). The SLC interatomic potential (*40*) was chosen to describe zeolite atoms with core-shell polarization. To overcome the distance sensitivity of the SLC model, the previous UFF optimization step was required.

Section S2. Zeolite generation results

Section S2-1. Generated non–user-desired zeolite structures

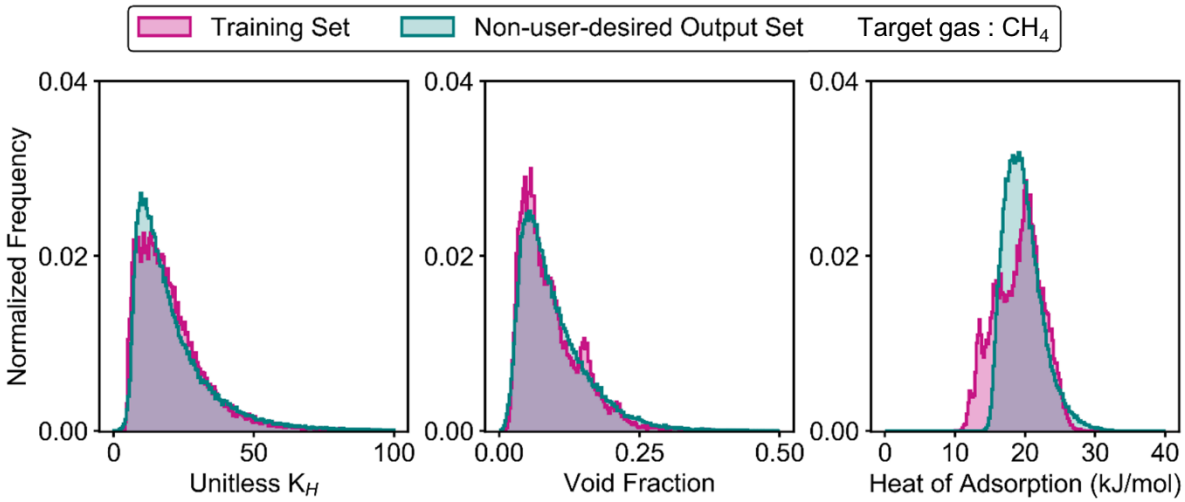

Fig. S4. Distributions (methane KH, methane void fraction, and methane heat of adsorption) for non–user-desired generation. From 31,713 training set structures (pink), and one million non-user-desired ZeoGAN output shapes (green).

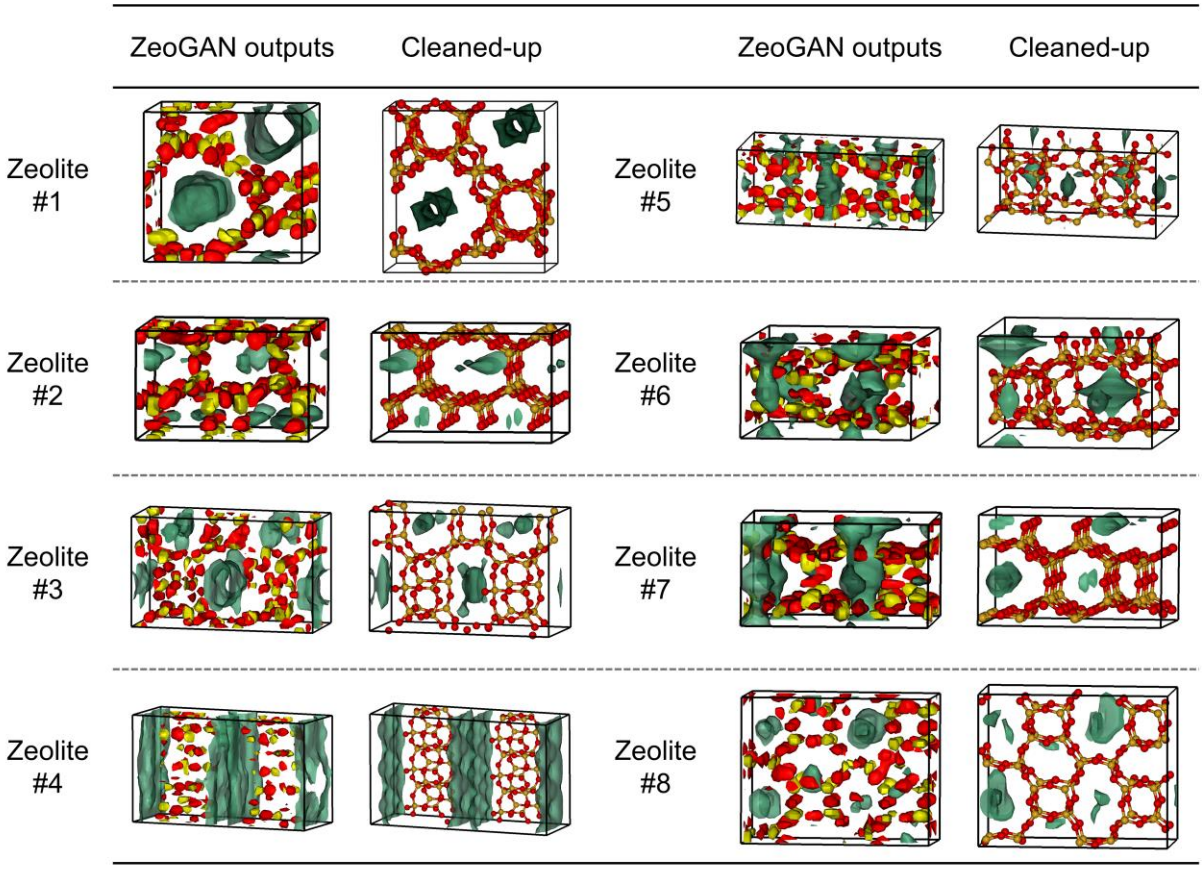

Fig. S5. Summary of non–user-desired zeolite shapes (both materials and energy shapes) and corresponding cleaned-up structures (both materials and energy shapes). The cleaned-up energy shapes were obtained from classical molecular simulations.

Table S2. Various methane properties of the eight non–user-desired zeolites. The methane properties were obtained from classical molecular simulations.

	Number of unique Si atoms	Unitless K_H		Void Fraction		Heat of Adsorption (kJ/mol)	
		ZeoGAN output	After clean-up	ZeoGAN output	After clean-up	ZeoGAN output	After clean-up
Zeolite #1	2	34.16	46.31	0.13	0.05	18.94	23.58
Zeolite #2	2	16.62	0.62	0.03	0.02	34.34	18.11
Zeolite #3	6	27.93	15.34	0.10	0.04	22.61	22.45
Zeolite #4	1	17.52	5.55	0.20	0.27	16.10	10.43
Zeolite #5	3	29.79	11.97	0.09	0.03	25.42	23.92
Zeolite #6	3	13.31	14.52	0.05	0.04	21.20	22.44
Zeolite #7	6	11.17	6.74	0.05	0.02	19.80	24.06
Zeolite #8	10	12.57	8.97	0.03	0.03	27.76	21.80

Section S2-2. Generated user-desired zeolite structures

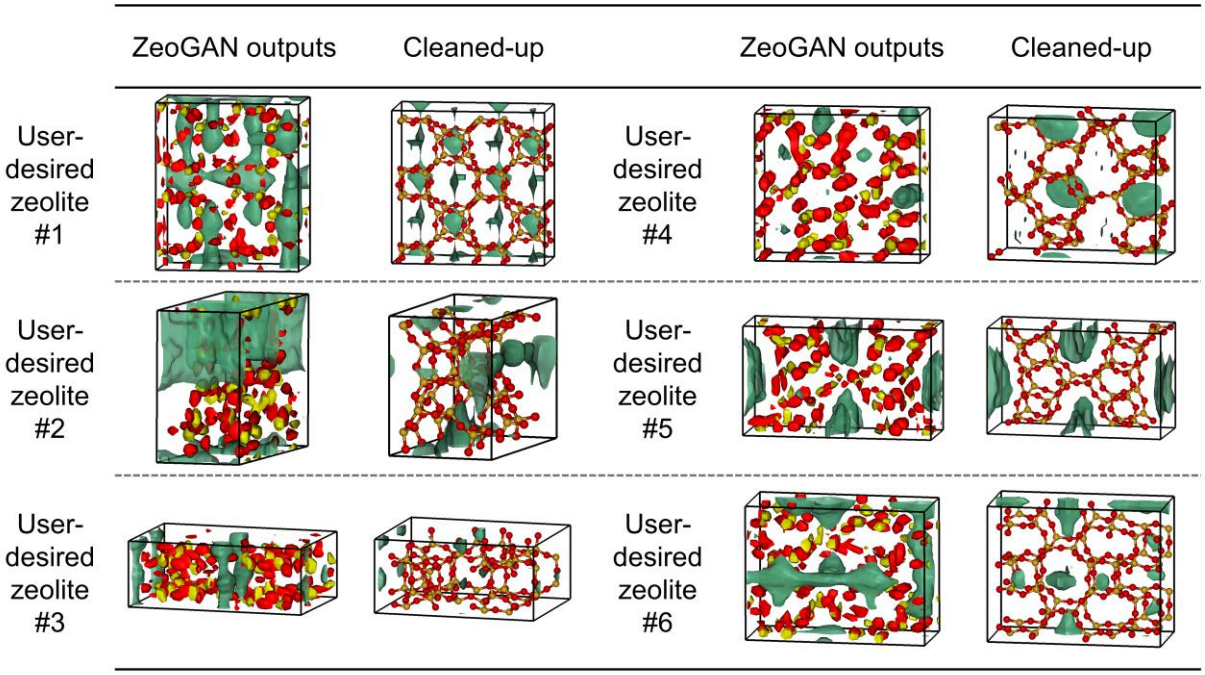

Fig. S6. Summary of user-desired zeolite shapes (both materials and energy shapes) and corresponding cleaned-up structures (both materials and energy shapes). The cleaned-up energy shapes were obtained from classical molecular simulations.

Table S3. Various methane properties of the six user-desired zeolites. The methane properties were obtained from classical molecular simulations.

	Number of unique Si atoms	Unitless K_H		Void Fraction		Heat of Adsorption (kJ/mol)	
		ZeoGAN output	After clean-up	ZeoGAN output	After clean-up	ZeoGAN output	After clean-up
User-desired zeolite #1	1	13.13	6.56	0.09	0.05	19.64	19.45
User-desired zeolite #2	5	35.36	12.75	0.15	0.05	18.62	21.01
User-desired zeolite #3	8	5.58	5.92	0.04	0.02	18.95	23.62
User-desired zeolite #4	2	6.80	7.06	0.02	0.03	21.90	21.92
User-desired zeolite #5	5	24.54	15.84	0.09	0.11	19.76	17.12
User-desired zeolite #6	7	10.86	5.89	0.06	0.03	19.83	23.00

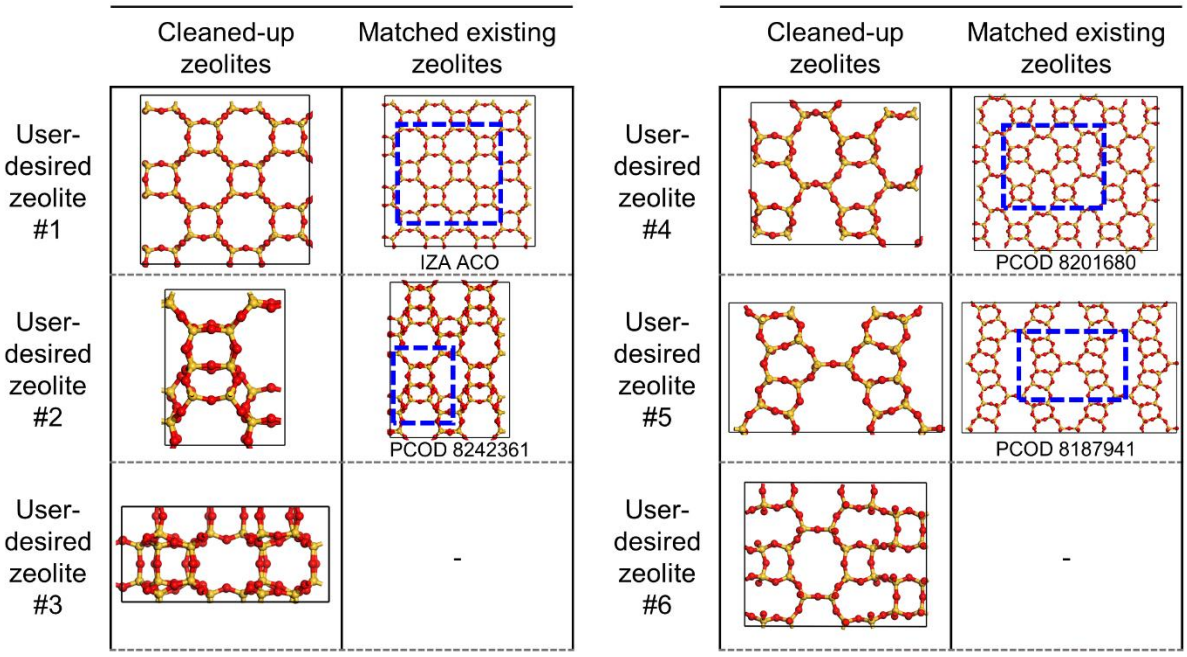

Fig. S7. Matching zeolites in IZA/PCOD for the user-desired zeolites. Blue dashed lines indicate the exact corresponding unit cell match.