Generative models for graph-based protein design----NeurIPS 2019

John Ingraham, Vikas K. Garg, Regina Barzilay, Tommi Jaakkola

Scaffold-based molecular design with a graph generative model----The Royal Society of Chemistry 2020

Jaechang Lim, Sang-Yeon Hwang, Seokhyun Moon, Seungsu Kimb and Woo Youn Kim

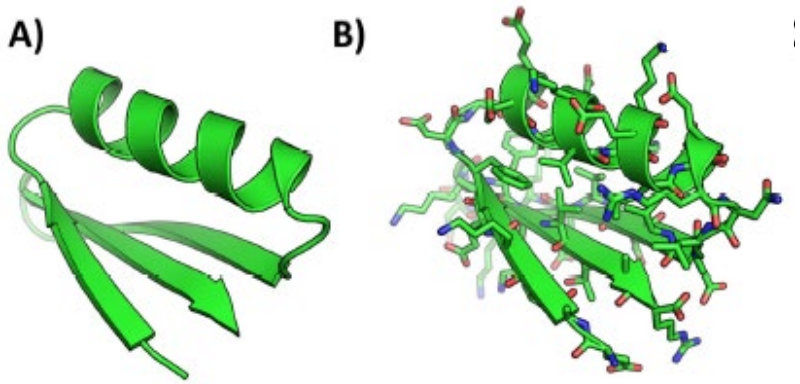
Presenter: Minjie Cheng

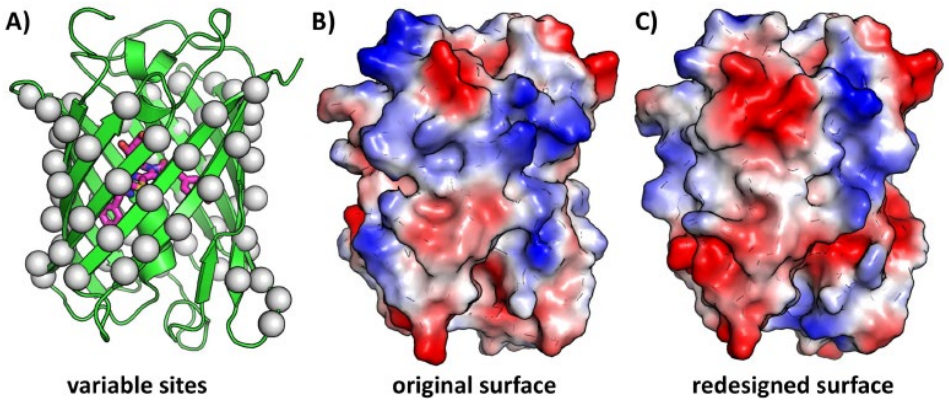
outline

1. Protein generation

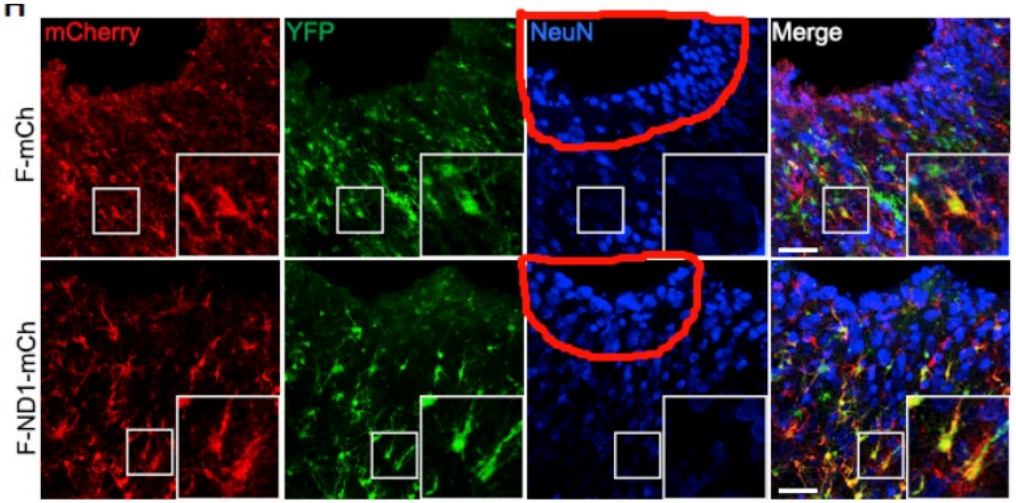
2. Molecular generation based on Scaffold

Motivation



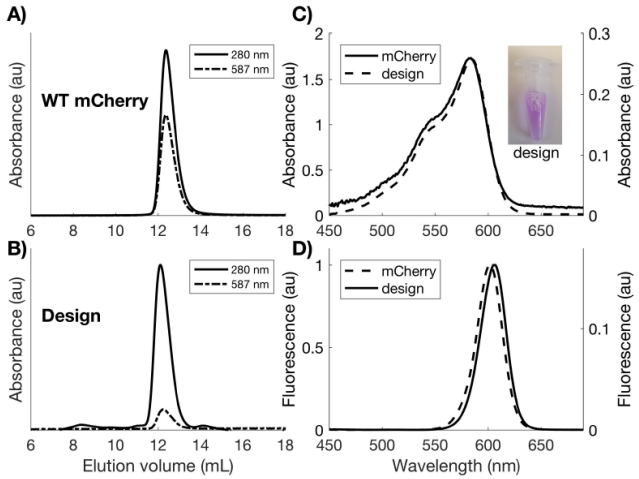


An excemple :Total surface redesign of mCherry



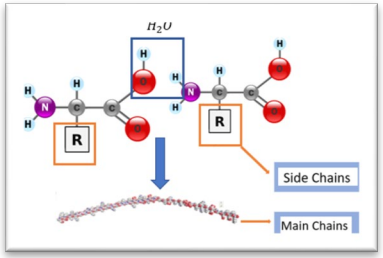
Revisting astrocyte to neuron conversion with lineage tracting in vivo （2021) --cell

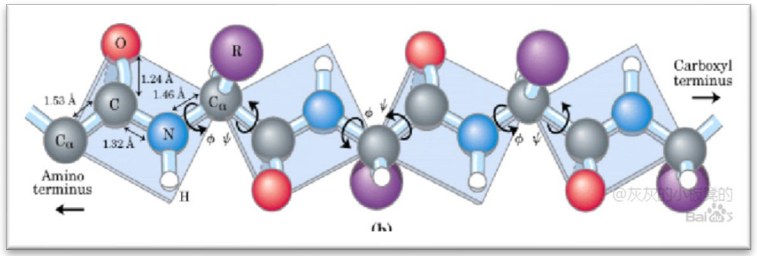
The ability to generate such diversity can be easily exploited to quickly engineer variants of RFP or other proteins that possess a range of desired properties



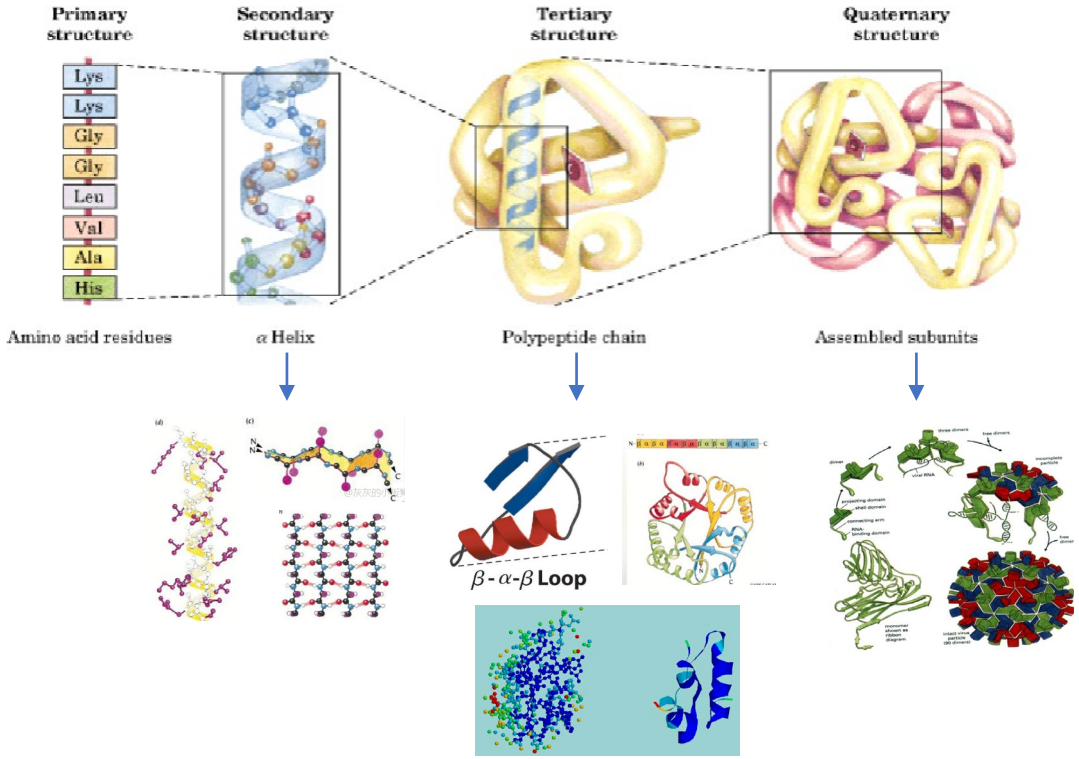
A general-purpose protein design framework based on mining sequence-structure relationships in known protein structures(2018)

Protein generation



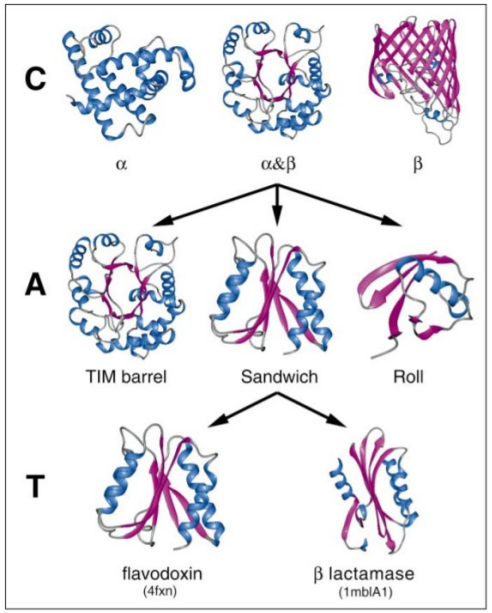


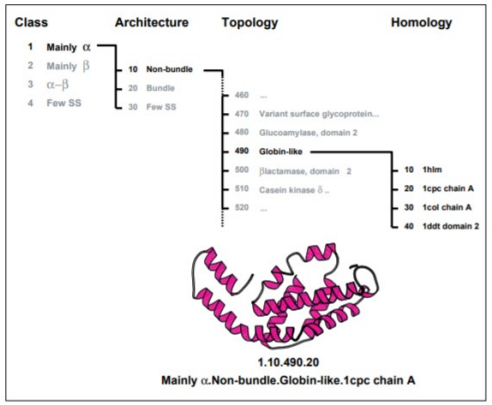
Flexible backbone & Precise atom locations

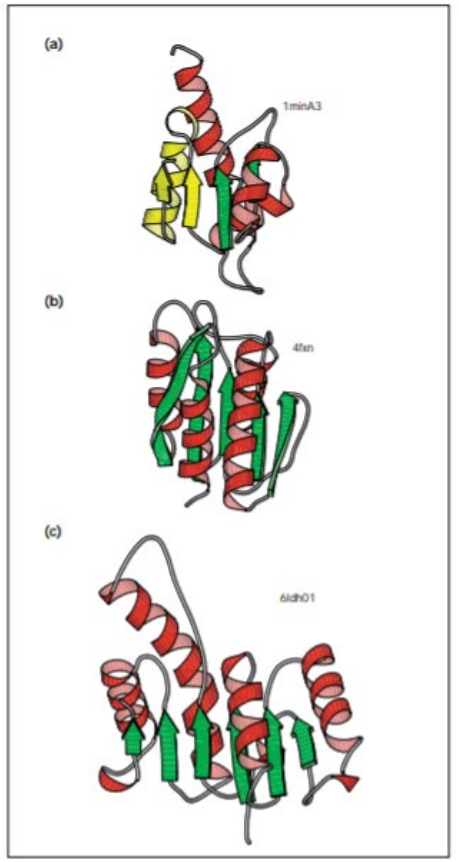


The four structure of proteins

CATH: a hierarchic classification of protein domain structures



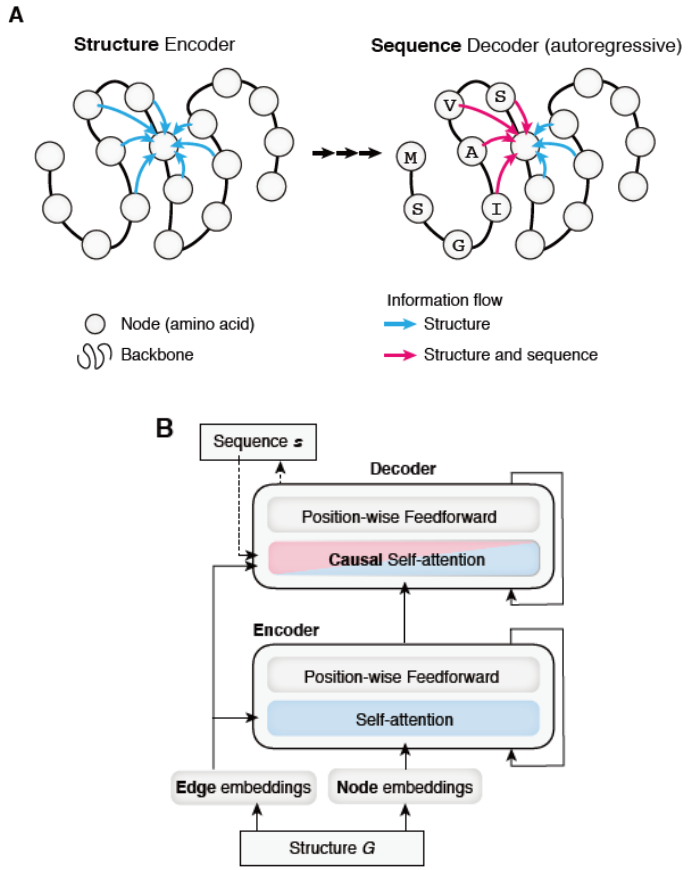




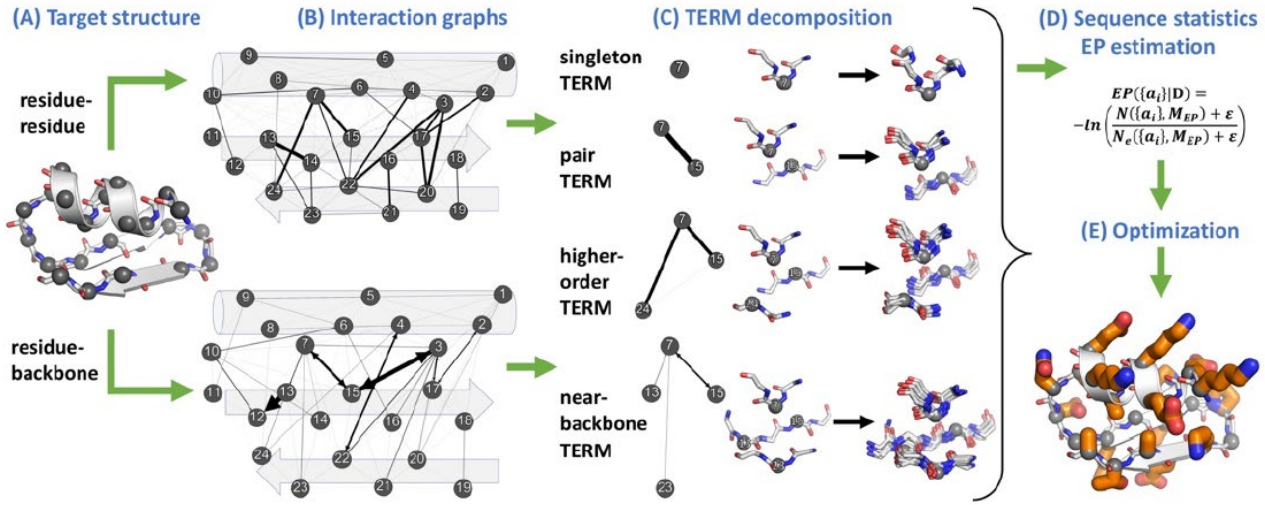
The 'Russian doll' effect for the flavodoxin fold family. (a–c) MOLSCRIPT diagrams for representatives from different homologous superfamilies (H-level) of the flavodoxin fold family (CAT number 3.40.50) in the α–β class. All members of the family contain recurring βαβ motifs, coloured yellow in (a), and the progression from four-stranded β sheet in 1minA3 through to five β strands in 4fxn to six β strands in 6ldh01, by addition of βαβ motifs, illustrates the Russian doll effect for this fold family. The smaller protein (1minA3, 110 residues) has two thirds the number of residues as the largest shown (6ldh01, 162 residues).

CATH — a hierarchic classification of protein domain structures (1997)
AlphaFold

Motivation of Structured Transformer

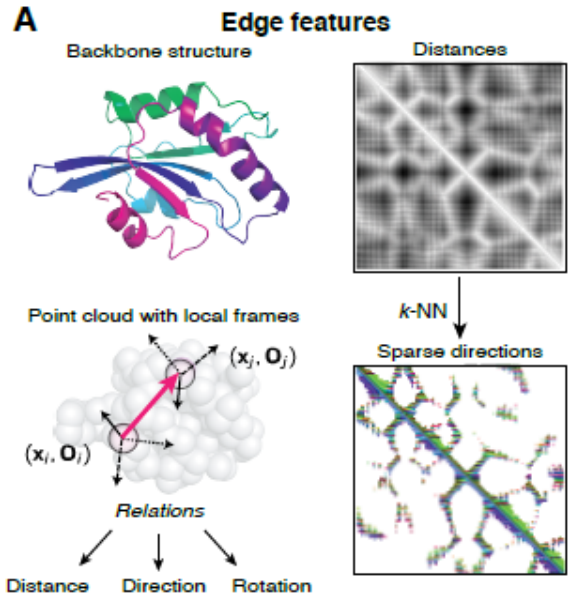


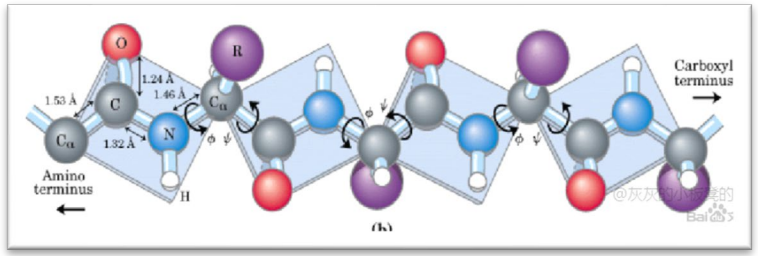
A graph-based, autoregressive model for protein sequences given 3D structures



A general-purpose protein design framework based on mining sequence-structure relationships in known protein structures(2018)

Notations





amino-acid backbone φ/ψ dihedral angle propensities
amino-acid backbone ω dihedral angle propensities

Rigid backbone features (e.g. distances and orientations)

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \quad \mathcal{V} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N\} \quad \mathcal{E} = \{\boldsymbol{e}_{ij}\}_{i \neq j} \quad \{\boldsymbol{x}_i \in \mathbb{R}^3 : 1 \leq i \leq N\}$$

edge encoding vector

$$\mathcal{E} = \{\boldsymbol{e}_{ij}\}_{i \neq j} \rightarrow \boldsymbol{e}_{ij}^{(s)}, \; \boldsymbol{e}_{ij}^{(p)}$$

Relative spatial encodings

$$\boldsymbol{e}_{ij}^{(s)} = \left(\mathrm{r}\left(\|\boldsymbol{x}_j - \boldsymbol{x}_i\|\right), \quad \boldsymbol{O}_i^T \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{\|\boldsymbol{x}_j - \boldsymbol{x}_i\|}, \quad \mathrm{q}\left(\boldsymbol{O}_i^T \boldsymbol{O}_j\right)\right)$$

Relative positional encodings

k-nearest neighbors

node features

$$(\phi_i, \psi_i, \omega_i)$$

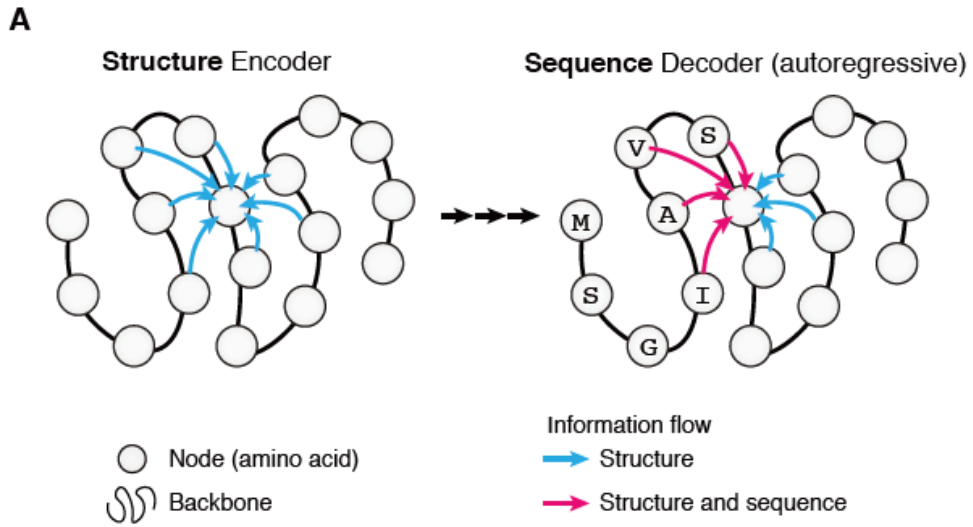
3-torus

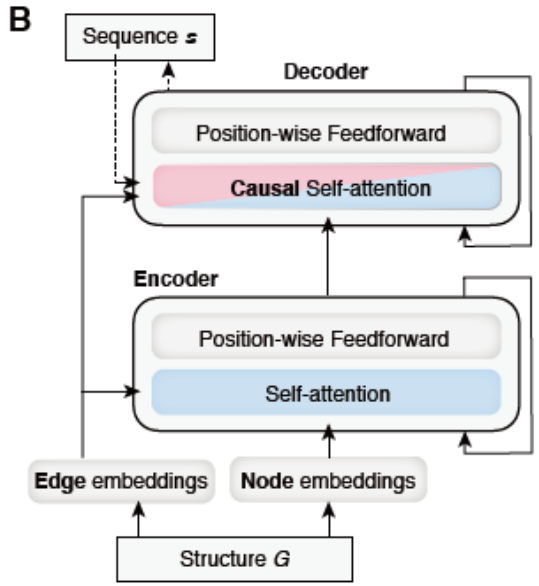
$$\{\sin, \cos\} \times (\phi_i, \psi_i, \omega_i)$$

Flexible backbone features (e.g. contacts, hydrogen bonds, and coarse backbone angles).

We combine the relative positional encodings with two binary edge features: contacts that indicate when the distance between C residues at i and j are hydrogen bonds which are directed and defined by the electrostatic less than 8 Angstroms and model of DSSP

The framework of Structured Transformer





Autoregressive decomposition

$$p(\boldsymbol{s}|\boldsymbol{x}) = \prod_i p(s_i|\boldsymbol{x}, \boldsymbol{s}_{<i}),$$

The framework of Structured Transformer

Encoder

$\boldsymbol{h}_i$ node embeddings

$$a_{ij}^{(\ell)} = \frac{\exp(m_{ij}^{(\ell)})}{\sum_{j' \in \mathrm{N}(i,k)} \exp(m_{ij'}^{(\ell)})}, \quad m_{ij}^{(\ell)} = \frac{\boldsymbol{q}_i^{(\ell)\top} \boldsymbol{z}_{ij}^{(\ell)}}{\sqrt{d}}$$

$$\boldsymbol{h}_i^{(\ell)} = \sum_{j \in N(i,k)} a_{ij}^{(\ell)} \boldsymbol{v}_{ij}^{(\ell)},$$

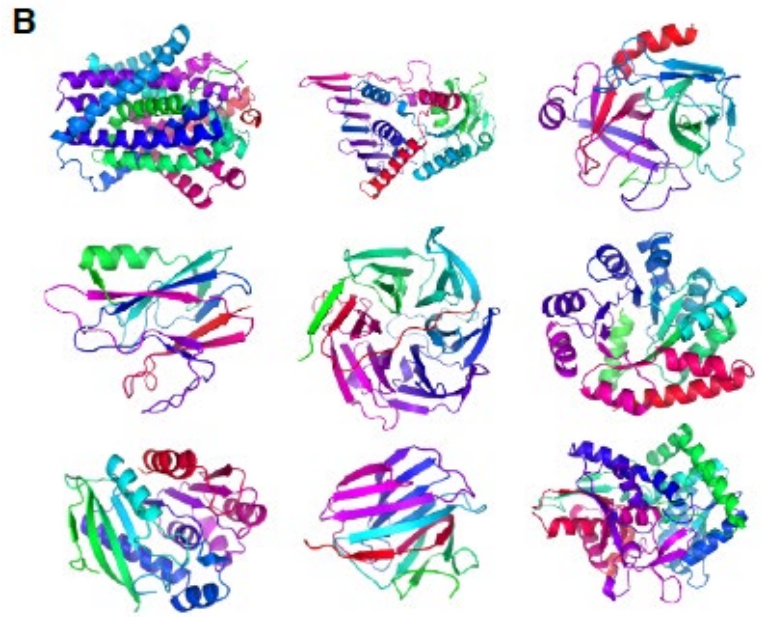
$$\Delta \boldsymbol{h}_i = \boldsymbol{W}_o \, \mathrm{Concat}\left(\boldsymbol{h}_i^{(1)}, \ldots, \boldsymbol{h}_i^{(L)}\right)$$

Decoder

$$\boldsymbol{r}_{ij}^{(\mathrm{dec})} = \begin{cases} (\boldsymbol{h}_j^{(\mathrm{dec})}, \boldsymbol{e}_{ij}, \mathrm{g}(\boldsymbol{s}_j)) & i > j \\ (\boldsymbol{h}_j^{(\mathrm{enc})}, \boldsymbol{e}_{ij}, \boldsymbol{0}) & i \leq j \end{cases}.$$

Dataset CATH 4.2

There is zero CAT overlap between these sets



Evaluation

1. Likelihood-based

   Test the ability of the generative model to give high likelihood to held out sequences.

2. Native sequence recovery

   Evaluate generated sequences vs the native sequences of templates.

3. Experimental comparison

   Compare the likelihoods of the model to high-throughput data from a de novo protein design experiment.

likelihood-based: Statistical comparison to likelihood-based models

Protein perplexities

Table 1: **Null perplexities** for common statistical models of proteins.

Null model	Perplexity	Conditioned on
Uniform	20.00	-
Natural frequencies	17.83	Random position in a natural protein
Pfam HMM profiles	11.64	Specific position in a specific protein family

The importance of structure & Improvement over deep profile-based methods

Test set	Short	Single chain	All
Structure-conditioned models			
Structured Transformer (ours)	**8.54**	**9.03**	**6.85**
SPIN2 [8]	12.11	12.61	-
Language models			
LSTM ($h = 128$)	16.06	16.38	17.13
LSTM ($h = 256$)	16.08	16.37	17.12
LSTM ($h = 512$)	15.98	16.38	17.13
Test set size	94	103	1120

Graph representations and attention mechanisms

Table 3: **Ablation of graph features and model components**. Test perplexities (lower is better).

Node features	Edge features	Aggregation	Short	Single chain	All
Rigid backbone					
Dihedrals	Distances, Orientations	Attention	8.54	9.03	6.85
Dihedrals	Distances, Orientations	PairMLP	**8.33**	**8.86**	**6.55**
C_α angles	Distances, Orientations	Attention	9.16	9.37	7.83
Dihedrals	Distances	Attention	9.11	9.63	7.87
Flexible backbone					
C_α angles	Contacts, Hydrogen bonds	Attention	11.71	11.81	11.51

native sequence recovery : Benchmarking protein redesign

Comparison to Rosetta

Method	Recovery (%)	Speed (AA/s) CPU	Speed (AA/s) GPU
Rosetta 3.10 `fixbb`	18.1	4.88×10^{-1}	N/A
Ours ($T = 0.1$)	**27.6**	$\mathbf{2.22 \times 10^{2}}$	$\mathbf{1.04 \times 10^{4}}$

(a) Single chain test set

Method	Recovery (%)
Rosetta, `fixbb` 1	33.1
Rosetta, `fixbb` 2	38.4
Ours ($T = 0.1$)	**39.2± 0.1**

(b) Ollikainen 40 benchmark

experimental comparison

Unsupervised anomaly detection for experimental protein design

Design	$\beta\beta\alpha\beta\beta_{37}$	$\beta\beta\alpha\beta\beta_{1498}$	$\beta\beta\alpha\beta\beta_{1702}$	$\beta\beta\alpha\beta\beta_{1716}$	$\alpha\beta\beta\alpha_{779}$
Rigid backbone	0.47	0.45	0.12	0.47	0.57
Flexible backbone	0.50	0.44	0.17	0.40	0.56

Design	$\alpha\beta\beta\alpha_{223}$	$\alpha\beta\beta\alpha_{726}$	$\alpha\beta\beta\alpha_{872}$	$\alpha\alpha\alpha_{134}$	$\alpha\alpha\alpha_{138}$
Rigid backbone	0.36	0.11	0.21	0.24	0.33
Flexible backbone	0.33	0.21	0.23	0.36	0.41

the performance is not dependent on precise 3D geometric features (e.g. distances and orientations) but can also be realized with coarse information (e.g. contacts, hydrogen bonds, and coarse backbone angles).

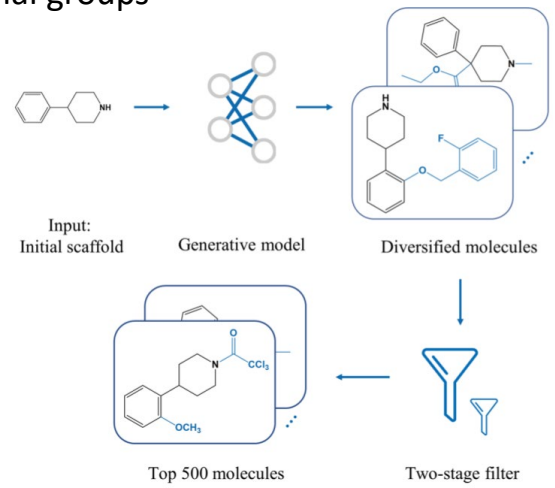
outline

1. Protein generation

2. Molecular generation based on Scaffold

Motivation

molecular scaffold & functional groups



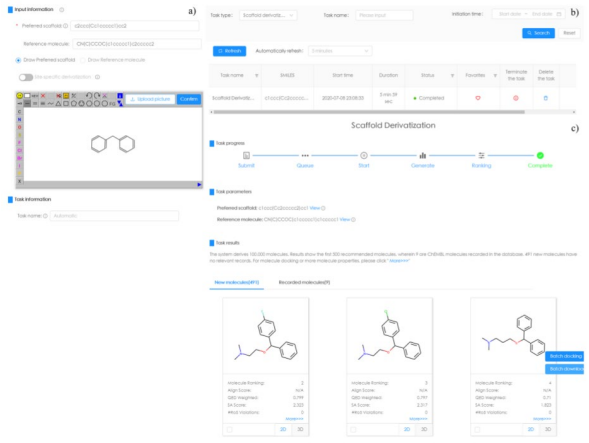


Figure 3. Illustration for the usage of AIScaffold: (a) inputs of the task, (b) task list interface, and (c) results of the scaffold diversification.

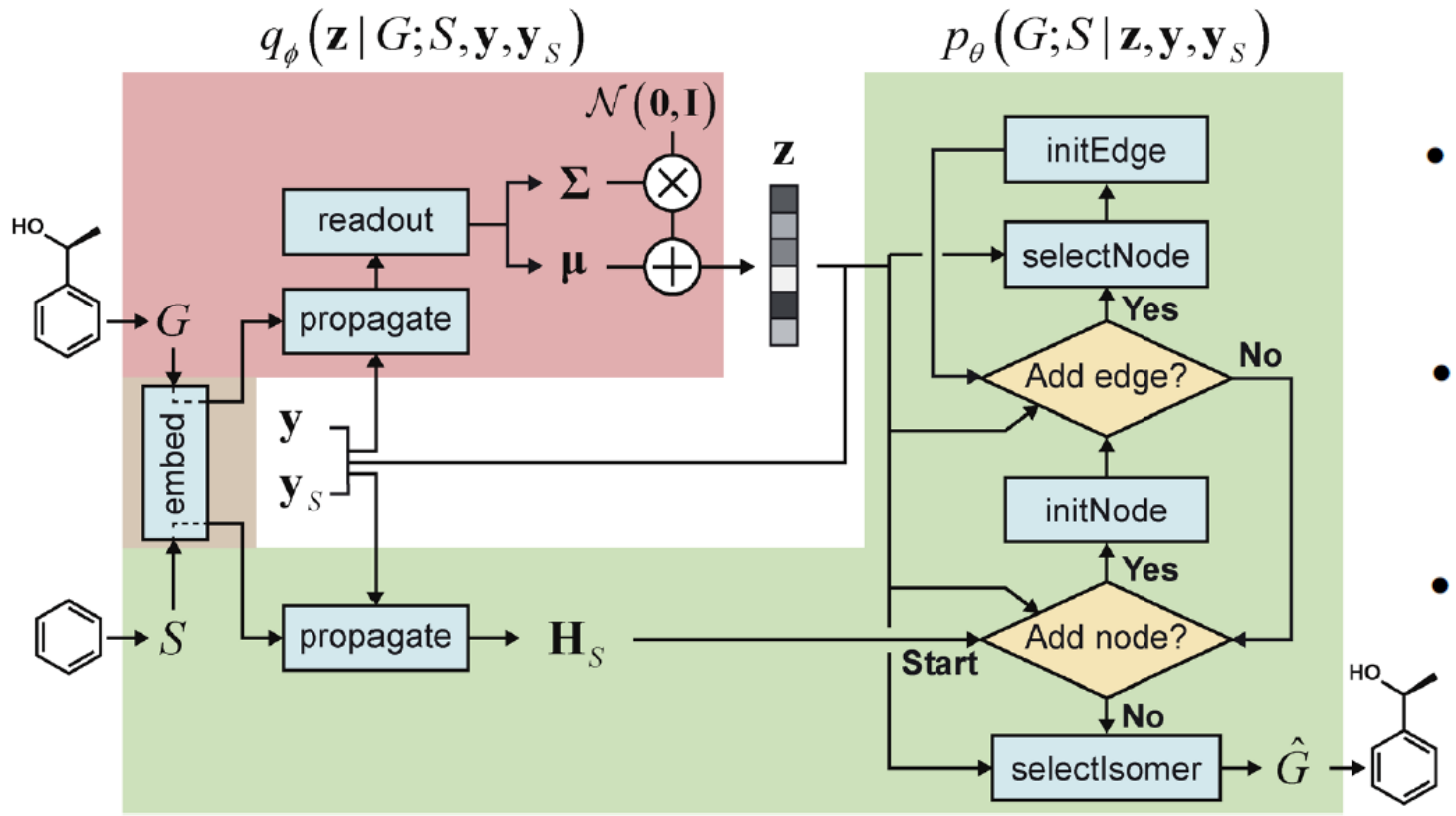
AIScaffold: A Web-Based Tool for Scaffold Diversification Using Deep Learning(2020)
Journal of Chemical Information and Modeling

Notations

Notation	Description
G	An arbitrary or whole-molecule graph
S	A molecular scaffold graph
$V(G)$	The node set of a graph G
$E(G)$	The edge set of a graph G
$\mathbf{h}_v$	A node feature vector
$\mathbf{h}_{uv}$	An edge feature vector
$\mathbf{H}_{V(G)}$	$\{\mathbf{h}_v : v \in V(G)\}$
$\mathbf{H}_{E(G)}$	$\{\mathbf{h}_{uv} : (u,v) \in E(G)\}$
$\mathbf{h}_G$	A readout vector summarizing $\mathbf{H}_{V(G)}$
$\mathbf{z}$	A latent vector to be decoded
$\mathbf{y}$	The vector of molecular properties of a whole-molecule
$\mathbf{y}_S$	The vector of molecular properties of a scaffold

Molecular weight (MW)
Topological polar surface area (TPSA)
Octanol-water partition coefficient (log P)

The framework



- Stage 1: node addition. estimated probabilities.
- Stage 2: edge addition. estimated probabilities.
- Stage 3: node selection. estimated probabilities.

If one desires to generate molecules with designated molecular properties, the corresponding property vectors y and y_S should be provided to condition the building process.

The framework

Encoder--Graph encoding

propagate

$$(\mathbf{H}_{V(G)}, \mathbf{H}_{E(G)}) = \text{embed}\,(G)$$

$$\mathbf{H}'_{V(G)} = propagate\left(\mathbf{H}_{V(G)}, \mathbf{H}_{E(G)}\right)$$

$$\mathbf{m}_v = \sum_{u:(u,v)\in E(G)} M(\mathbf{h}_u, \mathbf{h}_v, \mathbf{h}_{uv}) \quad \forall v \in V(G)$$

$$\mathbf{h}'_v = U(\mathbf{m}_v, \mathbf{h}_v) \quad \forall v \in V(G)$$

readout

$$\mathbf{h}_G = readout\left(\mathbf{H}'_{V(G)}\right).$$

$$\boldsymbol{\mu}_G = \text{MLP}^{\mu}\,(\mathbf{h}_G)$$
$$\boldsymbol{\sigma}_G = \exp\{\text{MLP}^{\sigma}\,(\mathbf{h}_G)/2\}$$
$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$
$$\mathbf{z} = \boldsymbol{\mu}_G + \sigma_G \odot \boldsymbol{\epsilon},$$

Decoder--Graph decoding

$$\hat{\mathbf{p}}^{\text{an}} = addNode(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z})$$

$$\hat{\mathbf{p}}^{\text{ae}} = addEdge(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z})$$

$$\hat{\mathbf{p}}^{\text{sn}} = selectNode(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z}).$$

$$\mathbf{h}_w = initNode(\mathbf{h}^0_w, \mathbf{H}_{V(G_t)})$$

$$\mathbf{h}_{vw} = initEdge(\mathbf{h}^0_{vw}, \mathbf{H}_{V(G_t)} \cup \mathbf{h}_w).$$

$$addNode(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z}) = f \circ \text{concat}(readout \circ propagate^{(k)}\,(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z}), \mathbf{z})$$

G_t to denote any transient (or completed) graph constructed from G_0.

Isomer selection

$$\hat{\mathbf{p}}^{\text{si}} = selectIsomer(\hat{G}, \mathbf{z}),$$

each isomeric graph $I \in \mathcal{I}(G)$.

$$(\mathbf{H}_{V(I)}, \mathbf{H}_{E(I)}) = \text{embed}\,(I)$$
$$\mathbf{H}'_{V(I)} = \text{propagate}^{(k)}\,(\mathbf{H}_{V(I)}, \mathbf{H}_{E(I)}, \mathbf{z})$$
$$\mathbf{h}_I = \frac{1}{|V(I)|} \sum_{v \in V(I)} \mathbf{h}'_v$$
$$\hat{p}^{si}_I = \sigma \circ \text{MLP}^{s} \circ \text{concat}\,(\mathbf{h}_I, \mathbf{z})\,.$$

Supplementary files

Algorithm 1 Scaffold-based graph generation

Inputs: G, S, $\mathbf{y}$, $\mathbf{y}_S$ ▷ Whole/scaffold graphs and properties

$G_0 \leftarrow S$

$\tilde{\mathbf{y}} \leftarrow \text{concat}(\mathbf{y}, \mathbf{y}_S)$

if $G \neq (\emptyset, \emptyset)$ **then** ▷ Learning phase

  $(\mathbf{H}_{V(G)}, \mathbf{H}_{E(G)}) \leftarrow \text{embed}(G)$

  $\mathbf{H}_{V(G)} \leftarrow \text{propagate}^{(k)}(\mathbf{H}_{V(G)}, \mathbf{H}_{E(G)}, \tilde{\mathbf{y}})$

  $\mathbf{z} \sim \text{reparam} \circ \text{readout}(\mathbf{H}_{V(G)})$ ▷ Vector representation of the target graph

else

  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Generation phase

end if

$\tilde{\mathbf{z}} \leftarrow \text{concat}(\mathbf{z}, \tilde{\mathbf{y}})$

$(\mathbf{H}_{V(G_0)}, \mathbf{H}_{E(G_0)}) \leftarrow \text{embed}(G_0)$ ▷ Node and edge feature vectors

$\mathbf{H}_{V(G_0)} \leftarrow \text{propagate}^{(k)}(\mathbf{H}_{V(G_0)}, \mathbf{H}_{E(G_0)}, \tilde{\mathbf{y}})$ ▷ Initial update of the scaffold nodes

$t \leftarrow 1$ ▷ Node addition counter

$v_t \sim \text{Cat} \circ \text{addNode}(\mathbf{H}_{V(G_{t-1})}, \mathbf{H}_{E(G_{t-1})}, \tilde{\mathbf{z}})$ ▷ Sample a node type or STOP

while $v_t \neq \text{STOP}$ **do**

  $V(G_t) \leftarrow V(G_{t-1}) \cup \{v_t\}$ ▷ Add the new node

  $\mathbf{H}_{V(G_t)} \leftarrow \mathbf{H}_{V(G_{t-1})} \cup \{\text{initNode}(v_t, \mathbf{H}_{V(G_{t-1})})\}$ ▷ Initialize and add a new node vector

  $E_{t,0} \leftarrow E(G_{t-1})$; $\mathbf{H}_{E_{t,0}} \leftarrow \mathbf{H}_{E(G_{t-1})}$ ▷ Prepare edge additions

  $i \leftarrow 1$ ▷ Edge addition counter

  $e_{t,i} \sim \text{Cat} \circ \text{addEdge}(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E_{t,i-1}}, \tilde{\mathbf{z}})$ ▷ Sample an edge type or STOP

  while $e_{t,i} \neq \text{STOP}$ **do**

    $v_{t,i} \sim \text{Cat} \circ \text{selectNode}(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E_{t,i-1}}, \tilde{\mathbf{z}})$ ▷ Sample a node to connect

    $E_{t,i} \leftarrow E_{t,i-1} \cup \{(v_t, v_{t,i})\}$ ▷ Add the new edge (with type $e_{t,i}$)

    $\mathbf{H}_{E_{t,i}} \leftarrow \mathbf{H}_{E_{t,i-1}} \cup \{\text{initEdge}(e_{t,i}, \mathbf{H}_{V(G_t)})\}$ ▷ Initialize and add a new edge vector

    $i \leftarrow i + 1$

    $e_{t,i} \sim \text{Cat} \circ \text{addEdge}(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E_{t,i-1}}, \tilde{\mathbf{z}})$ ▷ Sample a next edge type or STOP

  end while

  $\mathbf{H}_{E(G_t)} \leftarrow \mathbf{H}_{E_{t,i-1}}$

  $E(G_t) \leftarrow E_{t,i-1}$

  $G_t \leftarrow (V(G_t), E(G_t))$

  $t \leftarrow t + 1$

  $v_t \sim \text{Cat} \circ \text{addNode}(\mathbf{H}_{V(G_{t-1})}, \mathbf{H}_{E(G_{t-1})}, \tilde{\mathbf{z}})$ ▷ Sample a next node type or STOP

end while

$G_t^* \sim \text{Cat} \circ \text{selectIsomer}(G_t, \tilde{\mathbf{z}})$ ▷ Assign the stereoisomerism

return G_t^*

Concat denotes the vector concatenation

Cat denotes a categorical distribution

∘ denotes the function composition

$$\hat{\mathbf{p}}^{an} = \text{addNode}(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z})$$
$$= \text{softmax} \circ \text{MLP}^{an} \circ \text{concat}\left(\text{readout} \circ \text{propagate}^{(k)}(\mathbf{H}_{V(G_t)}, \mathbf{H}_{E(G_t)}, \mathbf{z}), \mathbf{z}\right),$$

$$i \sim \text{Cat}(\hat{\mathbf{p}}^{an}) \quad (1 \leq i \leq |\mathcal{A}| + 1).$$

If i ≤ |A|, the model adds a new node with the i-th chemical element Ai , or else the building process terminates

In the present work, we used the atom types in an indexed family *A* = (*Ai*) = (C, N, O, F, P, S, Cl, Br) and the bond types in another indexed family *B* = (*Bi*) = (single-bond, double-done, triple-bond)

The model assigns only the basic types in A and B during graph building and specifies stereoisomerism at the final stage of generation

Objective function

$$\log p(G;S) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi}\left[\log p_\theta(G;S|\mathbf{z})\right] - D_{\mathrm{KL}}\left[q_\phi(\mathbf{z}|G;S)\|p(\mathbf{z})\right]$$

$D_{\mathrm{KL}}[\cdot\|\cdot]$ Kullback–Leibler divergence

$p(\mathbf{z})$ The prior distribution, defined to be standard normal

explicit constraint G is a supergraph of S.

Supplementary files

(S_i, G_i)

To describe the graph building loss, let us express the stepwise transitions from S_i to G_i by a finite sequence $(G_{i,0}, G_{i,1}, \cdots, G_{i,T})$, where $G_{i,0} = S_i$ and $G_{i,T} = G_i$. The transition from $G_{i,t}$ to $G_{i,t+1}$ conforms to the true probability vector $\mathbf{p}_{i,t}$, which has one unity value for the correct building action and zeros for the others. During learning, the model reconstructs each G_i from S_i by estimating a sequence $(\hat{\mathbf{p}}_{i,0}, \hat{\mathbf{p}}_{i,1}, \cdots, \hat{\mathbf{p}}_{i,T-1})$. Then the individual graph building loss can be defined by

$$l_i^{\mathrm{build}} = -\sum_t \mathbf{p}_{i,t} \cdot \log\left(\hat{\mathbf{p}}_{i,t}\right). \tag{25}$$

$$l_i^{\mathrm{build}} = -\sum_t \mathbf{p}_{i,t} \cdot \log\left(\hat{\mathbf{p}}_{i,t}\right)$$

$$l_i^{\mathrm{isomer}} = -\sum_{I\in\mathcal{I}(G_i)} \left(p_I^{si}\log\left(\hat{p}_I^{si}\right) + \left(1-p_I^{si}\right)\log\left(1-\hat{p}_I^{si}\right)\right)$$

$$l_i^{\mathrm{KL}} = -\frac{1}{2}\sum_j \left(1+\log\left(\sigma_{G_i,j}^2\right) - \mu_{G_i,j}^2 - \sigma_{G_i,j}^2\right)$$

$\mu_{G_i,j}$ and $\sigma_{G_i,j}$ are the j-th elements of $\boldsymbol{\mu}_{G_i}$ and $\boldsymbol{\sigma}_{G_i}$.

$$\sum_i \left(l_i^{\mathrm{build}} + l_i^{\mathrm{isomer}} + \beta l_i^{\mathrm{KL}}\right)$$

Validity, uniqueness and novelty analysis

$$\text{Validity} = \frac{\#\ \text{of valid graphs}}{\#\ \text{of generated graphs}}$$

$$\text{Uniqueness} = \frac{\#\ \text{of non-duplicate, valid graphs}}{\#\ \text{of valid graphs}}$$

$$\text{Novelty} = \frac{\#\ \text{of unique graphs not in the training set}}{\#\ \text{of unique graphs}}.$$

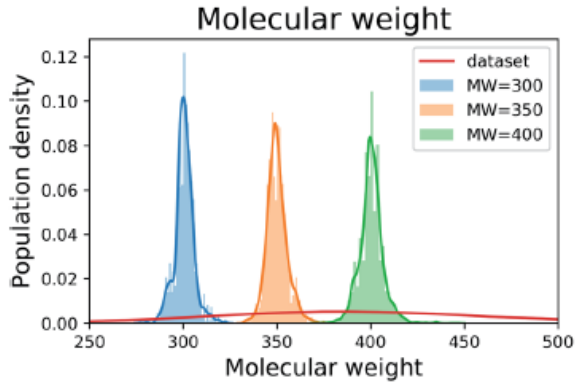
MAD : The mean absolute difference between designated property values and the property values of generated molecules

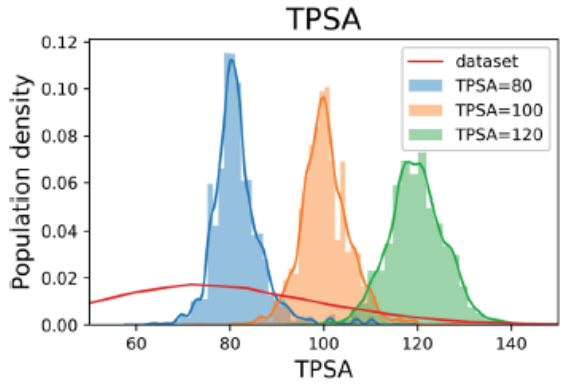
Define a graph to be valid if it satisfies basic chemical requirements such as valency (RDKit can determine the validity of generated graphs)

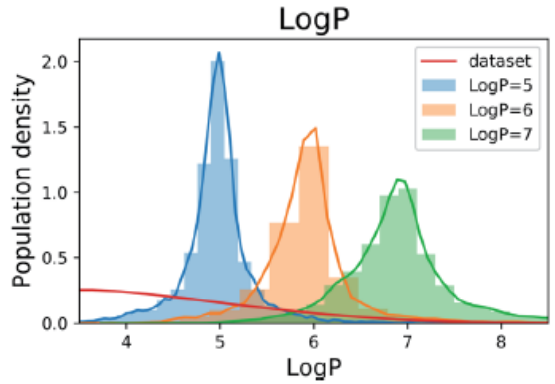
Result

Table 2 Validity, uniqueness and novelty of the molecules generated by our model, and the results from other molecular generative models

Model	Validity (%)	Uniqueness (%)	Novelty (%)
Ours (MW)	98.6	85.4	98.7
Ours (TPSA)	93.0	84.9	99.1
Ours (log *P*)	97.8	86.4	99.3
GraphVAE[31]	55.7	87.0	61.6
MolGAN[31]	98.1	10.4	94.2
JTVAE[24]	100.0	—	—
MolMP[27]	95.2–97.0	—	91.2–95.1
SMILES VAE[27]	80.4	—	79.3
SMILES RNN[27]	93.2	—	89.9







This shows that despite the narrowed search space, our model successfully generated new molecules having desired properties.

Result

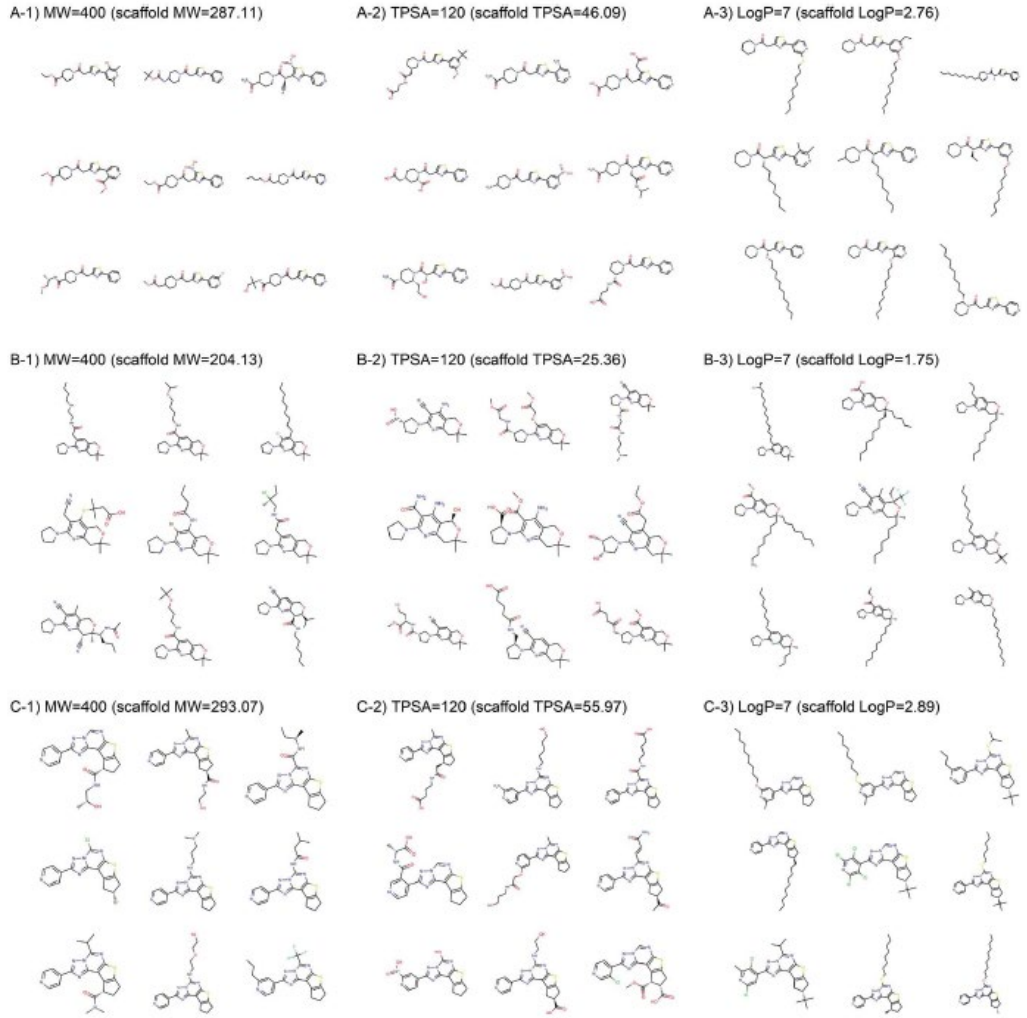


Fig. 3 Example molecules generated from three scaffolds. The indicated values are the target conditions of the generation and the property values of the scaffolds.

Property	Validity (%)	Uniqueness (%)	MAD
MW (seen scaffolds)	98.4	88.6	6.72
MW (unseen scaffolds)	98.4	83.5	6.09
TPSA (seen scaffolds)	93.2	87.0	8.32
TPSA (unseen scaffolds)	92.5	82.9	9.82
log *P* (seen scaffolds)	98.2	91.1	0.28
log *P* (unseen scaffolds)	97.1	87.0	0.36

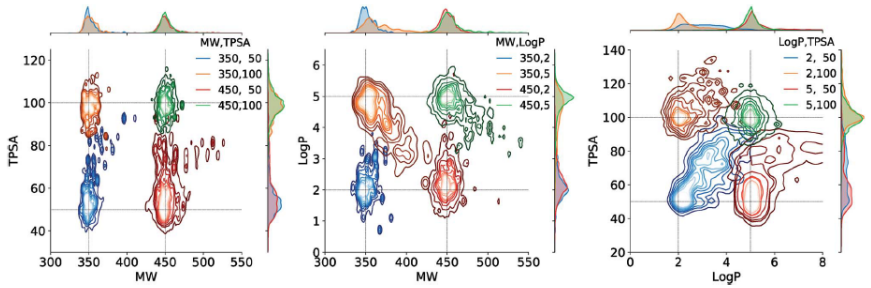


Fig. 4 Estimated joint distributions of the property values of generated molecules. The legends show the target values used for the generations. In all distributions, the innermost contour encloses 10%, the outermost encloses 90%, and each n-th in the middle encloses $n \times 10\%$ of the population. On the upper and right ends of each plot are the marginal distributions of the properties on horizontal and vertical axes, respectively.

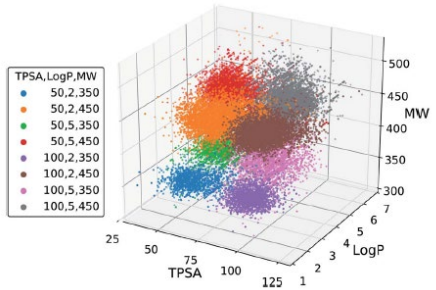


Fig. 5 Scatter plot of the property values of generated molecules. The legend lists the eight sets of property values used for the generations.

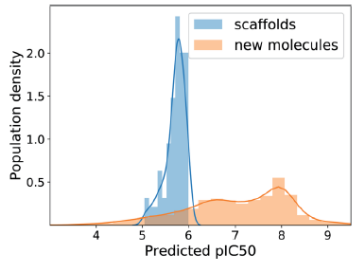


Fig. 6 Distributions of human-EGFR pIC_{50} values predicted for the test scaffolds and generated molecules in our semi-supervised learning experiment. We extended our model into a semi-supervised VAE by adding a property predictor.

Octanol-water partition coefficient (log P)—hydrophobic groups
Topological polar surface area (TPSA)—polar functional groups

Result

Table 4 Statistical comparison of the performance on single-, double- and triple-property controls

Properties	MW MAD	TPSA MAD	log *P* MAD	Validity (%)	Novelty (%)
MW	7.99	—	—	98.6	98.7
TPSA	—	8.57	—	93.0	99.1
log *P*	—	—	0.29	97.8	99.3
MW & TPSA	8.04	7.06	—	93.5	99.4
MW & log *P*	11.59	—	0.45	97.0	99.6
TPSA & log *P*	—	9.62	0.60	94.5	99.6
All	16.23	10.95	0.73	93.9	99.8