Cost-aware Graph Generation: A Deep Bayesian Optimization Approach

Jiaxu Cui [1,2], Bo Yang [1,2], Bingyi Sun [1,2,4], Jiming Liu [3]

[1] College of Computer Science and Technology, Jilin University, China

[2] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, China

[3] Department of Computer Science, Hong Kong Baptist University, Hong Kong

[4] National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, China

jxcui16@mails.jlu.edu.cn, ybo@jlu.edu.cn, bysun15@mails.jlu.edu.cn, jiming@comp.hkbu.edu.hk

Motivation

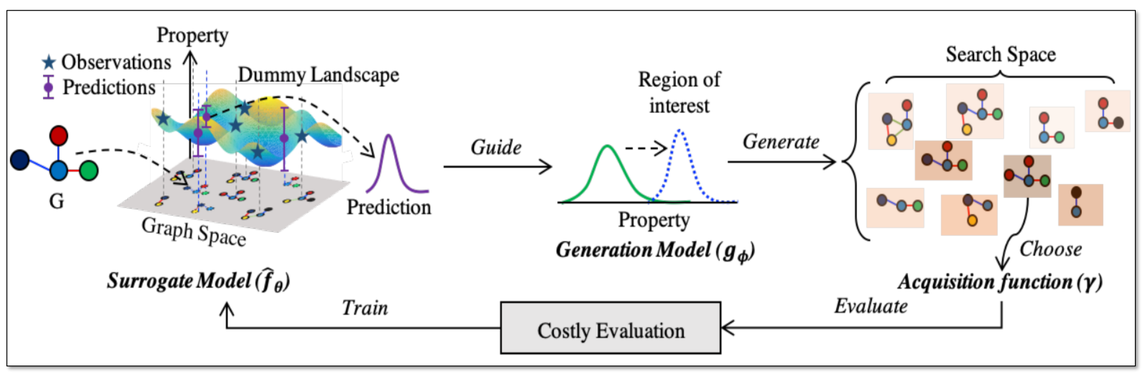
- Goal-oriented graph generation models can produce optimal graphs for given objectives, beyond learning the distribution of an existing dataset. They can help many practical and challenging applications, such as discovering the molecules with the best drug characteristics and designing neural architectures with the excellent performance.
- Current models still need many evaluations.
- Graph evaluations are usually very expensive in terms of computation resource, time, money, energy, and environment.
- Such these high costs will become a bottleneck in practical applications.

Goal

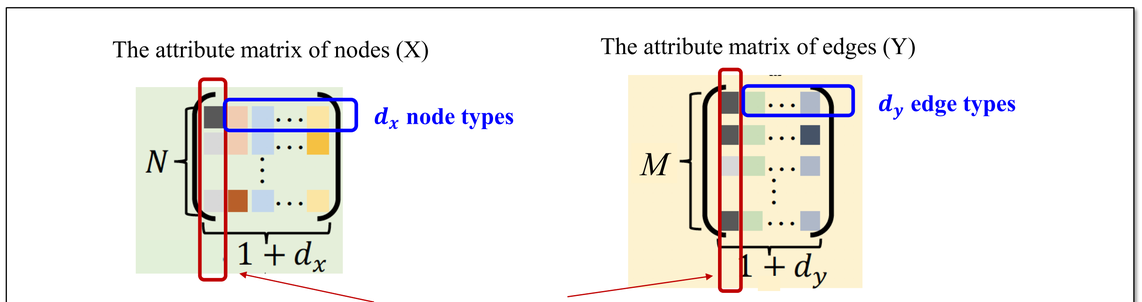
- Goal: to generate the optimal graphs at as low cost as possible.

Solution

- Main idea is to bring the advantage of Bayesian optimization to the goal-oriented graph generation task.
- Proposed framework: Cost-Aware Graph Generation (CAGG)



- Representation of graphs: A graph G with d_x node types and d_y edge types as consisting of four tuples (V, E, X, Y), where V is a set of nodes, E⊆(V×V) is a set of edges, and $X \subseteq \mathbb{R}^{|V|\times(1+d_x)}$ and $Y \subseteq \mathbb{R}^{|E|\times(1+d_y)}$ are the attribute matrices of all nodes and edges.



- Surrogate model:
  - ✓ Should avoid the hand-crafted kernel
  - ✓ Should have the ability to approach real expensive-to-evaluate black-box function under a small number of evaluations

We propose a Bayesian graph neural network as the surrogate.

Embedding layer: $F_e^{(0)} = (1 - Y_{e,0}) \times \psi_E^{(em)}(Y_{e,1:}),$
$H_i^{(0)} = (1 - X_{i,0}) \times \psi_V^{(em)}(X_{i,1:}),$

GNN layer: $F_e^{(t)} = (1 - Y_{e,0}) \times \psi_E^{(gn)}([F_e^{(t-1)}, H_i^{(t-1)}, H_j^{(t-1)}]),$
$H_i^{(t)} = (1 - X_{i,0}) \times \psi_V^{(gn)}([H_i^{(t-1)}, \sum_{e\in\aleph(i)} F_e^{(t-1)}]),$

Pooling layer: $\boldsymbol{h}_G = concat(\sum_{i\in V} H_i^{(t)} \mid t = 1, 2, ..., T),$

Readout layer: a MLP

The predictive distribution for a new graph G':

$$p(z' \mid G', \mathcal{D}) = \int p(z' \mid G', \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathcal{D}) d\boldsymbol{\theta},$$

Use the Monte Carlo dropout technology to deal with the integral:

$$\mathcal{N}(\mu(G'), \sigma^2(G')) \begin{cases} \mu(G') \approx \frac{1}{S}\sum_{i=1}^{S} \hat{f}_{\theta_i}(G'), \\ \sigma^2(G') \approx \frac{1}{S}\sum_{i=1}^{S} (\hat{f}_{\theta_i}(G') - \mu(G'))^2. \end{cases}$$

Surrogate model: Bayesian graph neural network

We test the proposed Bayesian graph neural network in the small sample setting.

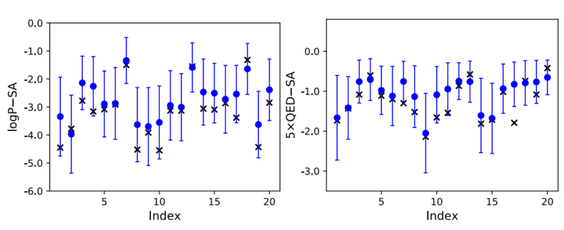


Figure S1: Visualization of predictions of the proposed surrogate model on 20 test samples randomly extracted from the QM9 dataset (Ramakrishnan et al. 2014). Black crosses denote the groundtruth and blue dots with error bars denote the predictions with a 95% confidence interval.

Good predictions and ranking

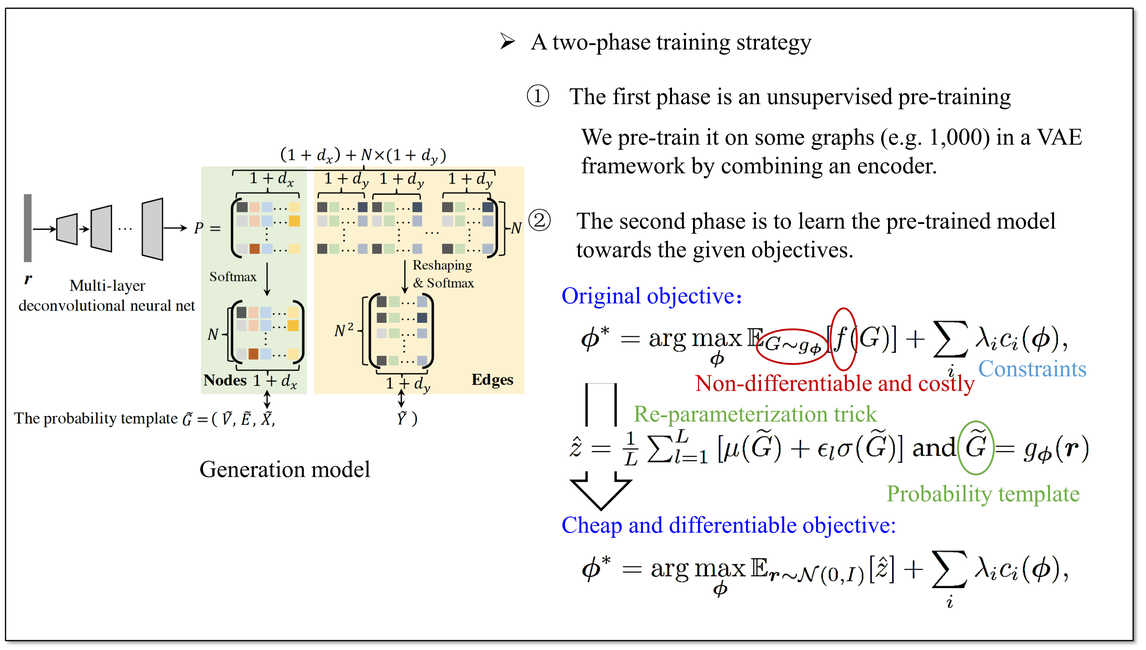
Surrogates	logP−SA	5×QED−SA
GNN-BLR	-1.752±2.283	-3.806±3.088
Surrogate of the CAGG	**-1.086±0.069**	**-1.004±0.027**

Table S1: Predictive performance with 20-fold cross validation of the surrogate model (GNN-BLR) proposed in DGBO (Cui, Yang, and Hu 2019) and our surrogate model, measured by log-likelihood (larger is better).

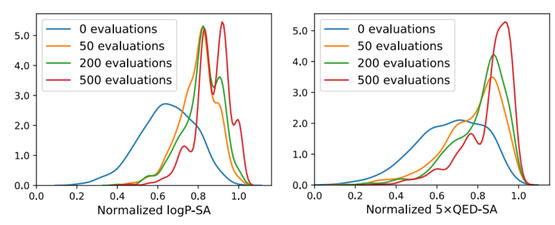
Better predictive performance

- Generation model
  - ✓ Produce desirable graphs as candidates.
  - ✓ Constrain the search space to contain more good graphs, so as to avoid unnecessary evaluations.

We use a multi-layer deconvolutional net as generation model and design a two-phase training strategy



We test the trained generation model: Can good graphs be generated as search space?



Shift to the desired direction as the optimization goes on

Methods	logP−SA		5×QED−SA	
	# Eval	CSP	# Eval	CSP
CAGG w/o Pre	443	71.12%	453	68.21%
CAGG w/o GO	272	52.94%	262	45.04%
CAGG	**128**	N/A	**144**	N/A

Table S2: An ablation study on the two-phase training strategy in the generation model. CAGG w/o Pre means a variant without unsupervised VAE pre-training; that is, it does not execute the lines 1-2 of Algorithm 1 in the main text. CAGG w/o GO means a variant without goal-oriented training while generating search space; that is, it does not execute the line 6 of Algorithm 1 in the main text (i.e., g_ϕ is always the same as the pre-trained g_{ori}). # Eval means the number of evaluations to find the optimum. CSP means the Cost Saving Percentage of our framework over other variants.

Both phases contribute to reducing costs.

- Acquisition function
  - ✓ We use Expected Improvement (EI) function.

Experiments

- Baselines and Settings
  - ✓ Representative and state-of-the-art baselines
  - ✓ Same hardware environment equipped with a four-core Intel i5 processor
- Results on two challenging applications, including MOLECULAR DISCOVERY and NEURAL ARCHITECTURE SEARCH
  - ✓ **Our method finds the comparable or optimal solution**
  - ✓ **Our method reduces the evaluation cost significantly (30%-95%).**

Goals	Methods	# Eval	Algorithm cost (hours)	Evaluation cost (hours)	Total cost: Hours	Total cost: CO_2e (lbs)	Total cost: Google Cloud Platform	CSP
logP−SA	Gentrl	3,000	4.3	3,000	3,004.3	412.1	US$1,254.6∼US$1616.3	95.70%
	GCPN	3,000	**0.2**	3,000	3,000.2	411.5	US$1,252.9∼US$1614.1	95.69%
	JTVAEBO	3,000	22.5	3,000	3,022.5	414.6	US$1,262.2∼US$1626.1	**95.73%**
	G2G	1,600	2.8	1,600	1602.8	219.8	US$669.3∼US$862.3	91.94%
	DGBO	189	0.3	189	189.3	26.0	US$79.1∼US$101.8	31.75%
	CAGG (ours)	**128**	1.2	**128**	**129.2**	**17.7**	**US$54.0∼US$69.6**	N/A
5×QED−SA	Gentrl	3,000	4.3	3,000	3,004.3	412.1	US$1,254.6∼US$1616.3	**95.16%**
	GCPN	3,000	**0.2**	3,000	3,000.2	411.5	US$1,252.9∼US$1614.1	**95.16%**
	JTVAEBO	1,550	21.5	1,550	1,571.5	215.6	US$656.3∼US$845.5	90.75%
	G2G	1,600	2.8	1,600	1602.8	219.8	US$669.3∼US$862.3	90.93%
	DGBO	448	1.0	448	449.0	61.6	US$187.5∼US$241.6	67.64%
	CAGG (ours)	**144**	1.3	**144**	**145.3**	**19.9**	**US$60.7∼US$78.2**	N/A

Table 1: Comparison of cost with molecular discovery methods. # Eval means the number of evaluations to find the optimal solution (lower is better). We set the maximum # Eval to 3,000. Algorithm cost represents the algorithm execution time, where the Gentrl, JTVAEBO, and CAGG contain the running time in pre-training and designing, the G2G only includes the training time, and both the DGBO and GCPN include only running time in searching or designing, because they do not require pre-training. Evaluation cost represents the cost of evaluating molecules, which is calculated based on # Eval and an hour for DFT calculation per molecular evaluation (Gilmer et al. 2017). CO_2e is the estimated CO_2 emission, which is calculated based on the carbon emission estimation model (Strubell, Ganesh, and McCallum 2019). Google Cloud Platform cost is calculated based on the price of on-demand c2-standard-8 instances. CSP means the Cost Saving Percentage of the CAGG over other baselines.

Methods	Total cost	CIFAR100	ImageNet16-120
ResNet	N/A	70.86	43.63
RS	205 hours	72.48±1.04	46.04±0.46
REINFORCE	205 hours	72.48±0.31	45.85±0.51
REA	205 hours	73.09±0.25	46.12±0.67
	50.3 hours	72.87±0.27	46.13±0.46
CAGG (ours)	140.9 hours	73.25±0.42	46.34±0.27
	201.5 hours	**73.38±0.16**	**46.37±0.24**

Table 2: Comparison of cost and classification accuracy with baselines for cell-based NAS. The total cost includes the algorithm execution time and evaluation costs. The evaluation cost per architecture is assigned to half an hour, which is estimated based on the running time on a personal computer. The last two columns show the test classification accuracy (%). All methods ran five times to eliminate random effects. We set the budget to 205 hours for all baselines and report the results found by the CAGG under various total costs.

Methods	Total cost	Indoor	Slice
RAND	12 hours	0.156±0.023	0.932±0.044
TreeBO	12 hours	0.168±0.023	0.759±0.079
NASBOT	12 hours	0.114±0.009	0.615±0.044
Auto-Keras	12 hours	0.112±0.010	0.870±0.054
NASGBO	12 hours	0.090±0.012	0.560±0.046
	4 hours	0.072±0.003	0.788±0.003
CAGG (ours)	8 hours	0.066±0.002	0.625±0.001
	12 hours	**0.063±0.001**	**0.433±0.010**

Table 3: Comparison of cost and the test regression mean squared error (lower is better) with baselines for multi-branch NAS. We set the budget to 12 hours and report the results found by the CAGG under 4, 8, and 12 hours.

Limitations and Future Work

- How to relax the limit of the pre-fixed maximum number of nodes when generating graphs
- How to introduce the cost difference between graphs to further reduce the cost
- Handling other complex generative tasks and multi-objective situation are also promising extensions

35th AAAI conference on Artificial intelligence