Multi-modal Molecule Structure-text Model for Text-based Retrieval and Editing

Shengchao Liu[1,2], Weili Nie[3], Chengpeng Wang[4], Jiarui Lu[1,2], Zhuoran Qiao[5], Ling Liu[6], Jian Tang[*1,7], Chaowei Xiao[*3,8], and Anima Anandkumar[*3,5]

1. <u>Introduction</u> –

- Existing methods –
    1. Focus on 2D/3D geometries, OR
    2. Involve expensive supervision, OR
    3. Avoid multi-modality in pre-training, OR
    4. Involve limited multi-modality using SMILES.

- Motivation, Text helps a lot!

2. <u>Brief results</u> –

- Developed a multi-modal foundation model.
- Model has **strong** zero-shot generalization capabilities to unseen tasks!
- Also create a dataset, PubChemSTM.
- SOTA performance.

3. Method –

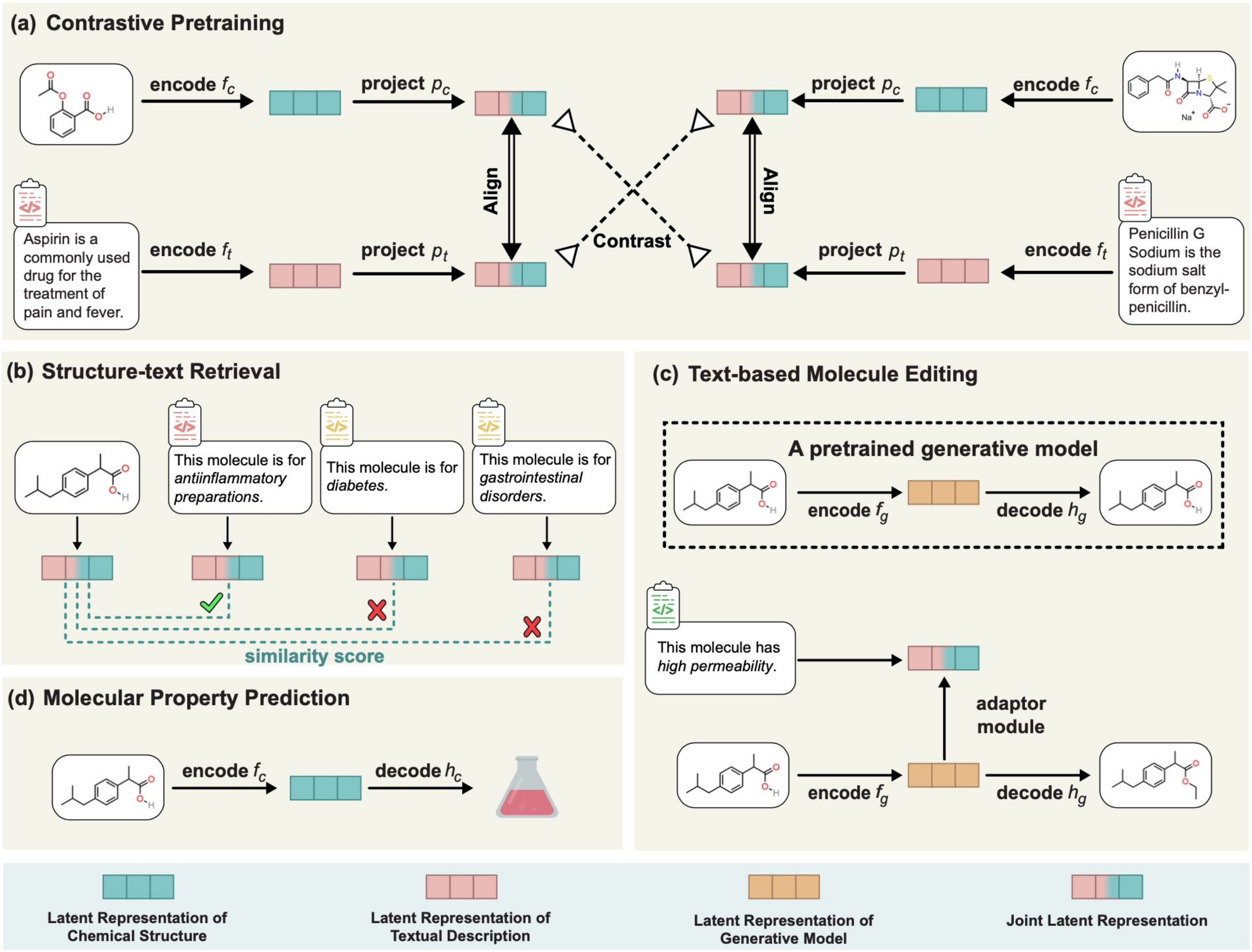


Table 4. Model specifications. # parameters in each model.

Branch	Model	# parameters
Chemical structure	MegaMolBART	10,010,635
	GIN	1,885,206
Textual description	SciBERT	109,918,464

3. *Method* –

- Loss functions –

$$\mathcal{L} = -\frac{1}{2}\left(\mathbb{E}_{\boldsymbol{x}_c,\boldsymbol{x}_t}\left[\log \sigma(E(\boldsymbol{x}_c,\boldsymbol{x}_t)\right] + \mathbb{E}_{\boldsymbol{x}_c,\boldsymbol{x}'_t}\left[\log(1-\sigma(E(\boldsymbol{x}_c,\boldsymbol{x}'_t))\right]\right) - \frac{1}{2}\left(\mathbb{E}_{\boldsymbol{x}_c,\boldsymbol{x}_t}\left[\log \sigma(E(\boldsymbol{x}_c,\boldsymbol{x}_t)\right] + \mathbb{E}_{\boldsymbol{x}'_c,\boldsymbol{x}_t}\left[\log(1-\sigma(E(\boldsymbol{x}'_c,\boldsymbol{x}_t))\right]\right)$$

$$\mathcal{L} = -\frac{1}{2}\mathbb{E}\Big[\log \frac{\exp(E(\boldsymbol{x}_c,\boldsymbol{x}_t))}{\exp(E(\boldsymbol{x}_c,\boldsymbol{x}_t)) + \sum_{\boldsymbol{x}_{t'}} \exp(E(\boldsymbol{x}_c,\boldsymbol{x}_{t'}))} + \log \frac{\exp(E(\boldsymbol{x}_c,\boldsymbol{x}_t))}{\exp(E(\boldsymbol{x}_c,\boldsymbol{x}_t)) + \sum_{\boldsymbol{x}_{c'}} \exp(E(\boldsymbol{x}_{c'},\boldsymbol{x}_t))}\Big]$$

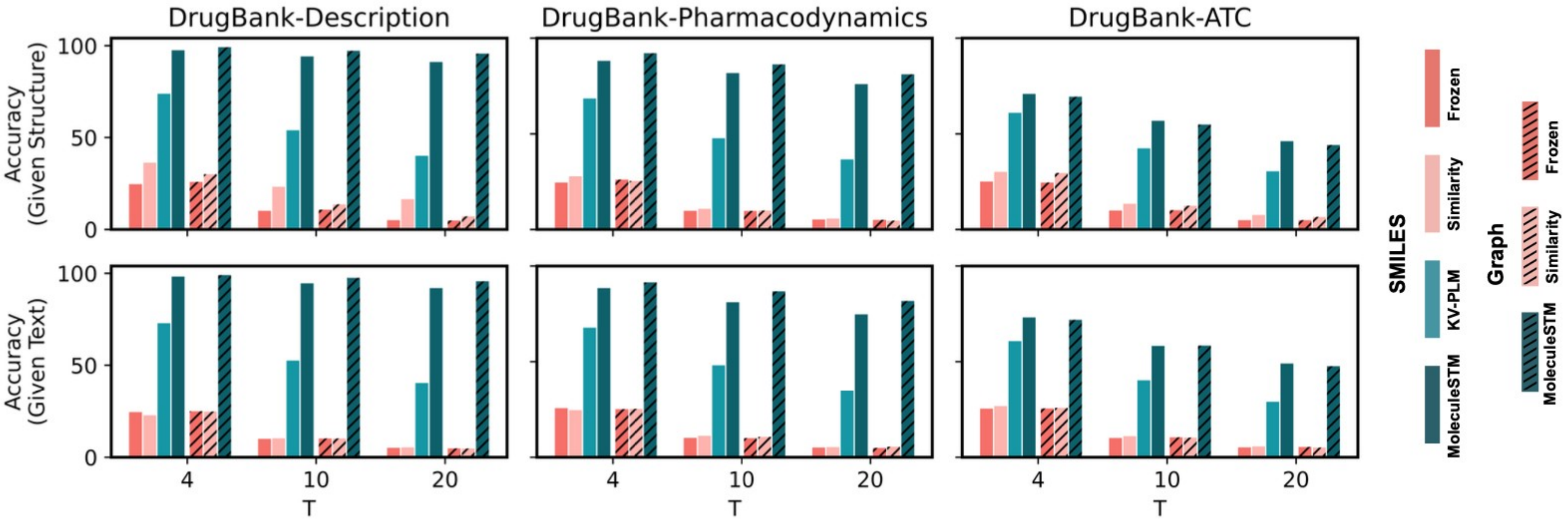
4. *Tasks* –

- Zero-shot structure-text retrieval,
- Zero-shot text-based molecule editing,
- Molecular property prediction.

4. Tasks –

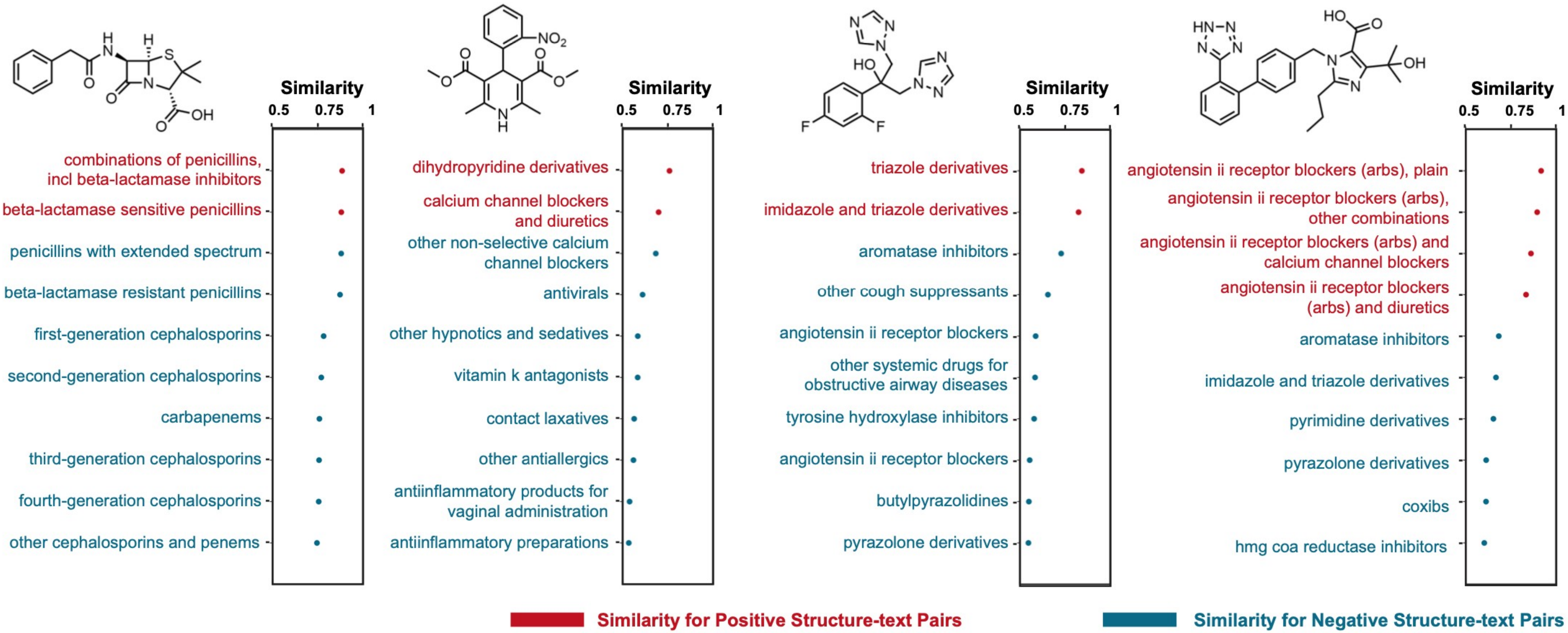
- 2 Objectives for tasks –
  1. *Open vocabulary,*
  2. *Compositionality.*

- **Zero-shot structure text retrieval.**



4. Tasks –

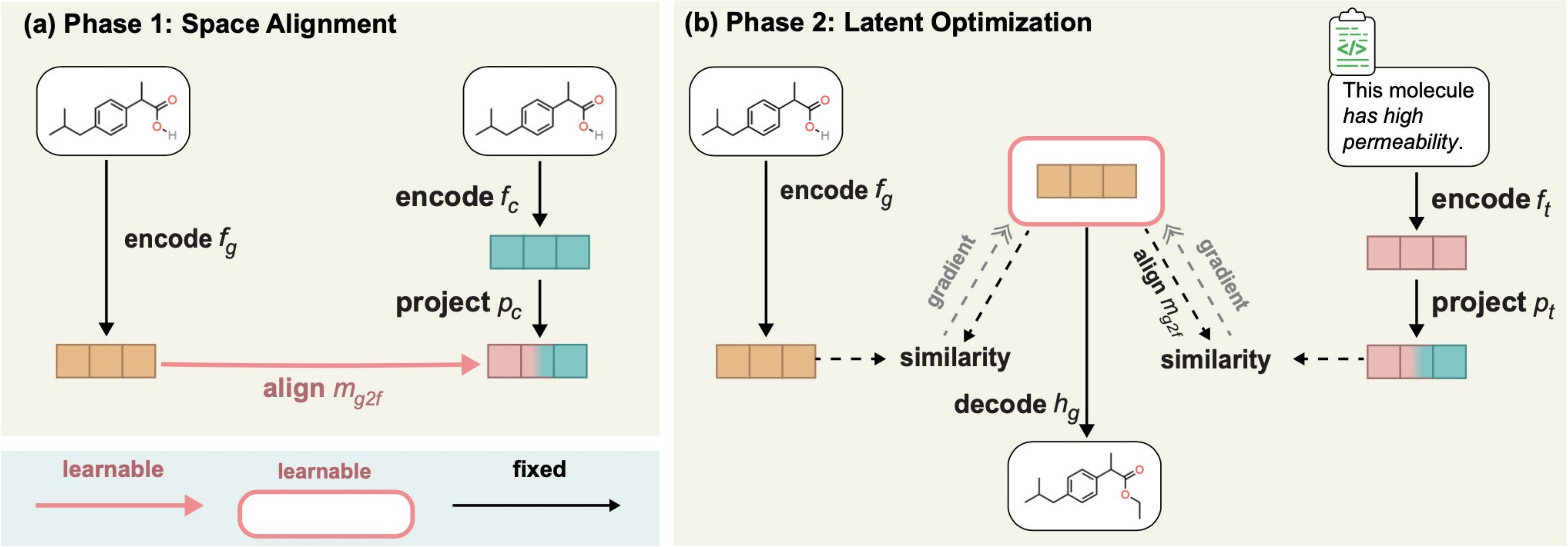
- **Zero-shot structure text retrieval. (Case Study)**



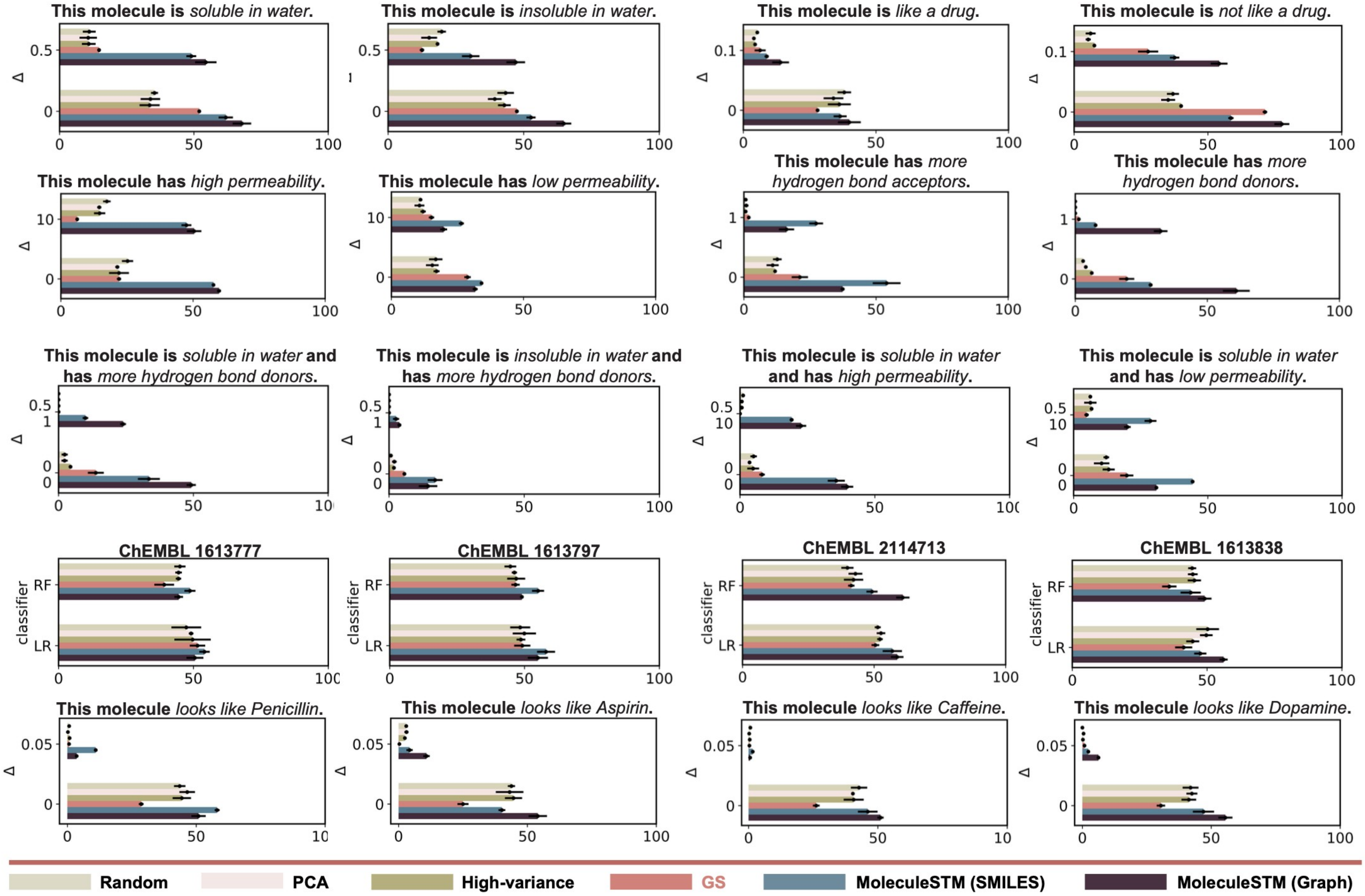
4. Tasks –

- **Zero-shot Text based Molecule editing.**

$$w = \underset{w \in \mathcal{W}}{\arg\min} \left(\mathcal{L}_{\text{cosine-sim}}\left(m_{g2f}(w), p_t \circ f_t(\boldsymbol{x}_t)\right) + \lambda \cdot \mathcal{L}_{l_2}\left(w, f_g(\boldsymbol{x}_{c,\text{in}})\right) \right)$$



Zero-shot Text based Molecule editing.

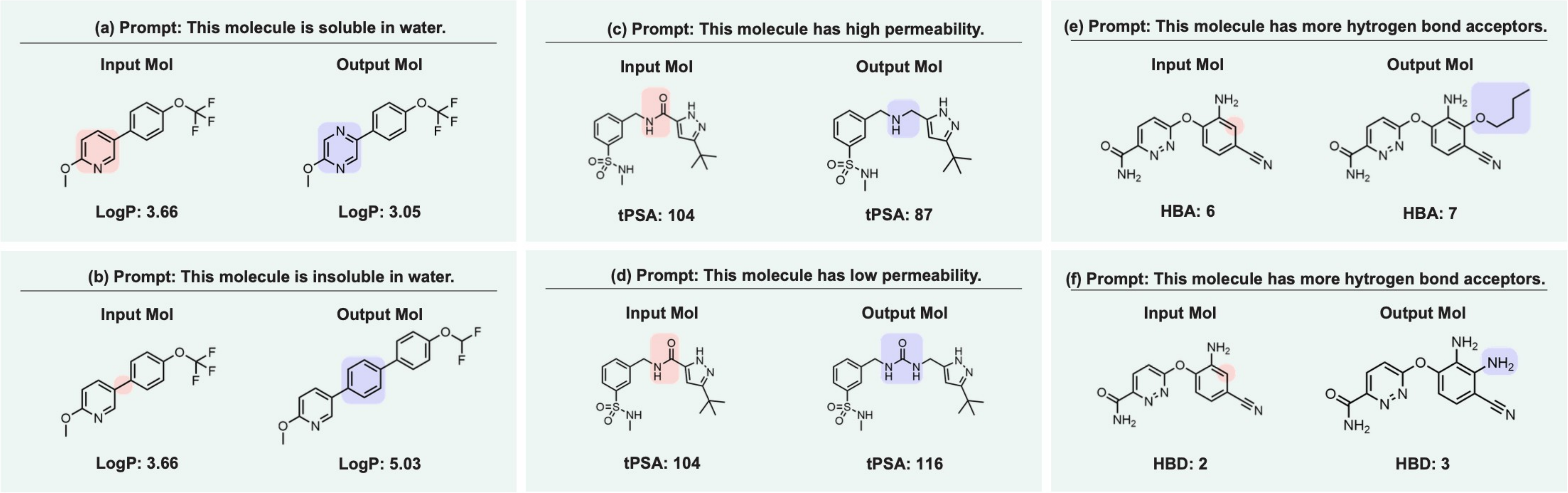


4. Tasks –

- **Zero-shot Text based Molecule editing. (Case Study)**

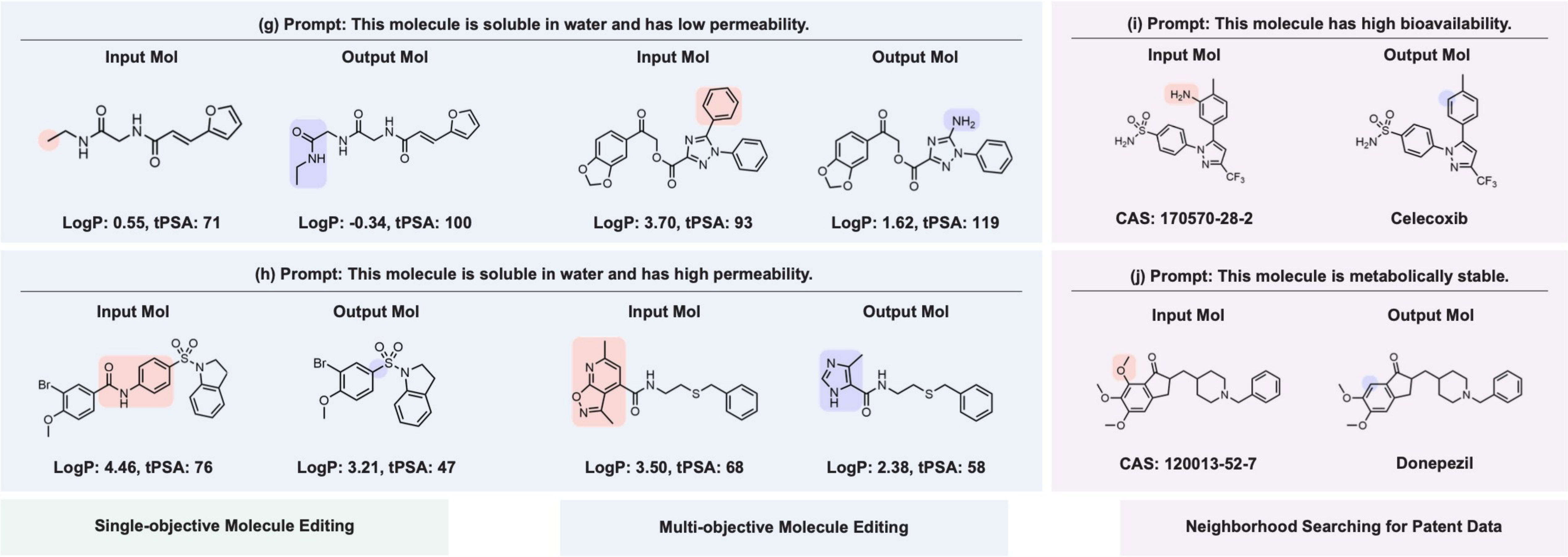
$$\text{hit}(\boldsymbol{x}_{c,\text{in}}, \boldsymbol{x}_t) = \begin{cases} 1, & \exists\lambda\text{, s.t. } \boldsymbol{x}_{c,\text{out}} = h_g\,(\text{w}; \lambda) \wedge \text{satisfy}\,(\boldsymbol{x}_{c,\text{in}}, \boldsymbol{x}_{c,\text{out}}, \boldsymbol{x}_t) \\ 0, & \text{otherwise} \end{cases}$$

$$\text{hit}(t) = \frac{\sum_{i=1}^{N} \text{hit}(\boldsymbol{x}^i_{c,\text{in}}, \boldsymbol{x}_t)}{N}$$



4. Tasks –

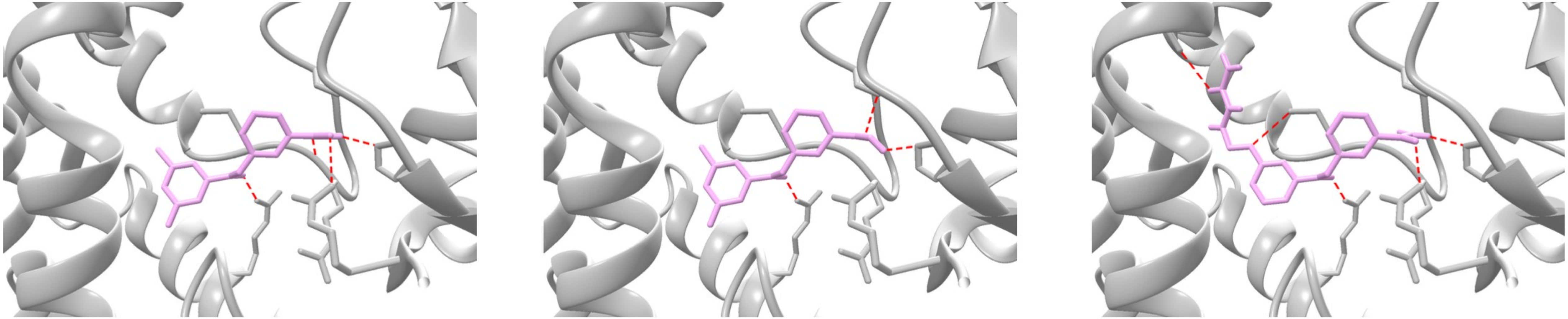
- **Zero-shot Text based Molecule editing. (Case Study)**



4. <u>*Tasks*</u> –

- **Zero-shot Text based Molecule editing. (Case Study)**

(k) Docking Visualization on CHEMBL1613777 (UniProt P33261)



Input Molecule
(docking score: -9.055)

Output Molecule with GS
(docking score: -8.843)

Output Molecule with MoleculeSTM
(docking score: -10.35)

4. Tasks –

- **Molecule property prediction.**

	method	BBBP ↑	Tox21 ↑	ToxCast ↑	Sider ↑	ClinTox ↑	MUV ↑	HIV ↑	Bace ↑	Avg ↑
SMILES	–	66.54±0.95	71.18±0.67	61.16±1.15	58.31±0.78	88.11±0.70	62.74±1.57	70.32±1.51	80.02±1.66	69.80
	MegaMolBART	68.89±0.17	73.89±0.67	63.32±0.79	59.52±1.79	78.12±4.62	61.51±2.75	71.04±1.70	**82.46±0.84**	69.84
	KV-PLM	70.50±0.54	72.12±1.02	55.03±1.65	59.83±0.56	**89.17±2.73**	54.63±4.81	65.40±1.69	78.50±2.73	68.15
	MoleculeSTM	**70.75±1.90**	**75.71±0.89**	**65.17±0.37**	**63.70±0.81**	86.60±2.28	**65.69±1.46**	**77.02±0.44**	81.99±0.41	**73.33**
Graph	–	63.90±2.25	75.06±0.24	64.64±0.76	56.63±2.26	79.86±7.23	70.43±1.83	76.23±0.80	73.14±5.28	69.99
	AttrMask	67.79±2.60	75.00±0.20	63.57±0.81	58.05±1.17	75.44±8.75	73.76±1.22	75.44±0.45	80.28±0.04	71.17
	ContextPred	63.13±3.48	74.29±0.23	61.58±0.50	60.26±0.77	80.34±3.79	71.36±1.44	70.67±3.56	78.75±0.35	70.05
	InfoGraph	64.84±0.55	76.24±0.37	62.68±0.65	59.15±0.63	76.51±7.83	72.97±3.61	70.20±2.41	77.64±2.04	70.03
	MolCLR	67.79±0.52	75.55±0.43	64.58±0.07	58.66±0.12	84.22±1.47	72.76±0.73	75.88±0.24	71.14±1.21	71.32
	GraphMVP	68.11±1.36	**77.06±0.35**	**65.11±0.27**	60.64±0.13	84.46±3.10	**74.38±2.00**	**77.74±2.51**	80.48±2.68	73.50
	MoleculeSTM	**69.98±0.52**	76.91±0.51	65.05±0.39	**60.96±1.05**	**92.53±1.07**	73.40±2.90	76.93±1.84	**80.77±1.34**	**74.57**

AdaProp: Learning Adaptive Propagation for Graph Neural Network based Knowledge Graph Reasoning

Yongqi Zhang*
zhangyongqi@4paradigm.com
4Paradigm Inc.
Beijing, China

Zhanke Zhou*
cszkzhou@comp.hkbu.edu.hk
Department of Computer Science,
Hong Kong Baptist University
Hong Kong, China

Quanming Yao†
qyaoaa@tsinghua.edu.cn
Department of Electronic
Engineering, Tsinghua University
Beijing, China

Xiaowen Chu
xwchu@ust.hk
Data Science and Analytics Thrust,
HKUST (Guangzhou)
Guangzhou, China

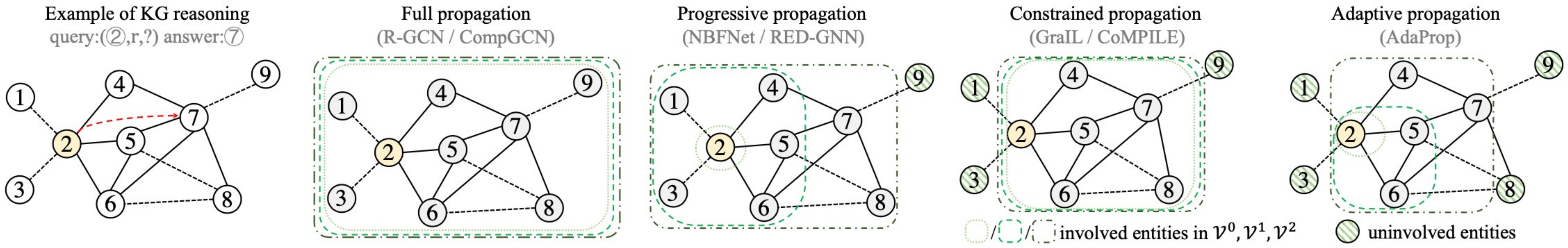
Bo Han
bhanml@comp.hkbu.edu.hk
Department of Computer Science,
Hong Kong Baptist University
Hong Kong, China

1. <u>Introduction</u> –

- *"Adaptively sample semantically relevant entities during propagation."*
- Reduced computation cost!
- Propose an *incremental sampling* scheme.
- Past work – GNNs, triplet learners, path learners
- GNN based method.
- SOTA performance.

2. <u>Related work</u> –

- GNN for KG reasoning (*Full propagation*, *Progressive propagation*, *Constrained propagation*)
- Sampling in GNNs (*node-wise*, *layer-wise*, *subgraph-sampling*)


Example of KG reasoning
query:(②,r,?) answer:⑦
1
2
3
4
5
6
7
8
9
Full propagation
(R-GCN / CompGCN)
Progressive propagation
(NBFNet / RED-GNN)
Constrained propagation
(GraIL / CoMPILE)
Adaptive propagation
(AdaProp)
/ / involved entities in $\mathcal{V}^0, \mathcal{V}^1, \mathcal{V}^2$
uninvolved entities

3. Method –

- *GNN for KG reasoning – F(**w**, $^{hat}G^{L}$)*
- Progressive methods ✅ but very heavy!
- So

$$\widehat{\mathcal{G}}^{L}_{e_q,r_q} = \{\mathcal{V}^{0}_{e_q,r_q}, \mathcal{V}^{1}_{e_q,r_q}, \dots, \mathcal{V}^{L}_{e_q,r_q}\},$$

$$\text{s.t. } \mathcal{V}^{\ell}_{e_q,r_q} = \begin{cases} \{e_q\} & \ell = 0 \\ S(\mathcal{V}^{\ell-1}_{e_q,r_q}) & \ell = 1 \dots L \end{cases}.$$

- Problems? Heuristics not good enough + might miss out on e_a.
- *Current sampling algorithms* suffer from these problems.
- *Proposed algorithm, **incremental sampling*** –

1. Reduce number of entities + preserve connections.
2. Has a *relation-dependent* sampling.

3. ***Method*** –

- *Majority of answers lie close to query entity*, impose $\mathcal{V}^0_{e_q,r_q} \subseteq \mathcal{V}^1_{e_q,r_q} \cdots \subseteq \mathcal{V}^L_{e_q,r_q}$

Table 6: Distance distribution (in %) of queries in $\mathcal{Q}_{tst}$.

distance	1	2	3	4	5	>5
WN18RR	34.9	9.3	21.5	7.5	8.9	17.9
FB15k237	0.0	73.4	25.8	0.2	0.1	0.5
NELL-995	40.9	17.2	36.5	2.5	1.3	1.6
YAGO3-10	56.0	12.9	30.1	0.5	0.1	0.4

- Sample as S(.) = SAMP(CAND(.))

$$\overline{\mathcal{V}}^{\ell}_{e_q,r_q} := \texttt{CAND}(\mathcal{V}^{\ell-1}_{e_q,r_q}) = \mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q}) \setminus \mathcal{V}^{\ell-1}_{e_q,r_q}$$

$$\mathcal{V}^{\ell}_{e_q,r_q} := \mathcal{V}^{\ell-1}_{e_q,r_q} \cup \texttt{SAMP}(\overline{\mathcal{V}}^{\ell}_{e_q,r_q})$$

3. Method –

- *Now for SAMP(.)* –

$$p^{\ell}(e) := \exp\left(g(\boldsymbol{h}_e^{\ell};\boldsymbol{\theta}^{\ell})/\tau\right) \Big/ \sum_{e' \in \overline{\mathcal{V}}_{e_q,r_q}^{\ell}} \exp\left(g(\boldsymbol{h}_{e'}^{\ell};\boldsymbol{\theta}^{\ell})/\tau\right)$$

$$G_e = g(\boldsymbol{h}_e^{\ell};\boldsymbol{\theta}^{\ell}) - \log(-\log U_e)$$

- Then optimize jointly –

$$\boldsymbol{w}^*, \boldsymbol{\theta}^* = \arg\min_{\boldsymbol{w},\boldsymbol{\theta}} \sum_{(e_q,r_q,e_a)\in\mathcal{Q}_{\text{tra}}} \mathcal{L}\left(F(\boldsymbol{w}, \widehat{\mathcal{G}}_{e_q,r_q}^{L}(\boldsymbol{\theta})), e_a\right)$$

$$\mathcal{L}\left(F(\boldsymbol{w}, \widehat{\mathcal{G}}^{L}(\boldsymbol{\theta})), e_a\right) = -\sum_{e_o \in \mathcal{V}_{e_q,r_a}^{L}} y_{e_o}\log(\phi_{e_o}) + (1-y_{e_o})\log(1-\phi_{e_o})$$

3. <u>Method</u> –

- *Advantages* –
    1. Entity-efficient, linear w.r.t. layers. (Proposition 1)
    2. **Layer-wise connections can be preserved!** (Proposition 2)
    3. Proposed sampling strategy has more chance of preserving *'good'* entities compared to others for same number of nodes.

PROPOSITION 1. *The number of involved entities in the propagation path* ($\left|\bigcup_{\ell=0\ldots L} \mathcal{V}^{\ell}_{e_q,r_q}\right|$) *of incremental sampling is bounded by* $O(LK)$.

PROPOSITION 2. *For all the entities* $e \in \mathcal{V}^{\ell-1}_{e_q,r_q}$ *with incremental sampling, there exists at least one entity* $e' \in \mathcal{V}^{\ell}_{e_q,r_q}$ *and relation* $r \in \mathcal{R}$ *such that* $(e, r, e') \in \mathcal{E}$.

3. Method – Full Algorithm

Algorithm 1 AdaProp: learning adaptive propagation path.

Require: query $(e_q, r_q, ?)$, $\mathcal{V}^0_{e_q,r_q} = \{e_q\}$, steps L, number of sampled entities K, and functions $\text{MESS}(\cdot)$ and $\text{AGG}(\cdot)$.

for $\ell = 1 \dots L$ **do**

  get the neighboring entities $\mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q}) = \cup_{e \in \mathcal{V}^{\ell-1}_{e_q,r_q}} \mathcal{N}(e)$, the newly-visited entities $\overline{\mathcal{V}}^{\ell}_{e_q,r_q} = \mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q}) \setminus \mathcal{V}^{\ell-1}_{e_q,r_q}$, and edges $\mathcal{E}^{\ell} = \{(e_s, r, e_o) | e_s \in \mathcal{V}^{\ell-1}_{e_q,r_q}, e_o \in \mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q})\}$;

  obtain $\boldsymbol{m}^{\ell}_{(e_s,r,e_o)} := \text{MESS}(\boldsymbol{h}^{\ell-1}_{e_s}, \boldsymbol{h}^{\ell-1}_{e_o}, \boldsymbol{h}^{\ell}_{r}, \boldsymbol{h}^{\ell}_{r_q})$ for edges $(e_s, r, e_o) \in \mathcal{E}^{\ell}$;

  obtain $\boldsymbol{h}^{\ell}_{e_o} := \delta\big(\text{AGG}\big(\boldsymbol{m}^{\ell}_{(e_s,r,e_o)}, (e_s, r, e_o) \in \mathcal{E}^{\ell}\big)\big)$ for entities $e_o \in \mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q})$;

  logits computation: obtain the Gumbel logits $G_{e_o} = g(\boldsymbol{h}^{\ell}_{e_o}; \boldsymbol{\theta}^{\ell}) - \log(-\log U_{e_o})$ with $U_{e_o} \sim \text{Uniform}(0, 1)$ for entities $e_o \in \overline{\mathcal{V}}^{\ell}_{e_q,r_q}$;

  candidate sampling: obtain sampled entities $\widetilde{\mathcal{V}}^{\ell}_{e_q,r_q} = \{\arg\text{top}_K G_{e_o}, e_o \in \overline{\mathcal{V}}^{\ell}_{e_q,r_q}\}$;

  straight-through: $\boldsymbol{h}^{\ell}_{e} := \big(1 - \texttt{no\_grad}(\boldsymbol{p}^{\ell}(e)) + \boldsymbol{p}^{\ell}(e)\big) \cdot \boldsymbol{h}^{\ell}_{e}$ for entities $e \in \widetilde{\mathcal{V}}^{\ell}_{e_q,r_q}$;

  update propagation path: update $\mathcal{V}^{\ell}_{e_q,r_q} = \mathcal{V}^{\ell-1}_{e_q,r_q} \cup \widetilde{\mathcal{V}}^{\ell}_{e_q,r_q}$;

end for

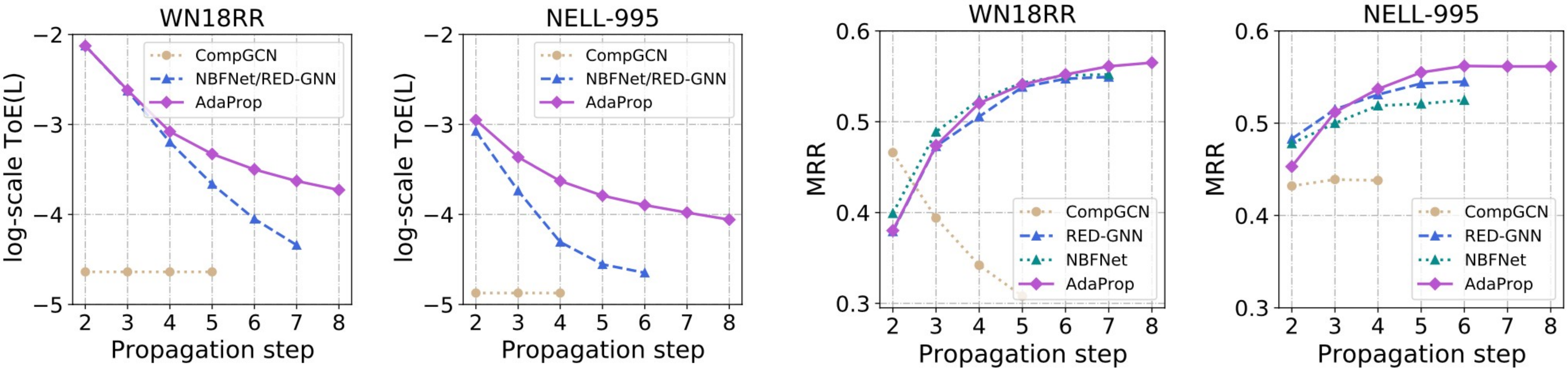
return $f\big(\boldsymbol{h}^{L}_{e_o}; \boldsymbol{w}^{\top}\big)$ for each $e_o \in \mathcal{V}^{L}_{e_q,r_q}$.

4. *Experiments* –

- Compare against strong inductive and transductive baselines.
- Also introduced a new metric to quantify effectiveness of the sampling algorithm.

$$\mathtt{ToE}(L) = \mathtt{TC}(L) / \mathtt{EI}(L)$$

- **Transductive setting.**



4. ***Experiments*** –

- **Transductive setting.**

type	models	Family			UMLS			WN18RR			FB15k237			NELL-995			YAGO3-10		
		MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
non-GNN	ConvE	0.912	83.7	98.2	0.937	92.2	96.7	0.427	39.2	49.8	0.325	23.7	50.1	0.511	44.6	61.9	0.520	45.0	66.0
	QuatE	0.941	89.6	99.1	0.944	90.5	99.3	0.480	44.0	55.1	0.350	25.6	53.8	0.533	46.6	64.3	0.379	30.1	53.4
	RotatE	0.921	86.6	98.8	0.925	86.3	99.3	0.477	42.8	57.1	0.337	24.1	53.3	0.508	44.8	60.8	0.495	40.2	67.0
	MINERVA	0.885	82.5	96.1	0.825	72.8	96.8	0.448	41.3	51.3	0.293	21.7	45.6	0.513	41.3	63.7	–	–	–
	DRUM	0.934	88.1	<u>99.6</u>	0.813	67.4	97.6	0.486	42.5	58.6	0.343	25.5	51.6	0.532	46.0	66.2	0.531	45.3	67.6
	RNNLogic	0.881	85.7	90.7	0.842	77.2	96.5	0.483	44.6	55.8	0.344	25.2	53.0	0.416	36.3	47.8	0.554	50.9	62.2
	RLogic	–	–	–	–	–	–	0.47	44.3	53.7	0.31	20.3	50.1	–	–	–	0.36	25.2	50.4
GNNs	CompGCN	0.933	88.3	99.1	0.927	86.7	<u>99.4</u>	0.479	44.3	54.6	0.355	26.4	53.5	0.463	38.3	59.6	0.421	39.2	57.7
	NBFNet	0.989	**98.8**	98.9	0.948	92.0	**99.5**	<u>0.551</u>	<u>49.7</u>	<u>66.6</u>	<u>0.415</u>	<u>32.1</u>	**59.9**	0.525	45.1	63.9	0.550	47.9	68.6
	RED-GNN	**0.992**	**98.8**	**99.7**	<u>0.964</u>	<u>94.6</u>	99.0	0.533	48.5	62.4	0.374	28.3	55.8	<u>0.543</u>	<u>47.6</u>	<u>65.1</u>	0.559	48.3	68.9
	AdaProp	<u>0.988</u>	98.6	99.0	**0.969**	**95.6**	**99.5**	**0.562**	**49.9**	**67.1**	**0.417**	**33.1**	<u>58.5</u>	**0.554**	**49.3**	**65.5**	**0.573**	**51.0**	**68.5**

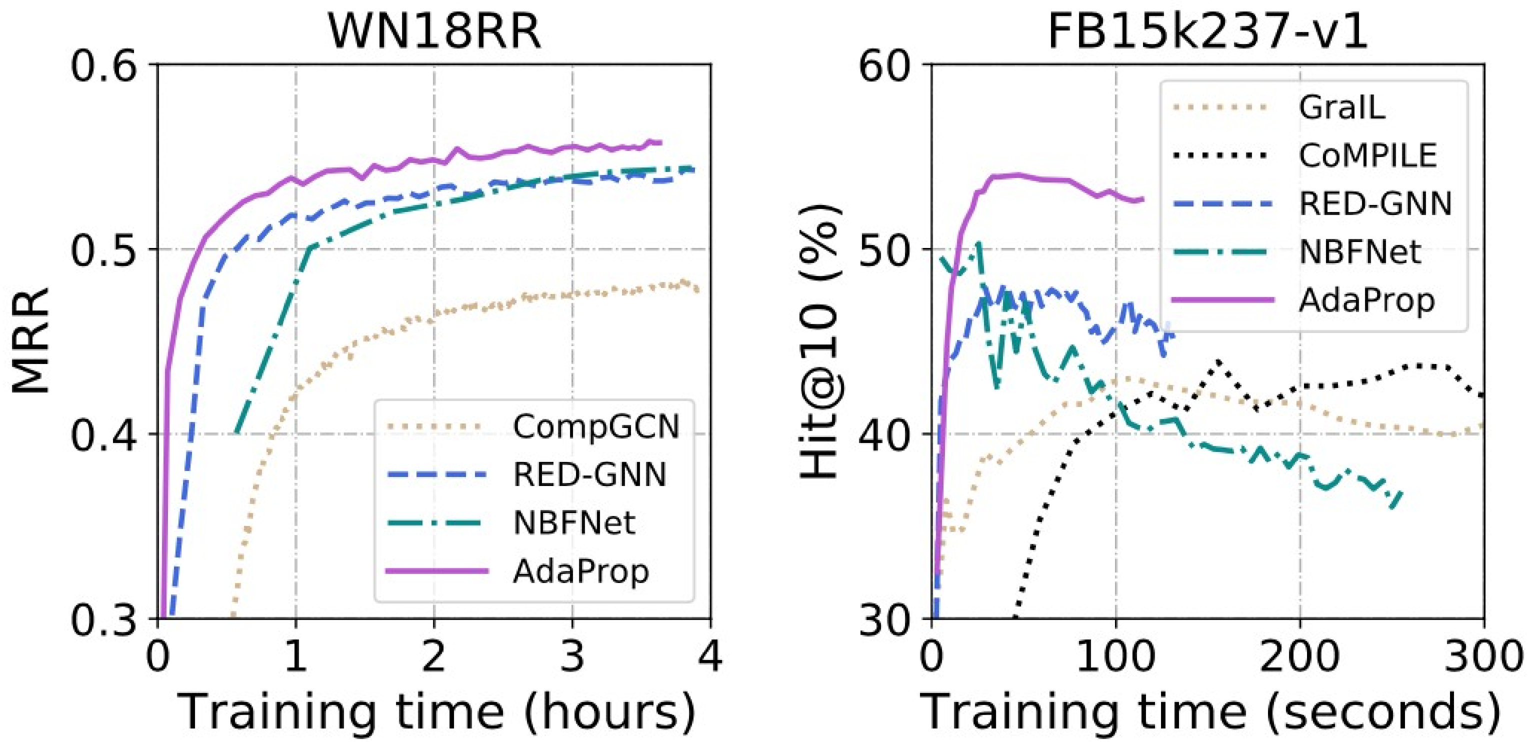
4. *Experiments* –

- **Inductive setting.**

metric	methods	WN18RR V1	WN18RR V2	WN18RR V3	WN18RR V4	FB15k237 V1	FB15k237 V2	FB15k237 V3	FB15k237 V4	NELL-995 V1	NELL-995 V2	NELL-995 V3	NELL-995 V4
Hit@10 (%)	RuleN	73.0	69.4	40.7	68.1	44.6	59.9	60.0	60.5	76.0	51.4	53.1	48.4
	Neural LP	77.2	74.9	47.6	70.6	46.8	58.6	57.1	59.3	87.1	56.4	57.6	53.9
	DRUM	77.7	74.7	47.7	70.2	47.4	59.5	57.1	59.3	87.3	54.0	57.7	53.1
	GraIL	76.0	77.6	40.9	68.7	42.9	42.4	42.4	38.9	56.5	49.6	51.8	50.6
	CoMPILE	74.7	74.3	40.6	67.0	43.9	45.7	44.9	35.8	57.5	44.6	51.5	42.1
	NBFNet	82.7	79.9	56.3	70.2	51.7	63.9	58.8	55.9	79.5	63.5	60.6	59.1
	RED-GNN	79.9	78.0	52.4	72.1	48.3	62.9	60.3	62.1	86.6	60.1	59.4	55.6
	AdaProp	**86.6**	**83.6**	**62.6**	**75.5**	**55.1**	**65.9**	**63.7**	**63.8**	**88.6**	**65.2**	**61.8**	**60.7**

4. Experiments –

- **Training time.**



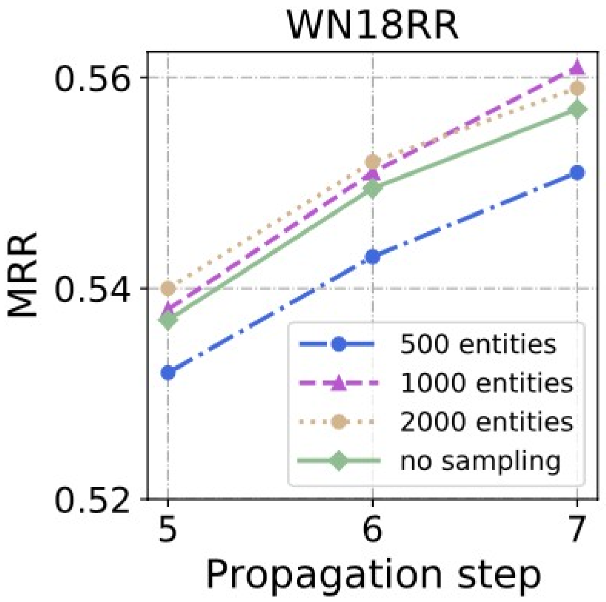
5. *Ablation* –

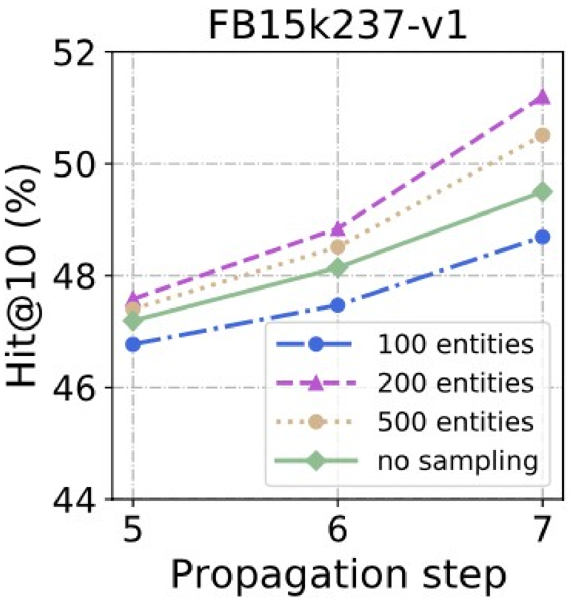
- **Importance of sampling strategies and learning.**

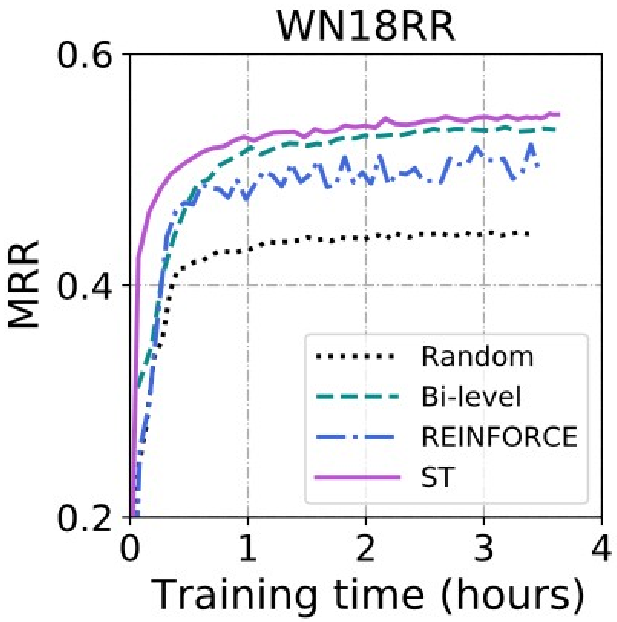
learn	methods	WN18RR			FB15k237-v1		
		$\texttt{EI}(L)$	$\texttt{ToE}(L)$	MRR	$\texttt{EI}(L)$	$\texttt{ToE}(L)$	Hit@10
not learned	Node-wise	4831	1.38E−4	.416	585	1.35E−3	38.9
	Layer-wise	5035	1.46E−4	.428	554	1.45E−3	37.2
	Subgraph	5098	1.57E−4	.461	578	1.50E−3	40.5
	Incremental	4954	1.61E−4	.472	559	1.52E−3	40.1
learned	Node-wise	4913	1.52E−4	.529	561	1.47E−3	50.4
	Layer-wise	4871	1.64E−4	.533	556	1.55E−3	52.4
	Incremental	4749	1.78E−4	.562	564	1.57E−3	55.1

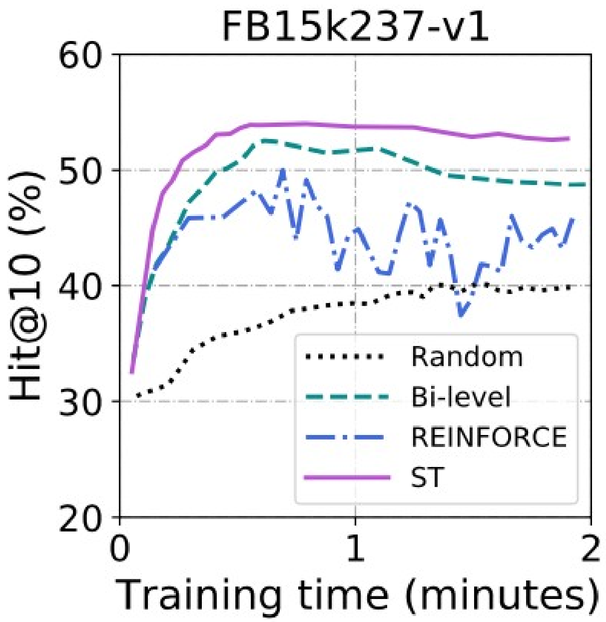
5. Ablation –

- **Influence of K & Comparison of learning strategies.**



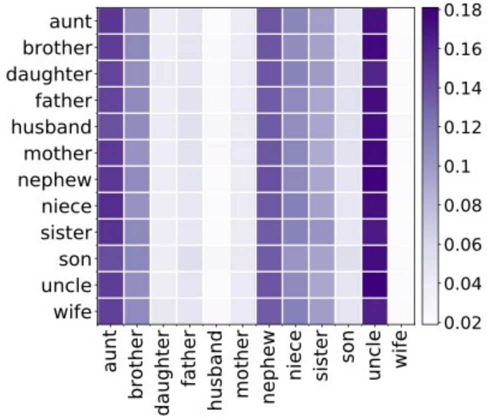




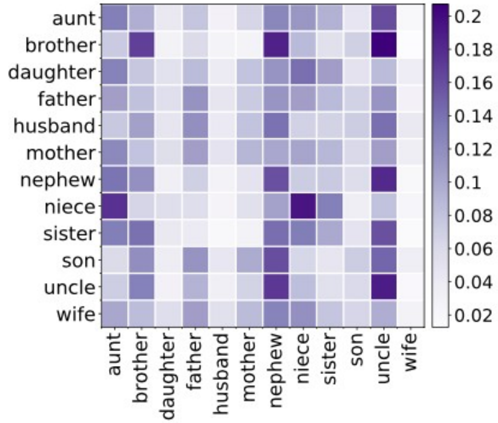


6. Case Study –

- **Left : demonstration that sampling is semantic aware.**
- **Right : example paths of AdaProp vs Progressive algorithms.**

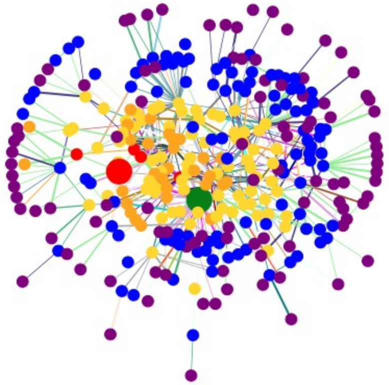


(a) Progressive on Family

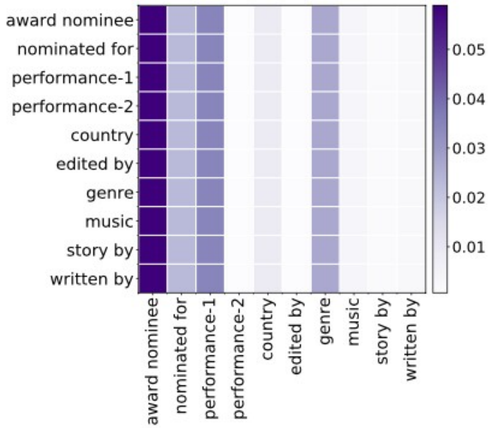


(b) AdaProp on Family

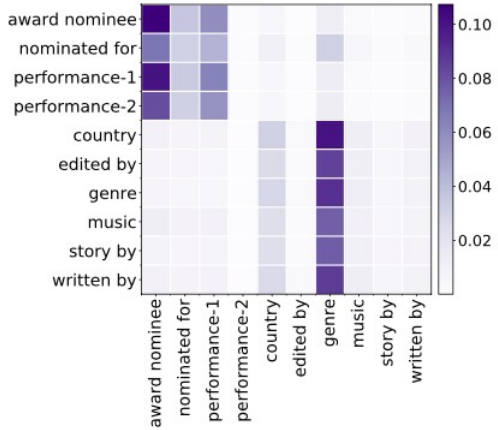
(a) AdaProp, q_1



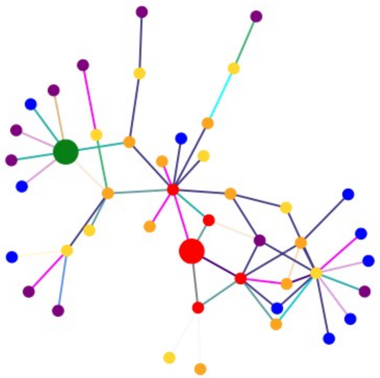
(b) Progressive, q_1



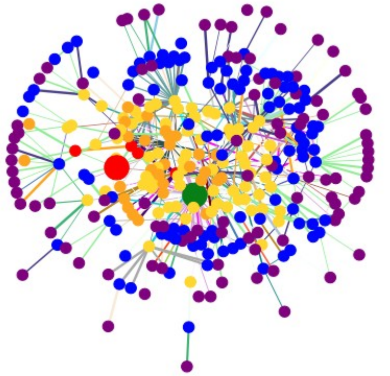
(c) Progressive on FB15k237



(d) AdaProp on FB15k237



(c) AdaProp, q_2



(d) Progressive, q_2