Combating Bilateral Edge Noise for Robust Link Prediction

Zhanke Zhou

Hong Kong Baptist University

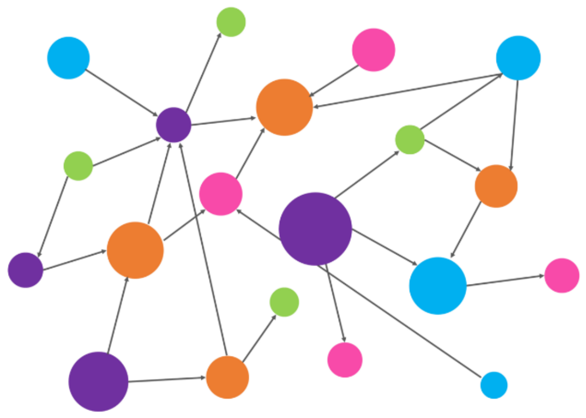
with Jiangchao Yao, Jiaxu Liu, Xiawei Guo, Quanming Yao, Li He, Liang Wang, Bo Zheng, Bo Han

[NeurIPS'23 Paper]: https://openreview.net/pdf?id=ePkLqJh5kw [Code]: https://github.com/tmlr-group/RGIB

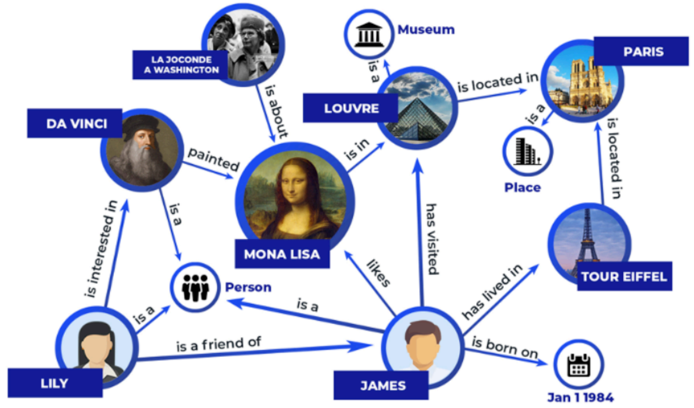
Outline

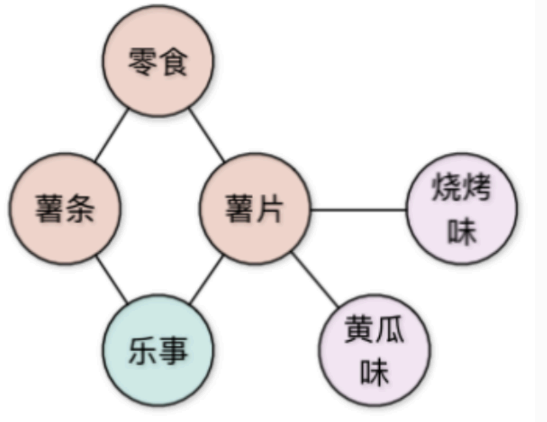
- Introduction
- Method
- Experiments
- Summary

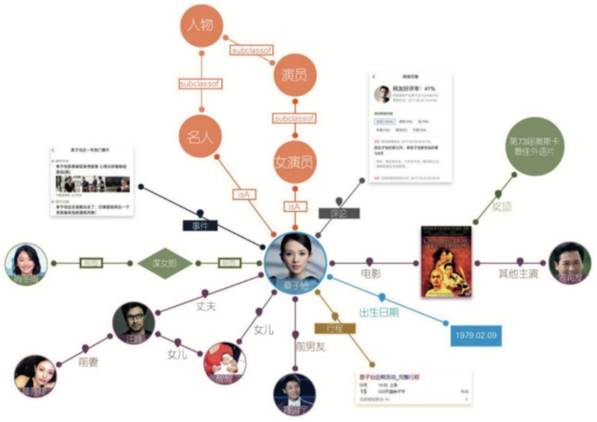
Introduction | background

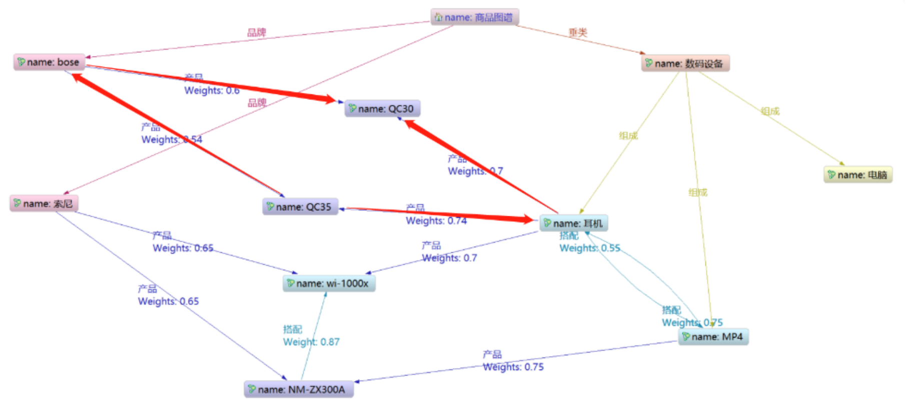


Graph: a general form of data expression







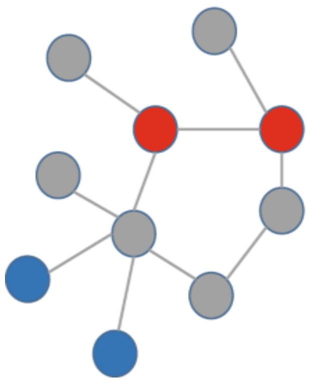


Introduction | background

The link prediction task

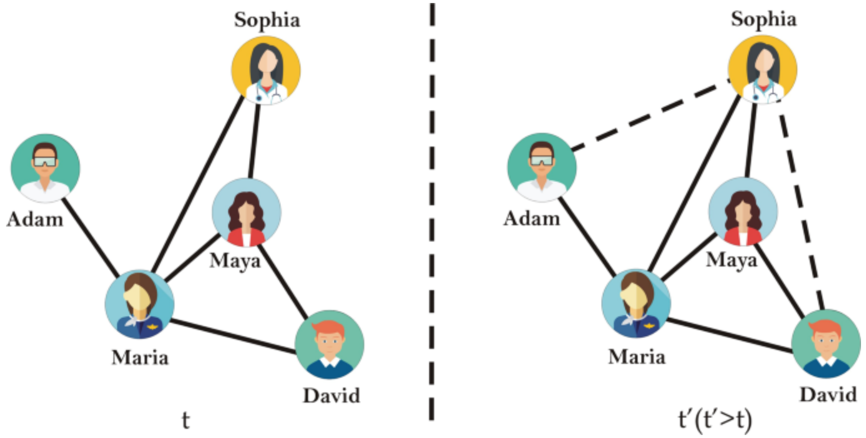
- based on the observed links
- to predict the latent links between the nodes

node-level



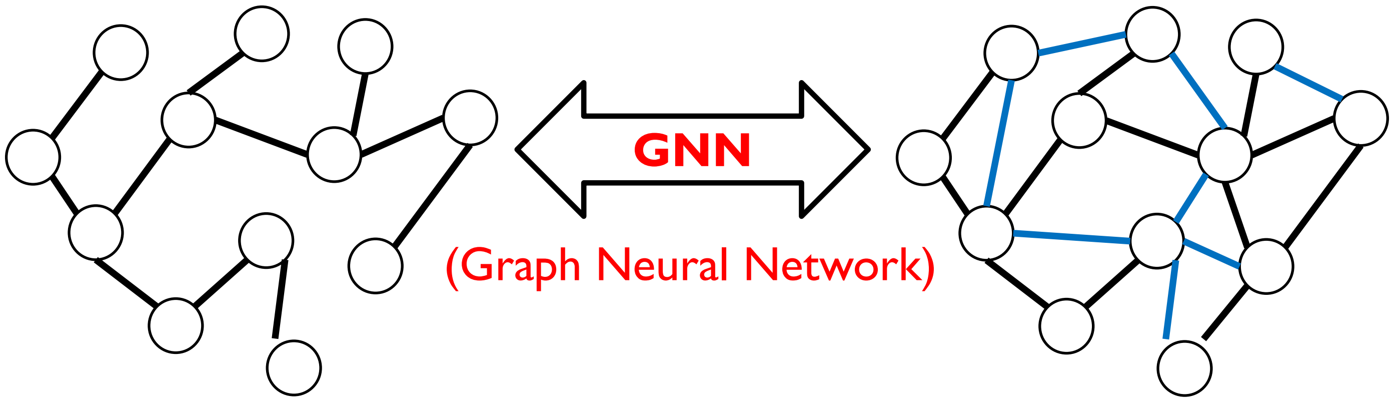
link-level

（most relevant to the recommendation system）

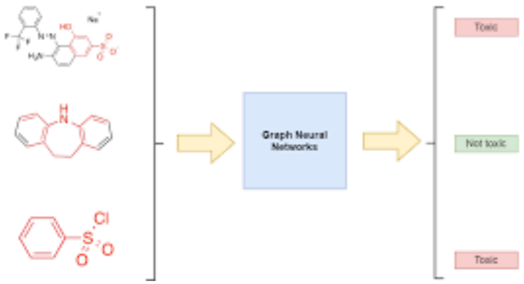


Observed graph

Predictive graph



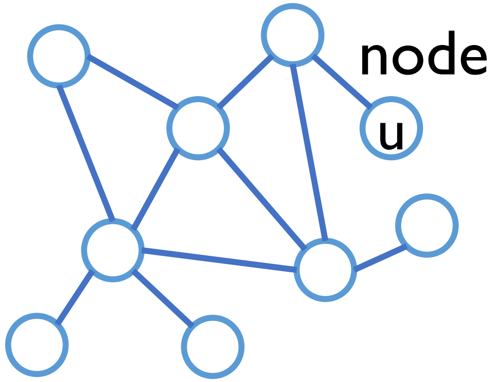
graph-level



Introduction | graph representation learning

- GNN for link prediction on graphs

encode: $\mathrm{n}_u \rightarrow \boldsymbol{h}_u : \mathbb{R}^d$



$$\boldsymbol{m}^{\ell}_{(u,v)} = \mathrm{MESS}(\boldsymbol{h}^{\ell-1}_u, \boldsymbol{h}^{\ell-1}_v, \boldsymbol{e}_{uv})$$

$$\boldsymbol{h}^{\ell}_v = \delta\Big(\mathrm{AGG}\big(\boldsymbol{m}^{\ell}_{(u,v)}, u \in \mathcal{N}(v)\big)\Big)$$

$\boldsymbol{h}_u$: node representation

$\mathbb{R}^d$

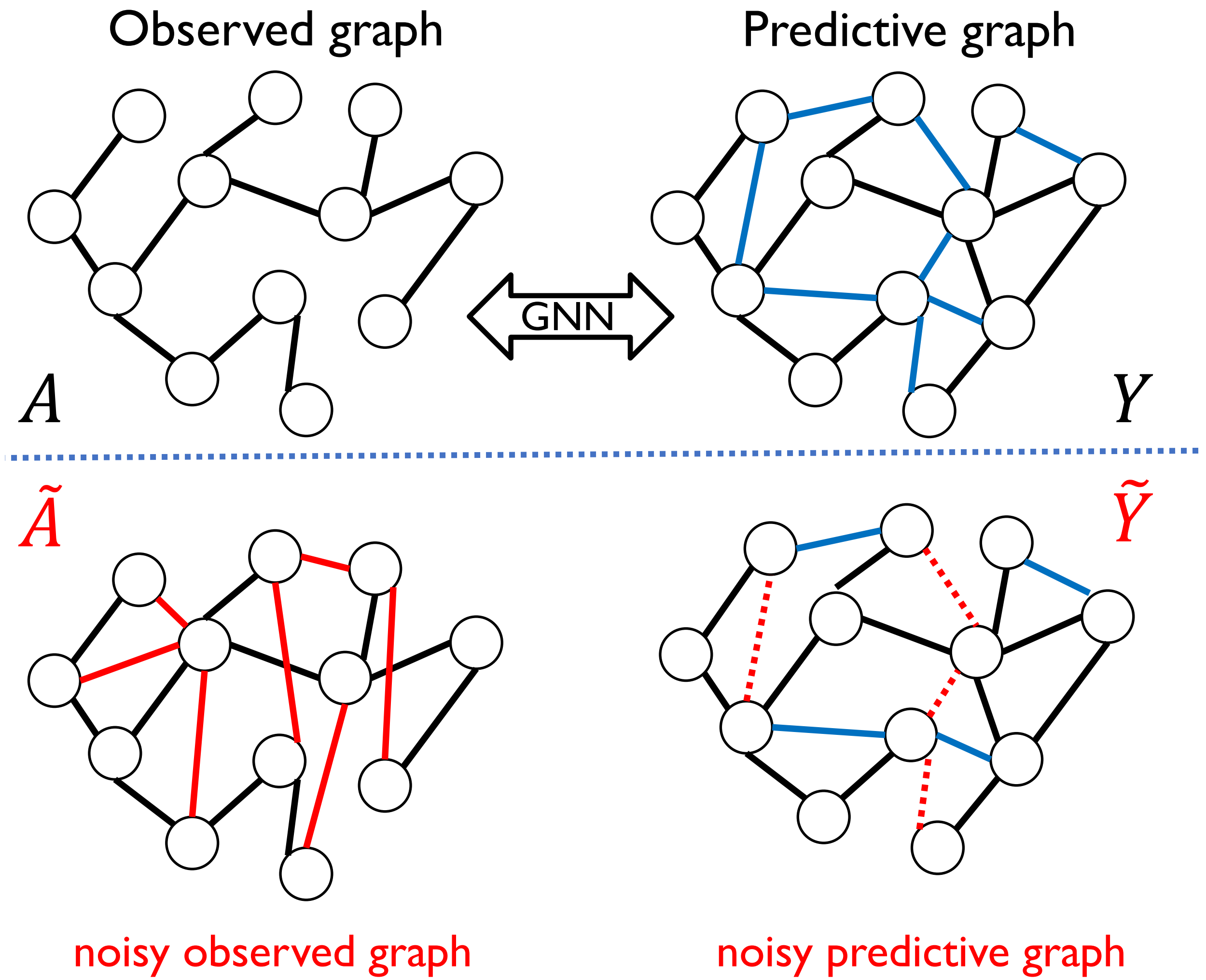
decode: $\boldsymbol{\phi}_{uv} = \mathrm{READOUT}(\boldsymbol{h}_u, \boldsymbol{h}_v) \rightarrow \mathbb{R}$

optimization: $\mathcal{L} = \sum_{e_{uv} \in \mathcal{E}^{train}} -y_{ij} \log(\boldsymbol{\phi}_{uv}) + (1 - y_{ij}) \log(1 - \boldsymbol{\phi}_{uv})$

Introduction | problem setup

ideal case
(clean data)

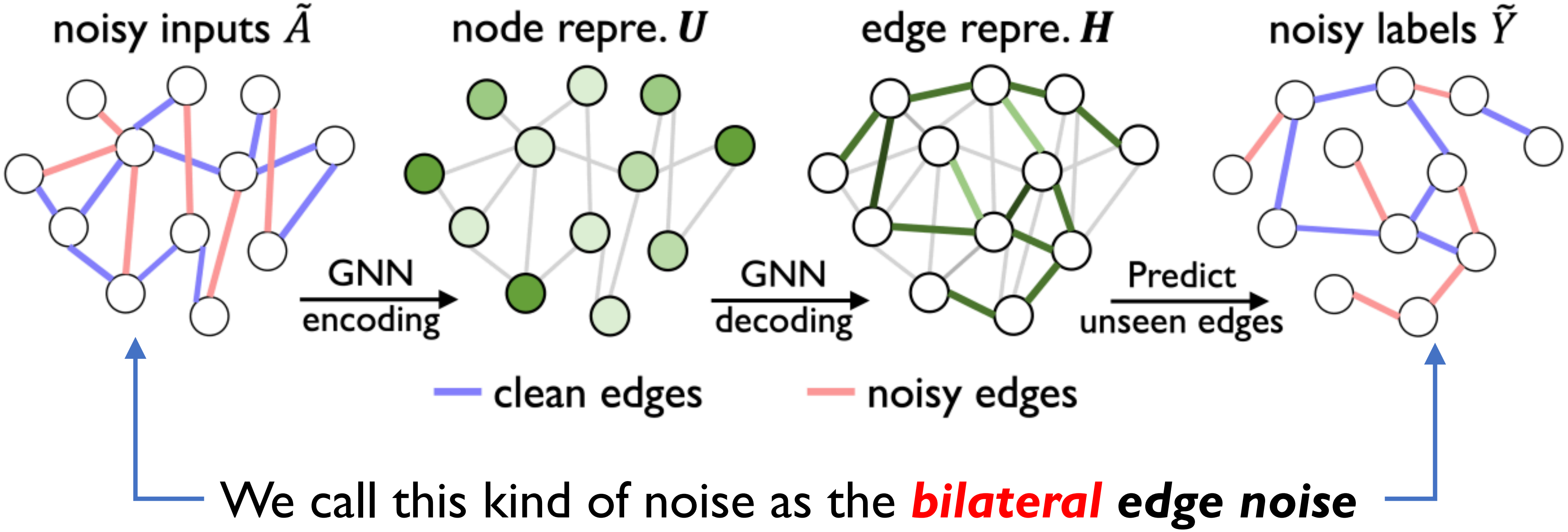
practical case
(with bilateral noise)



Introduction | problem setup

In practical scenarios,

- the observed graph is often with noisy edges (input noise)
- the predictive graph often contains noisy labels (label noise)
- these two kinds of noise can exist at the same time (by random split)



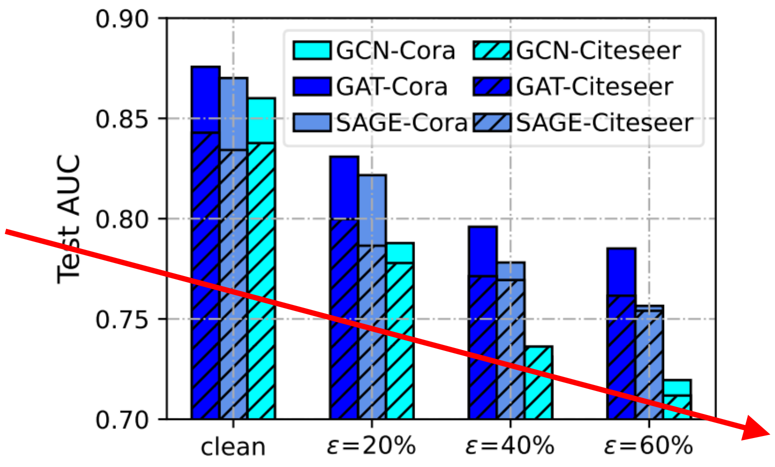
We call this kind of noise as the ***bilateral* edge noise**

Research problem: how to improve the robustness of GNNs under edge noise 🤔

Introduction | problem setup

Definition 3.1 (Bilateral edge noise). *Given a clean training data, i.e., observed graph $\mathcal{G} = (A, X)$ and labels $Y \in \{0, 1\}$ of query edges, the noisy adjacence $\tilde{A}$ is generated by directly adding edge noise to the original adjacent matrix A while keeping the node features X unchanged. The noisy labels $\tilde{Y}$ are similarly generated by adding edge noise to the labels Y. Specifically, given a noise ratio ε_a, the noisy edges A' ($\tilde{A} = A + A'$) are generated by flipping the zero element in A as one with the probability ε_a. It satisfies that $A' \odot A = O$ and $\varepsilon_a = {}^{|\texttt{nonzero}(\tilde{A})| - |\texttt{nonzero}(A)|}/_{|\texttt{nonzero}(A)|}$. Similarly, noisy labels are generated and added to the original labels, where $\varepsilon_y = {}^{|\texttt{nonzero}(\tilde{Y})| - |\texttt{nonzero}(Y)|}/_{|\texttt{nonzero}(Y)|}$.*

Link prediction performance in AUC with the bilateral edge noise



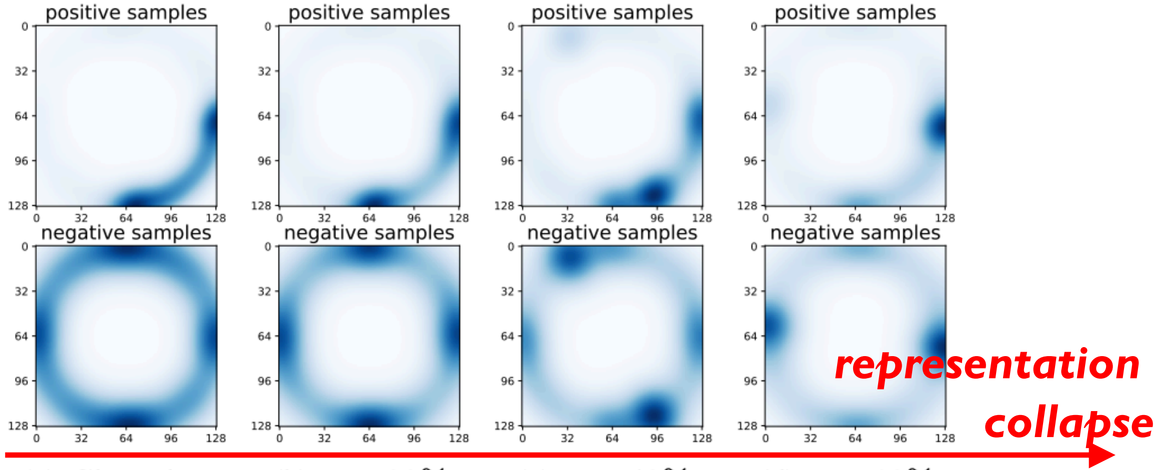
performance drop

Inspecting the representation distribution:

Table 1: Mean values of alignment, which are calculated as the L2 distance of representations of two randomly perturbed graphs $\tilde{A}_1^i, \tilde{A}_2^i$, *i.e.*, $\texttt{Align} = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{H}_1^i - \boldsymbol{H}_2^i||_2$. Representation $\boldsymbol{H}_1^i = f_{\boldsymbol{w}}(\tilde{A}_1^i, X)$ and $\boldsymbol{H}_2^i = f_{\boldsymbol{w}}(\tilde{A}_2^i, X)$.

dataset	Cora	Citeseer
clean	.616	.445
ε=20%	.687	.586
ε=40%	.695	.689
ε=60%	.732	.696

representation collapse



(a) Clean data (b) $\varepsilon = 20\%$ (c) $\varepsilon = 40\%$ (d) $\varepsilon = 60\%$

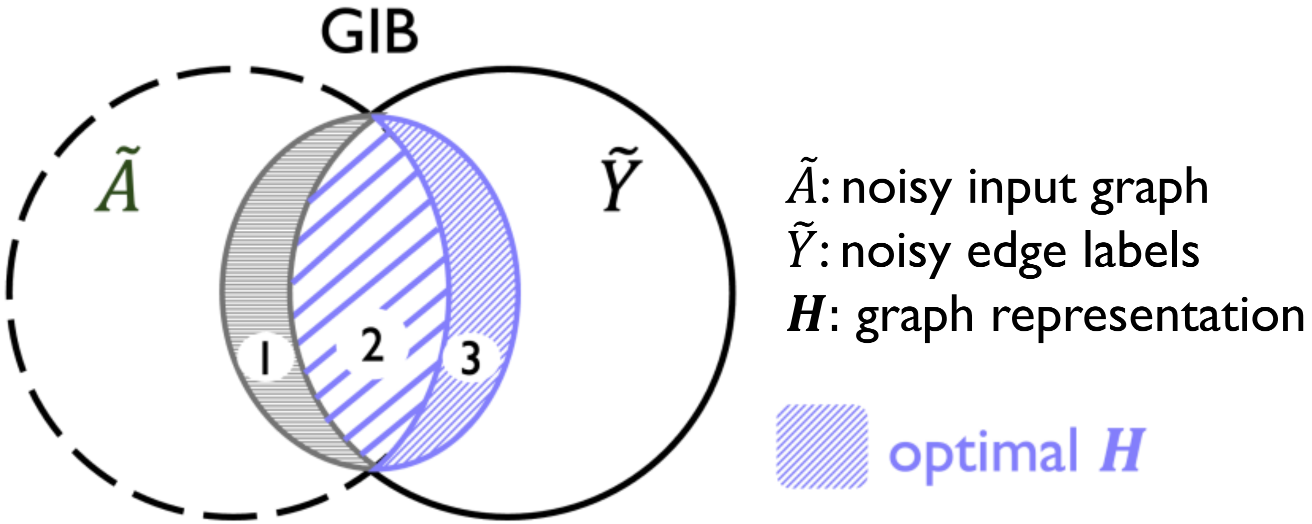
Figure 4: Uniformity distribution on Cora dataset. Representations of query edges in the test set are mapped to unit circle of $\mathbb{R}^2$ with normalization followed by the Gaussian kernel density estimation as [35]. Both positive and negative edges are expected to be uniformly distributed.

Research problem: how to improve the robustness of GNNs under edge noise 🤔

Outline

- Introduction
- Method
- Experiments
- Summary

Graph Information Bottleneck (GIB)



defend the input perturbation

$$\boxed{\min \mathrm{GIB} \triangleq -I(\boldsymbol{H}; \tilde{Y})}, \quad \boxed{\text{s.t. } I(\boldsymbol{H}; \tilde{A}) < \gamma,}$$

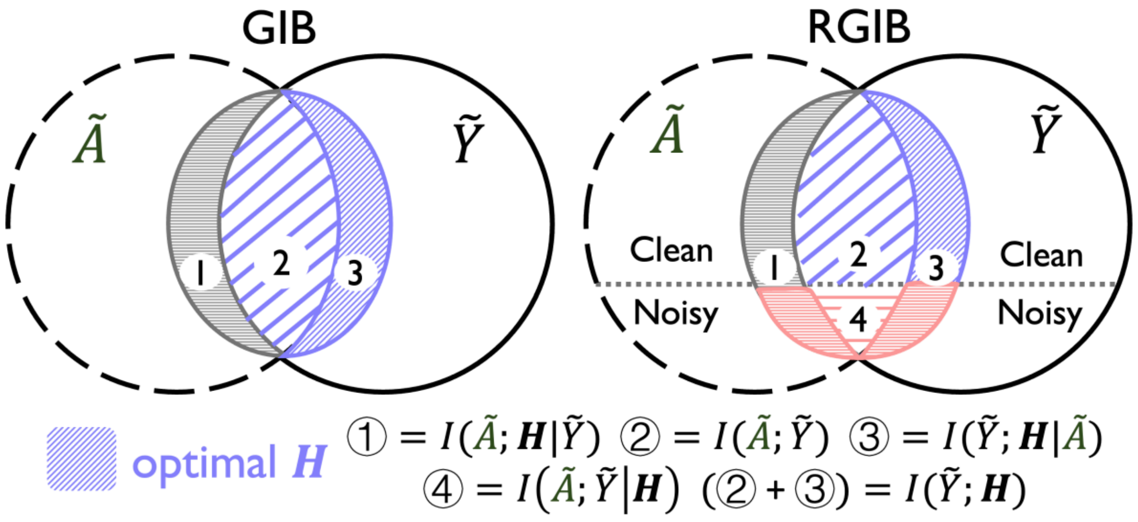
However, GIB is intrinsically vulnerable to label noise

since it entirely preserves the label supervision

Robust Graph Information Bottleneck (**R**GIB)

$$\min \mathrm{GIB} \triangleq -I(\boldsymbol{H};\tilde{Y}),\ \text{s.t. } I(\boldsymbol{H};\tilde{A}) < \gamma,$$

$\tilde{A}$: noisy input graph
$\tilde{Y}$: noisy edge labels
$\boldsymbol{H}$: graph representation

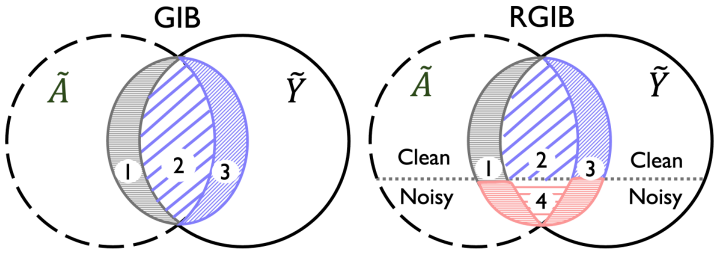


Definition 4.1 (Robust Graph Information Bottleneck)**.** *Based on the above analysis, we propose a new learning objective to balance informative signals regarding* $\boldsymbol{H}$*, as illustrated in Fig. 5(a), i.e.,*

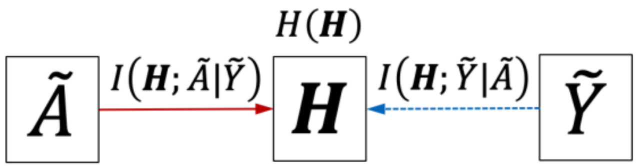
$$\min \mathit{RGIB} \triangleq -I(\boldsymbol{H};\tilde{Y}),\quad s.t.\ \gamma_H^- < H(\boldsymbol{H}) < \gamma_H^+, I(\boldsymbol{H};\tilde{Y}|\tilde{A}) < \gamma_Y,\ I(\boldsymbol{H};\tilde{A}|\tilde{Y}) < \gamma_A. \quad (2)$$

Specifically, constraints on $H(\boldsymbol{H})$ *encourage a diverse* $\boldsymbol{H}$ *to prevent representation collapse* ($>\gamma_H^-$) *and also limit its capacity* ($<\gamma_H^+$) *to avoid over-fitting. Another two MI terms,* $I(\boldsymbol{H};\tilde{Y}|\tilde{A})$ *and* $I(\boldsymbol{H};\tilde{A}|\tilde{Y})$*, mutually regularize posteriors to mitigate the negative impact of bilateral noise on* $\boldsymbol{H}$*. The complete derivation of RGIB and a further comparison of RGIB and GIB are in Appendix B.2.*

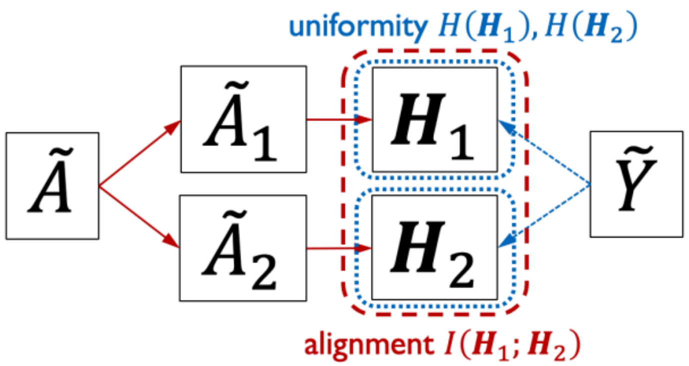
Robust Graph Information Bottleneck



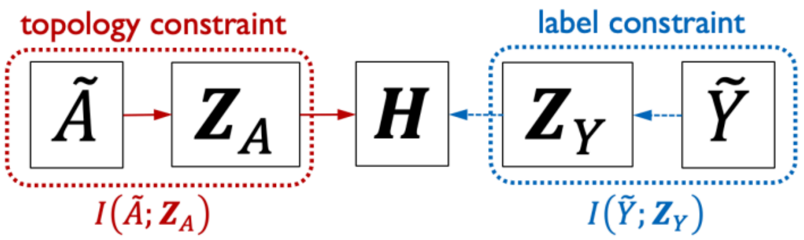
$$\min RGIB \triangleq -I(\boldsymbol{H};\tilde{Y}), \ \ s.t.\ \gamma_H^- < H(\boldsymbol{H}) < \gamma_H^+, I(\boldsymbol{H};\tilde{Y}|\tilde{A}) < \gamma_Y, \ I(\boldsymbol{H};\tilde{A}|\tilde{Y}) < \gamma_A.$$



RGIB



RGIB-SSL

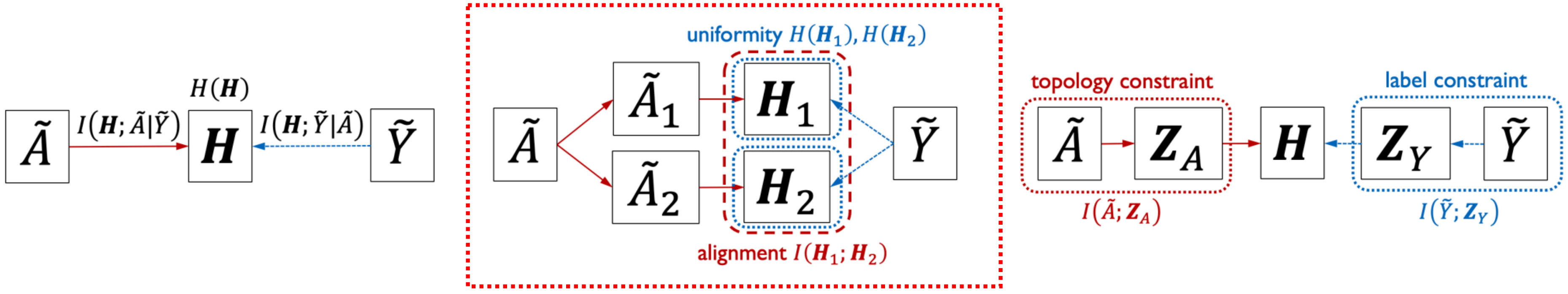


RGIB-REP

Two practical implementations of RGIB:

- RGIB-SSL explicitly optimizes the representation $\boldsymbol{H}$ with the self-supervised regularization
- RGIB-REP implicitly optimizes $\boldsymbol{H}$ by purifying the noisy $\tilde{A}$ and $\tilde{Y}$ with the reparameterization mechanism

RGIB with Self-Supervised Learning (***RGIB-SSL***)



RGIB **RGIB-SSL** **RGIB-REP**

$$\min \text{RGIB-SSL} \triangleq -\underbrace{\lambda_s(I(\boldsymbol{H}_1;\tilde{Y})+I(\boldsymbol{H}_2;\tilde{Y}))}_{\text{supervision}} - \underbrace{\lambda_u(H(\boldsymbol{H}_1)+H(\boldsymbol{H}_2))}_{\text{uniformity}} - \underbrace{\lambda_a I(\boldsymbol{H}_1;\boldsymbol{H}_2)}_{\text{alignment}}.$$

To achieve a tractable approximation of the MI terms

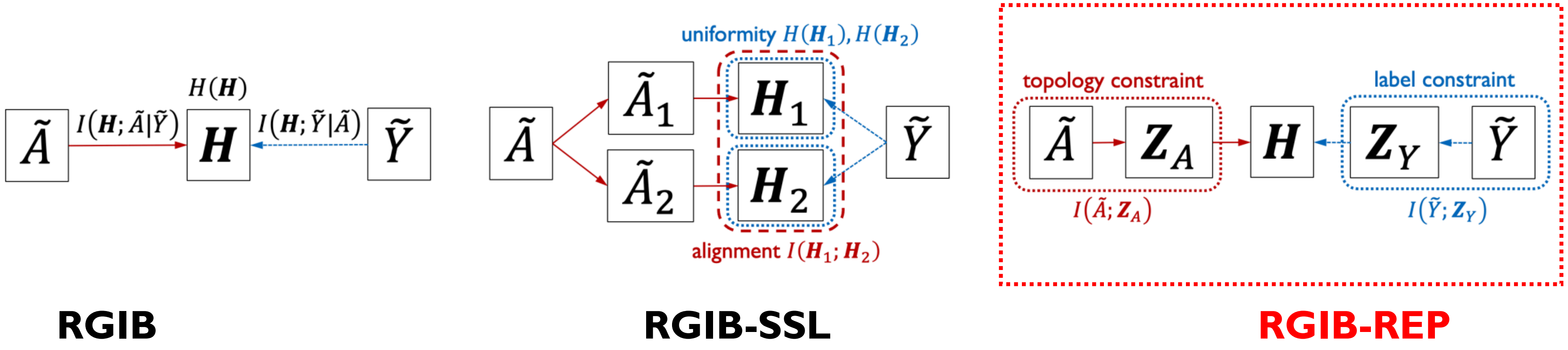
- we adopt the contrastive learning technique and contrast pair of samples,
- i.e., perturbed $\tilde{A}_1, \tilde{A}_2$ that are sampled from the augmentation distribution $\mathbb{P}(\tilde{A})$

$$\mathcal{R}_{align} = \textstyle\sum_{i=1}^{N} \mathcal{R}_i^{pos} + \mathcal{R}_i^{neg}$$

$$\mathcal{R}_{unif} = \textstyle\sum_{ij,mn}^{K} e^{-\left\|\boldsymbol{h}_{ij}^1-\boldsymbol{h}_{mn}^1\right\|_2^2} + e^{-\left\|\boldsymbol{h}_{ij}^2-\boldsymbol{h}_{mn}^2\right\|_2^2}$$

$$\mathcal{L} = \lambda_s \mathcal{L}_{cls} + \lambda_a \mathcal{R}_{align} + \lambda_u \mathcal{R}_{unif}$$

RGIB with Data Reparameterization (***RGIB-REP***)



$$\min \text{RGIB-REP} \triangleq -\underbrace{\lambda_s I(\boldsymbol{H};\boldsymbol{Z}_Y)}_{\text{supervision}} + \underbrace{\lambda_A I(\boldsymbol{Z}_A;\tilde{A})}_{\text{topology constraint}} + \underbrace{\lambda_Y I(\boldsymbol{Z}_Y;\tilde{Y})}_{\text{label constraint}}.$$

Latent variables $\boldsymbol{Z}_Y$ and $\boldsymbol{Z}_A$ are clean signals extracted from noisy $\tilde{Y}$ and $\tilde{A}$.

- their complementary parts $\boldsymbol{Z}_{Y'}$ and $\boldsymbol{Z}_{A'}$ are considered as noise, satisfying $\tilde{Y} = \boldsymbol{Z}_Y + \boldsymbol{Z}_{Y'}$ and $\tilde{A} = \boldsymbol{Z}_A + \boldsymbol{Z}_{A'}$.

$\mathrm{I}(\mathrm{H};\boldsymbol{Z}_Y)$ measures the supervised signals with selected samples $\boldsymbol{Z}_Y$

$\mathrm{I}(\boldsymbol{Z}_A;\tilde{A})$ and $\mathrm{I}(\boldsymbol{Z}_Y;\tilde{Y})$ help to select the clean and task-relevant information from $\tilde{A}$ and $\tilde{Y}$.

Outline

- Introduction
- Method
- Experiments
- Summary

Experiments | Method comparison under ***bilateral noise***

method	Cora			Citeseer			Pubmed			Facebook			Chameleon			Squirrel		
	20%	40%	60%	20%	40%	60%	20%	40%	60%	20%	40%	60%	20%	40%	60%	20%	40%	60%
Standard	.8111	.7419	.6970	.7864	.7380	.7085	.8870	.8748	.8641	.9829	.9520	.9438	.9616	.9496	.9274	.9432	.9406	.9386
DropEdge	.8017	.7423	.7303	.7635	.7393	.7094	.8711	.8482	.8354	.9811	.9682	.9473	.9568	.9548	.9407	.9439	.9377	.9365
NeuralSparse	.8190	.7318	.7293	.7765	.7397	.7148	.8908	.8733	.8630	.9825	.9638	.9456	.9599	.9497	.9402	.9494	.9309	.9297
PTDNet	.8047	.7559	.7388	.7795	.7423	.7283	.8872	.8733	.8623	.9725	.9674	.9485	.9607	.9514	.9424	.9485	.9326	.9304
Co-teaching	.8197	.7479	.7030	.7533	.7238	.7131	.8943	.8760	.8638	.9820	.9526	.9480	.9595	.9516	.9483	.9461	.9352	.9374
Peer loss	.8185	.7468	.7018	.7423	.7345	.7104	.8961	.8815	.8566	.9807	.9536	.9430	.9543	.9533	.9267	.9457	.9345	.9286
Jaccard	.8143	.7498	.7024	.7473	.7324	.7107	.8872	.8803	.8512	.9794	.9579	.9428	.9503	.9538	.9344	.9443	.9327	.9244
GIB	.8198	.7485	.7148	.7509	.7388	.7121	.8899	.8729	.8544	.9773	.9608	.9417	.9554	.9561	.9321	.9472	.9329	.9302
SupCon	.8240	.7819	.7490	.7554	.7458	.7299	.8853	.8718	.8525	.9588	.9508	.9297	.9561	.9531	.9467	.9473	.9348	.9301
GRACE	.7872	.6940	.6929	.7632	.7242	.6844	.8922	.8749	.8588	.8899	.8865	.8315	.8978	.8987	.8949	.9394	.9380	.9363
RGIB-REP	.8313	.7966	.7591	.7875	.7519	.7312	.9017	.8834	.8652	**.9832**	**.9770**	.9519	**.9723**	**.9621**	**.9519**	**.9509**	**.9455**	**.9434**
RGIB-SSL	**.8930**	**.8554**	**.8339**	**.8694**	**.8427**	**.8137**	**.9225**	**.8918**	**.8697**	.9829	.9711	**.9643**	.9655	.9592	.9500	.9499	.9426	.9425

Experiments | Method comparison under ***unilateral noise***

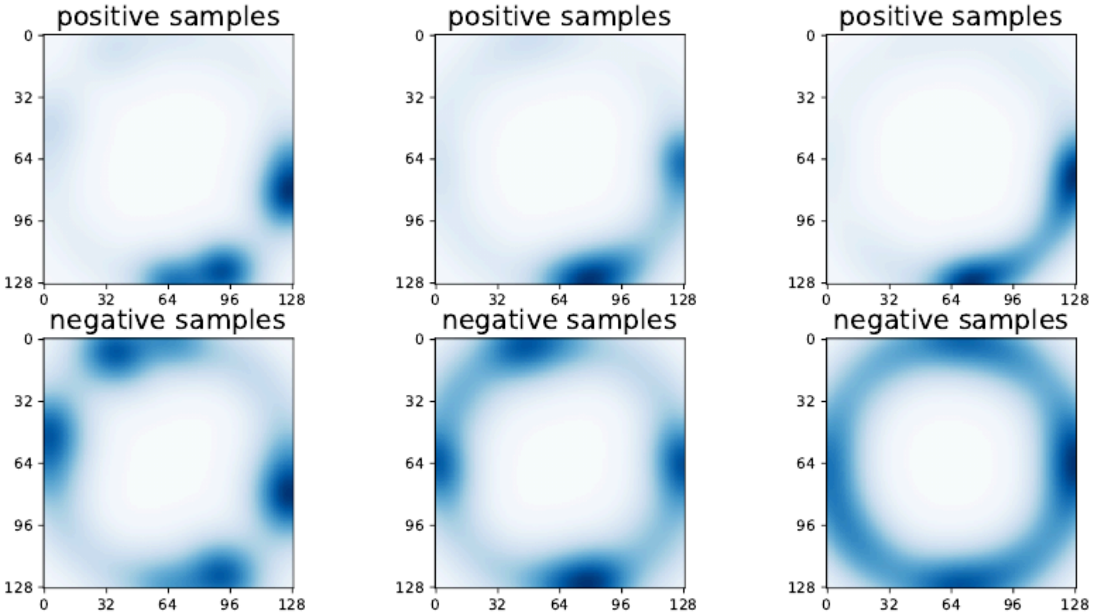
input noise	Cora 20%	Cora 40%	Cora 60%	Citeseer 20%	Citeseer 40%	Citeseer 60%	Pubmed 20%	Pubmed 40%	Pubmed 60%	Facebook 20%	Facebook 40%	Facebook 60%	Chameleon 20%	Chameleon 40%	Chameleon 60%	Squirrel 20%	Squirrel 40%	Squirrel 60%
Standard	.8027	.7856	.7490	.8054	.7708	.7583	.8854	.8759	.8651	.9819	.9668	.9622	.9608	.9433	.9368	.9416	.9395	.9411
DropEdge	.8338	.7826	.7454	.8025	.7730	.7473	.8682	.8456	.8376	.9803	.9685	.9531	.9567	.9433	.9432	.9426	.9376	.9358
NeuralSparse	.8534	.7794	.7637	.8093	.7809	.7468	.8931	.8720	.8649	.9712	.9691	.9583	.9609	.9540	.9348	.9469	.9403	.9417
PTDNet	.8433	.8214	.7770	.8119	.7811	.7638	.8903	.8776	.8609	.9725	.9668	.9493	.9610	.9457	.9360	.9469	.9400	.9379
Co-teaching	.8045	.7871	.7530	.8059	.7753	.7668	.8931	.8792	.8606	.9712	.9707	.9714	.9524	.9446	.9447	.9462	.9425	.9306
Peer loss	.8051	.7866	.7517	.8106	.7767	.7653	.8917	.8811	.8643	.9758	.9703	.9622	.9558	.9482	.9412	.9362	.9386	.9336
Jaccard	.8200	.7838	.7617	.8176	.7776	.7725	.8987	.8764	.8639	.9784	.9702	.9638	.9507	.9436	.9364	.9388	.9345	.9240
GIB	.8002	.8099	.7741	.8070	.7717	.7798	.8932	.8808	.8618	.9796	.9647	.9650	.9605	.9521	.9416	.9390	.9406	.9397
SupCon	.8349	.8301	.8025	.8076	.7767	.7655	.8867	.8739	.8558	.9647	.9517	.9401	.9606	.9536	.9468	.9372	.9343	.9305
GRACE	.7877	.7107	.6975	.7615	.7151	.6830	.8810	.8795	.8593	.9015	.8833	.8395	.8994	.9007	.8964	.9392	.9378	.9363
RGIB-REP	.8624	.8313	.8158	.8299	.7996	.7771	.9008	.8822	.8687	**.9833**	**.9723**	**.9682**	**.9705**	**.9604**	.9480	**.9495**	**.9432**	.9405
RGIB-SSL	**.9024**	**.8577**	**.8421**	**.8747**	**.8461**	**.8245**	**.9126**	**.8889**	**.8693**	.9821	.9707	.9668	.9658	.9570	**.9486**	.9479	.9429	**.9429**

label noise	Cora 20%	Cora 40%	Cora 60%	Citeseer 20%	Citeseer 40%	Citeseer 60%	Pubmed 20%	Pubmed 40%	Pubmed 60%	Facebook 20%	Facebook 40%	Facebook 60%	Chameleon 20%	Chameleon 40%	Chameleon 60%	Squirrel 20%	Squirrel 40%	Squirrel 60%
Standard	.8281	.8054	.8060	.7965	.7850	.7659	.9030	.9039	.9070	.9882	.9880	.9886	.9686	.9580	.9362	.9720	.9720	.9710
DropEdge	.8363	.8273	.8148	.7937	.7853	.7632	.9313	.9201	.9240	.9673	.9771	.9776	.9580	.9579	.9578	.9608	.9603	.9698
NeuralSparse	.8524	.8246	.8211	.7968	.7921	.7752	.9272	.9136	.9089	.9781	.9781	.9784	.9583	.9583	.9571	.9633	.9626	.9625
PTDNet	.8460	.8214	.8138	.7968	.7765	.7622	.9219	.9099	.9093	.9879	.9880	.9783	.9585	.9576	.9665	.9633	.9623	.9626
Co-teaching	.8446	.8209	.8157	.7974	.7877	.7913	.9315	.9291	.9319	.9762	.9797	.9638	.9642	.9650	.9533	.9675	.9641	.9655
Peer loss	.8325	.8036	.8069	.7991	.7990	.7751	.9126	.9101	.9210	.9769	.9750	.9734	.9621	.9501	.9569	.9636	.9694	.9696
Jaccard	.8289	.8064	.8148	.8061	.7887	.7689	.9098	.9135	.9096	.9702	.9725	.9758	.9603	.9659	.9557	.9529	.9512	.9501
GIB	.8337	.8137	.8157	.7986	.7852	.7649	.9037	.9114	.9064	.9742	.9703	.9771	.9651	.9582	.9489	.9641	.9628	.9601
SupCon	.8491	.8275	.8256	.8024	.7983	.7807	.9131	.9108	.9162	.9647	.9567	.9553	.9584	.9580	.9477	.9516	.9595	.9511
GRACE	.8531	.8237	.8193	.7909	.7630	.7737	.9234	.9252	.9255	.8913	.8972	.8887	.9053	.9074	.9075	.9171	.9174	.9166
RGIB-REP	.8554	.8318	.8297	.8083	.7846	.7945	.9357	.9343	.9332	**.9884**	**.9883**	**.9889**	**.9785**	**.9797**	**.9785**	**.9735**	**.9733**	**.9737**
RGIB-SSL	**.9314**	**.9224**	**.9241**	**.9204**	**.9218**	**.9250**	**.9594**	**.9604**	**.9613**	.9857	.9881	.9857	.9730	.9752	.9744	.9727	.9729	.9726

Experiments | The learned ***representations***

Table 5: Comparison of alignment. Here, std. is short for *standard training*, and SSL/REP are short for RGIB-SSL/RGIB-REP, respectively.

dataset	Cora			Citeseer		
method	std.	REP	SSL	std.	REP	SSL
clean	.616	<u>.524</u>	**.475**	.445	<u>.439</u>	**.418**
$\varepsilon=20\%$	.687	<u>.642</u>	**.543**	.586	<u>.533</u>	**.505**
$\varepsilon=40\%$	.695	<u>.679</u>	**.578**	.689	<u>.623</u>	**.533**
$\varepsilon=60\%$	.732	<u>.704</u>	**.615**	.696	<u>.647</u>	**.542**



(a) Standard (b) RGIB-REP (c) RGIB-SSL

Figure 6: Uniformity distribution on Citeseer with $\varepsilon=40\%$.

Experiments | Ablation study

Table 6: Comparison on different schedulers. SSL/REP are short for RGIB-SSL/RGIB-REP. Experiments are performed with a 4-layer GAT and $\epsilon = 40\%$ mixed edge noise.

dataset	Cora		Citeseer		Pubmed	
method	SSL	REP	SSL	REP	SSL	REP
$constant$	.8398	**.7927**	**.8227**	**.7742**	.8596	**.8416**
$linear(\cdot)$	.8427	.7653	.8167	.7559	**.8645**	.8239
$sin(\cdot)$	**.8436**	.7924	.8132	.7680	.8637	.8275
$cos(\cdot)$	.8334	.7833	.8088	.7647	.8579	.8372
$exp(\cdot)$	.8381	.7815	.8085	.7569	.8617	.8177

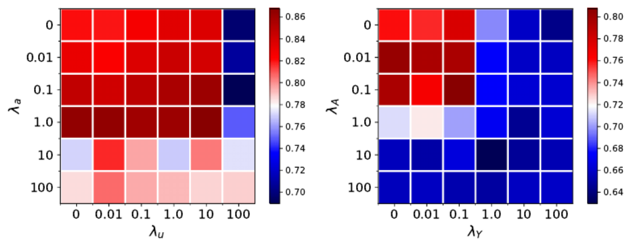
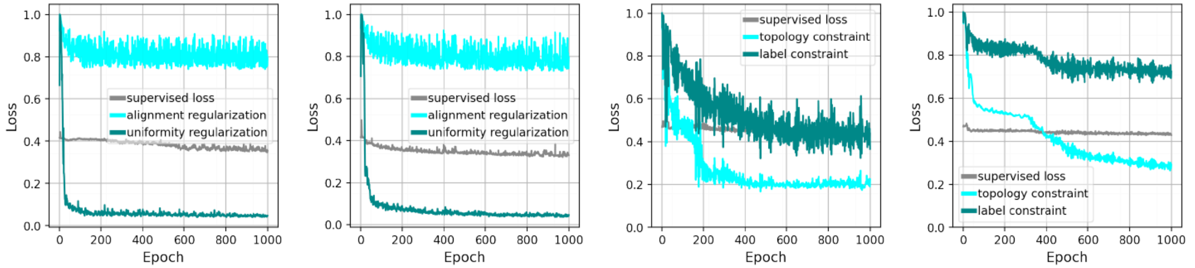


Figure 7: Grid search of hyper-parameter with RGIB-SSL (left) and RGIB-REP (right) on Cora dataset with bilateral noise $\epsilon = 40\%$. As can be seen, neither too large nor too small value can bring a good solution.



(a) RGIB-SSL on Cora (b) RGIB-SSL on Citeseer (c) RGIB-REP on Cora (d) RGIB-REP on Citeseer

Figure 8: Learning curves of RGIB-SSL and RGIB-REP with $\varepsilon = 40\%$ bilateral noise.

Table 7: Method comparison with a 4-layer GCN trained on the clean data.

method	Cora	Citeseer	Pubmed	Facebook	Chameleon	Squirrel
Standard	.8686	.8317	.9178	.9870	.9788	**.9725**
DropEdge	.8684	.8344	.9344	.9869	.9700	.9629
NeuralSparse	.8715	.8405	.9366	.9865	**.9803**	.9635
PTDNet	.8577	.8398	.9315	.9868	.9696	.9640
Co-teaching	.8684	.8387	.9192	.9771	.9698	.9626
Peer loss	.8313	.7742	.9085	.8951	.9374	.9422
Jaccard	.8413	.8005	.8831	.9792	.9703	.9610
GIB	.8582	.8327	.9019	.9691	.9628	.9635
SupCon	.8529	.8003	.9131	.9692	.9717	.9619
GRACE	.8329	.8236	.9358	.8953	.8999	.9165
RGIB-REP	.8758	.8415	.9408	**.9875**	.9792	.9680
RGIB-SSL	**.9260**	**.9148**	**.9593**	.9845	.9740	.9646

Table 8: Ablation study for RGIB-SSL and RGIB-REP with a 4-layer SAGE. Here, $\epsilon = 60\%$ indicates the 60% bilateral noise, while the ϵ_a/ϵ_y represent ratios of unilateral input/label noise.

variant	Cora			Chameleon		
	$\epsilon = 60\%$	$\epsilon_a = 60\%$	$\epsilon_y = 60\%$	$\epsilon = 60\%$	$\epsilon_a = 60\%$	$\epsilon_y = 60\%$
RGIB-SSL (full)	.8596	.8730	.8994	.9663	.9758	.9762
- w/o hybrid augmentation	.8150 (5.1%↓)	.8604 (1.4%↓)	.8757 (2.6%↓)	.9528 (1.3%↓)	.9746 (0.1%↓)	.9695 (0.6%↓)
- w/o self-adversarial	.8410 (2.1%↓)	.8705 (0.2%↓)	.8927 (0.7%↓)	.9655 (0.1%↓)	.9732 (0.2%↓)	.9746 (0.1%↓)
- w/o supervision ($\lambda_s = 0$)	.7480 (12.9%↓)	.7810 (10.5%↓)	.7820 (13.0%↓)	.8626 (10.7%↓)	.8628 (11.5%↓)	.8512 (12.8%↓)
- w/o alignment ($\lambda_a = 0$)	.8194 (4.6%↓)	.8510 (2.5%↓)	.8461 (5.9%↓)	.9613 (0.5%↓)	.9749 (0.1%↓)	.9722 (0.4%↓)
- w/o uniformity ($\lambda_u = 0$)	.8355 (2.8%↓)	.8621 (1.2%↓)	.8878 (1.3%↓)	.9652 (0.1%↓)	.9710 (0.4%↓)	.9751 (0.1%↓)
RGIB-REP (full)	.7611	.8487	.8095	.9567	.9706	.9676
- w/o edge selection ($Z_A \equiv \tilde{A}$)	.7515 (1.2%↓)	.8199 (3.3%↓)	.7890 (2.5%↓)	.9554(0.1%↓)	.9704 (0.1%↓)	.9661 (0.1%↓)
- w/o label selection ($Z_Y \equiv \tilde{Y}$)	.7533 (1.0%↓)	.8373 (1.3%↓)	.7847 (3.0%↓)	.9484(0.8%↓)	.9666 (0.4%↓)	.9594 (0.8%↓)
- w/o topology constraint ($\lambda_A = 0$)	.7355 (3.3%↓)	.7699 (9.2%↓)	.7969 (1.5%↓)	.9503(0.6%↓)	.9658 (0.4%↓)	.9635 (0.4%↓)
- w/o label constraint ($\lambda_Y = 0$)	.7381 (3.0%↓)	.8106 (4.4%↓)	.8032 (0.7%↓)	.9443(1.2%↓)	.9665 (0.4%↓)	.9669 (0.1%↓)

Outline

- Introduction
- Method
- Experiments
- Summary

Take home messages

1. In this work, we study the problem of link prediction with the ***bilateral edge noise***.
2. We propose the ***Robust Graph Information Bottleneck (RGIB)*** principle, aiming to extract reliable signals via decoupling and balancing the mutual information among inputs, labels, and representation.
3. Regarding the instantiation of RGIB, the self-supervised learning technique and data reparametrization mechanism are utilized to establish the ***RGIB-SSL and RGIB-REP***, respectively.
4. ***Empirical studies*** verify the denoising effect of the proposed RGIB under different noisy scenarios.

Thanks for your listening!

Zhanke Zhou cszkzhou@comp.hkbu.edu.hk