Appendices

A Additional Dataset Details

In this section, we provide some additional dataset details. All the datasets used in this work are publicly available.

Citation Networks CORA, CITESEER, and PUBMED are citation networks that were first used by Yang et al. (2016) and then commonly used as benchmarks in GNN-related literature (Kipf and Welling, 2016a; Veličković et al., 2017). In these citation networks, the nodes are published papers and features are bag-of-word vectors extracted from the corresponding paper. Links represent the citation relation between papers. We loaded the datasets with the `DGL`[2] package.

Social Network The FACEBOOK dataset[3] is a social network constructed from friends lists from Facebook (McAuley and Leskovec, 2012). The nodes are Facebook users and links indicate the friendship relation on Facebook. The node features were constructed from the user profiles and anonymized by McAuley and Leskovec (2012).

Drug-Drug Interaction Network The OGB-DDI dataset was constructed from a public Drug database (Wishart et al., 2018) and provided by the Open Graph Benchmark (OGB) (Hu et al., 2020). Each node in this graph represents an FDA-approved or experimental drug and edges represent the existence of unexpected effect when the two drugs are taken together. This dataset does not contain any node features, and it can be downloaded with the dataloader[4] provided by OGB.

B Details on Implementation and Hyperparameters

All the experiments in this work were conducted on a Linux server with Intel Xeon Gold 6130 Processor (16 Cores @2.1Ghz), 96 GB of RAM, and 4 RTX 2080Ti cards (11 GB of RAM each). Our method are implemented with `Python 3.8.5` with `PyTorch`. A list of used packages can be found in `requirements.txt`.

Baseline Methods For baseline methods, we use official code packages from the authors for MVGRL[5] (Hassani and Khasahmadi, 2020), SEAL[6] (Zhang and Chen, 2018), and LGLP[7] (Cai et al., 2021). We use a public implementation for VGAE[8] (Kipf and Welling, 2016b) and OGB implementations[9] for Node2Vec and baseline GNNs. For fair comparison, we set the size of node/link representations to be 256 of all methods.

CFLP We use the Adam optimizer with a simple cyclical learning rate scheduler (Smith, 2017), in which the learning rate waves cyclically between the given learning rate (lr) and 1e-4 in every 70 epochs (50 warmup steps and 20 annealing steps). We implement the GNN encoders with `torch_geometric`[10] (Fey and Lenssen, 2019). Same with the baselines, we set the size of all hidden layers and node/link representations of CFLP as 256. The graph encoders all have three layers and JKNet has mean pooling for the final aggregation layer. The decoder is a 3-layer MLP with a hidden layer of size 64 and ELU as the nonlinearity. As the Euclidean distance used in Eq. (3) has a range of $[0, \infty)$, the value of γ depends on the distribution of all-pair node embedding distances, which varies for different datasets. Therefore, we set the value of γ as the γ_{pct}-percentile of all-pair node embedding distances. Commands for reproducing the experiments are included in `README.md`.

[2] https://github.com/dmlc/dgl
[3] https://snap.stanford.edu/data/ego-Facebook.html
[4] https://ogb.stanford.edu/docs/linkprop/#data-loader
[5] https://github.com/kavehhassani/mvgrl
[6] https://github.com/facebookresearch/SEAL_OGB
[7] https://github.com/LeiCaiwsu/LGLP
[8] https://github.com/DaehanKim/vgae_pytorch
[9] https://github.com/snap-stanford/ogb/tree/master/examples/linkproppred/ddi
[10] https://pytorch-geometric.readthedocs.io/en/latest/

Table 6: Link prediction performances measured by Hits@50. Best performance and best baseline performance are marked with bold and underline, respectively.

	CORA	CITESEER	PUBMED	FACEBOOK	OGB-DDI
Node2Vec	63.64±0.76	54.57±1.40	50.73±1.10	43.91±1.03	24.34±1.67
MVGRL	29.97±3.06	26.48±0.98	16.96±0.56	17.06±0.19	12.03±0.11
VGAE	60.36±2.71	54.68±3.15	41.98±0.31	51.36±0.93	23.00±1.66
SEAL	51.68±2.85	54.55±1.77	42.85±2.03	57.20±1.85	40.85±2.97
LGLP	<u>71.43</u>±0.75	<u>69.98</u>±0.16	–	56.22±0.49	–
GCN	64.93±1.62	63.38±1.73	39.20±6.47	<u>69.90</u>±0.65	73.70±3.99
GSAGE	63.18±3.39	61.71±2.43	<u>54.81</u>±2.67	62.53±4.24	86.83±3.85
JKNet	62.64±1.40	62.26±2.10	45.16±3.18	68.81±1.76	<u>91.48</u>±2.41
Our proposed CFLP with different graph encoders					
CFLP w/ GCN	72.61±0.92	69.85±1.11	55.00±1.95	70.47±0.77	62.47±1.53
CFLP w/ GSAGE	73.25±0.94	64.75±2.27	58.16±1.40	63.89±2.08	83.32±3.61
CFLP w/ JKNet	**75.49**±1.54	**77.01**±1.92	**62.80**±0.79	**71.41**±0.61	**93.07**±1.14

Hyperparameter Searching Space We manually tune the following hyperparameters over range: $lr \in \{0.005, 0.01, 0.05, 0.1, 0.2\}$, $\alpha \in \{0.001, 0.01, 0.1, 1, 2\}$, $\beta \in \{0.001, 0.01, 0.1, 1, 2\}$, $\gamma_{pct} \in \{10, 20, 30\}$.

Treatments For the graph clustering or community detection methods we used as treatments, we use the implementation from `scikit-network`[11] for Louvain (Blondel et al., 2008), SpecC (Ng et al., 2001), PropC (Raghavan et al., 2007), and Ward (Ward Jr, 1963). We used implementation of K-core (Bader and Hogue, 2003) from `networkx`.[12] We used SBM (Karrer and Newman, 2011) from a public implementation by Funke and Becker (2019).[13] For CommN and Katz, we set $T_{i,j} = 1$ if the number of common neighbors or Katz index between v_i and v_j are greater or equal to 2 or 2 times the average of all Katz index values, respectively. For SpecC, we set the number of clusters as 16. For SBM, we set the number of communities as 16. These settings are fixed for all datasets.

C Additional Experimental Results

Link Prediction Tables 6 and 7 show the link prediction performance of Hits@50 and Average Precision (AP) by all methods. LGLP on PUBMED and OGB-DDI are missing due to the out of memory error when running the code package from the authors. Similar to the results in Tables 2 and 3, we observe that our CFLP on different graph encoders achieve similar or better performances compared with baselines, with the only exception of AP on FACEBOOK where most methods have close-to-perfect AP. We observe that CFLP with JKNet almost consistently achieves the best performance and outperforms baselines significantly on Hits@50. Specifically, compared with the best baseline, CFLP improves relatively by 6.8% and 0.9% on Hits@50 and AP, respectively.

Ablation Study For the ablative studies of $\mathcal{L}_{CF}$ (Eq. (9)) and $\mathcal{L}_{disc}$ (Eq. (10)), we show their effect by removing them from the integrated loss function (Eq. (11)). Table 8 shows the results of CFLP on CORA and CITESEER under different settings ($\alpha = 0$, $\beta = 0$, $\alpha = \beta = 0$, and original setting). We observe that CFLP in the original setting achieves the best performance. The performance drops significantly when having $\alpha = 0$, i.e., not using any counterfactual data during training. We note that having $\beta = 0$, i.e., not using the discrepancy loss, also lowers the performance. Therefore, both $\mathcal{L}_{CF}$ and $\mathcal{L}_{disc}$ are essential for improving the link prediction performance.

Sensitivity Analysis of γ Figure 3 shows the Hits@20 and AUC performance on link prediction of CFLP (with JKNet) on CORA and CITESEER with different treatments and γ_{pct}. We observe that the performance is generally good when $10 \leq \gamma_{pct} \leq 20$ and slightly worse when the value of γ_{pct} is too small or too large, showing that CFLP is robust to γ and the optimal γ is easy to find.

[11] https://scikit-network.readthedocs.io/

[12] https://networkx.org/documentation/

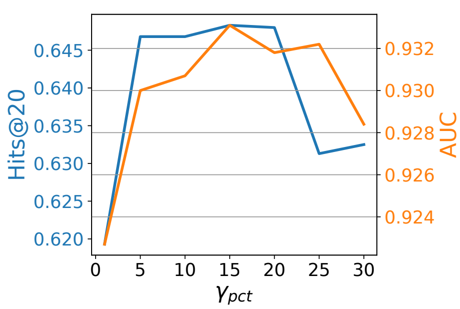
[13] https://github.com/funket/pysbm

Table 7: Link prediction performances measured by Average Precision (AP). Best performance and best baseline performance are marked with bold and underline, respectively.

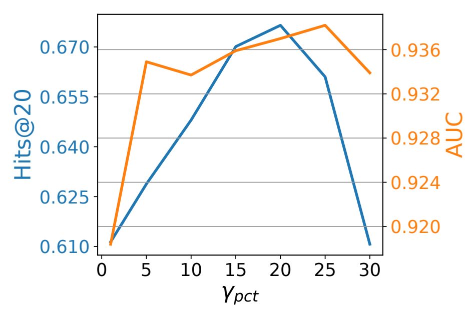
	CORA	CITESEER	PUBMED	FACEBOOK	OGB-DDI
Node2Vec	88.53±0.42	84.42±0.48	87.15±0.12	99.07±0.02	98.39±0.04
MVGRL	76.47±3.07	67.40±0.52	82.00±0.97	82.37±0.35	81.12±1.77
VGAE	89.89±0.50	86.97±0.78	95.97±0.16	98.60±0.04	95.28±0.11
SEAL	89.08±0.57	88.55±0.32	96.33±0.28	<u>**99.51**</u>±0.03	98.39±0.21
LGLP	<u>93.05</u>±0.03	<u>91.62</u>±0.09	–	98.62±0.01	–
GCN	91.42±0.45	90.87±0.52	96.19±0.88	99.42±0.02	99.86±0.03
GSAGE	91.52±0.46	89.43±1.15	<u>96.93</u>±0.11	99.27±0.06	99.93±0.01
JKNet	90.50±0.22	90.42±1.34	96.56±0.31	99.41±0.02	<u>99.95</u>±0.01
Our proposed CFLP with different graph encoders					
CFLP w/ GCN	93.77±0.49	91.84±0.20	97.16±0.08	99.40±0.01	99.60±0.03
CFLP w/ GSAGE	93.55±0.49	90.80±0.87	97.10±0.08	99.29±0.06	99.88±0.04
CFLP w/ JKNet	**94.24**±0.28	**93.92**±0.41	**97.69**±0.13	99.35±0.02	**99.96**±0.01

Table 8: Link prediction performance of CFLP (w/ JKNet) on CORA and CITESEER when removing $\mathcal{L}_{CF}$ or $\mathcal{L}_{disc}$ or both versus normal setting.

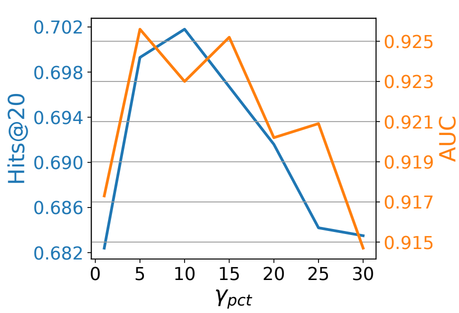
	CORA		CITESEER	
	Hits@20	AUC	Hits@20	AUC
CFLP ($\alpha = 0$)	58.58±0.23	89.16±0.93	65.49±2.18	91.01±0.64
CFLP ($\beta = 0$)	62.27±0.84	92.96±0.34	66.92±1.84	91.98±0.17
CFLP ($\alpha = \beta = 0$)	58.52±0.83	88.79±0.28	64.69±3.25	90.61±0.64
CFLP	**65.57**±1.05	**93.05**±0.24	**68.09**±1.49	**92.12**±0.47



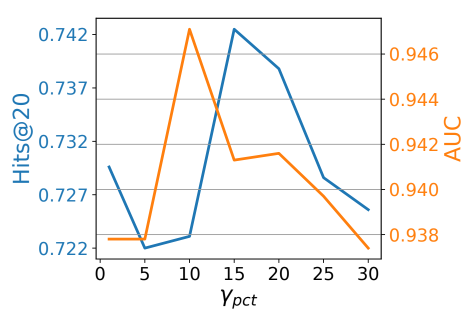
(a) Performances of CFLP on CORA when using K-core as treatment.



(b) Performances of CFLP on CORA when using SBM as treatment.



(c) Performances of CFLP on CITESEER when using K-core as treatment.



(d) Performances of CFLP on CITESEER when using SBM as treatment.

Figure 3: Hits@20 and AUC performances of CFLP (w/ JKNet) on CORA and CITESEER with different treatments w.r.t. different γ_{pct} value.