IEEE ICDM 2020
November 17-20, 2020

Sub-graph Contrast for Scalable Self-Supervised Graph Representation Learning

Yizhu Jiao[1], Yun Xiong[1], Jiawei Zhang[2], Yao Zhang[1],
Tianqi Zhang[1], Yangyong Zhu[1]

[1] Shanghai Key Laboratory of Data Science, Fudan University, China
[2] IFM Lab, Florida State University, USA

Datalogy
上海市数据科学重点实验室
Shanghai Key Laboratory of Data Science

IFM Lab
Information Fusion and Mining Laboratory

Learning from Unlabeled Data

Labeled Data

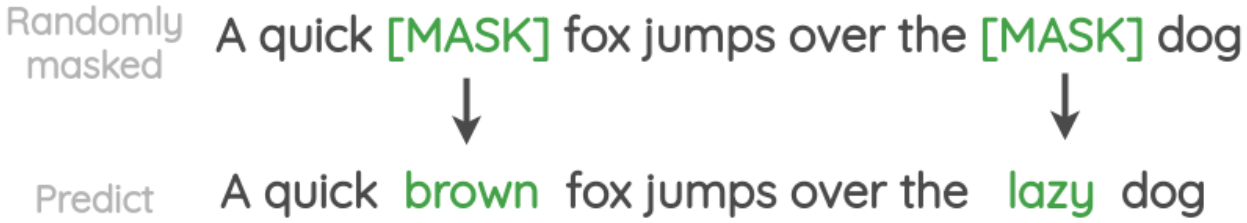
Expensive, Scarce

Unlabeled Data

Accessible, Abundant

Self-Supervised Learning in NLP and CV

- NLP : Masked Language Modeling



Yinhan Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach

- CV : Contrastive Learning of Visual Representations

Ting Chen et al. A Simple Framework for Contrastive Learning of Visual Representations

Self-Supervised Learning in Graph?

Challenges
1.Unordered vertexes
2.Extensive connections
3.Large scale

Image source :https://www.pinterest.fr

Self-Supervised Learning in Graph?

A node is strongly correlated to its regional neighbors while long-distance nodes hardly influence it.

Self-Supervised Learning in Graph?

Learn node representations
only with regional neighborhood ?

Subgraph-Based Representation Learning

Complete graph

Encoder

Subgraph

Encoder

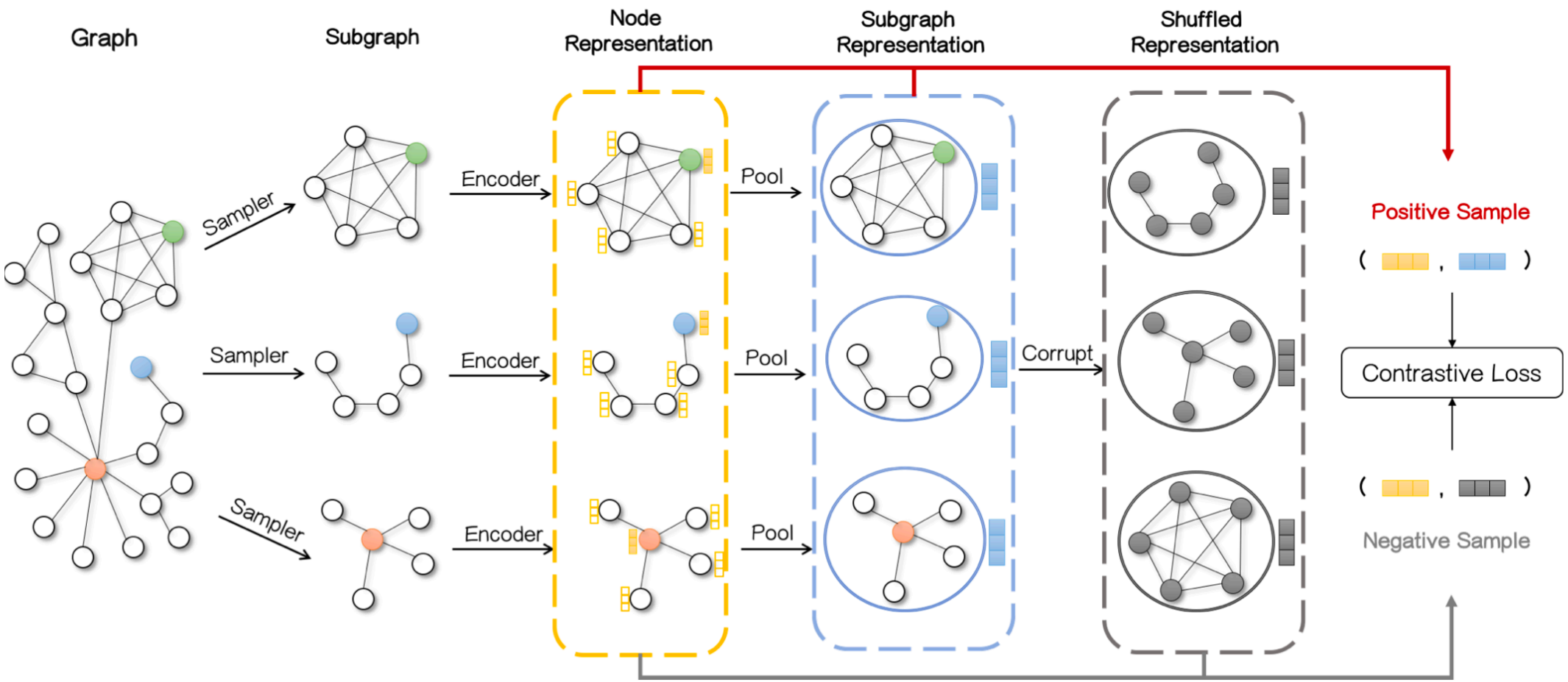
- Benefits
  - Lower training time
  - Lower computation memory costs
  - Parallel computation

Subgraph-Based Representation Learning

Question

1. How to sample subgraphs ?
2. How to gain node representations ?
3. How to design self-supervised pretext task ?

Architecture of SUBG-CON



Definition

Definition 1. (Data Augmentation on Graph): Given a graph $\mathcal{G} = (\mathbf{X}, \mathbf{A})$, where $\mathbf{X}$ denotes node features and $\mathbf{A}$ denotes relations, data augmentation is a strategy to produce a series of variant graphs $\mathcal{G}' = (\mathbf{X}', \mathbf{A}')$ using transformations on features and relations of $\mathcal{G}$.

- Such as node masking, feature corruption.
- Here we use subgraph sampling.

Subgraph Sampling

- To sample a context subgraph with regional structure information $\mathcal{G}_i = (\mathbf{X}_i, \mathbf{A}_i) \sim \mathcal{S}(\mathbf{X}, \mathbf{A})$
- Calculate importance scores by PPR

$$\mathbf{S} = \alpha \cdot (\mathbf{I} - (1 - \alpha) \cdot \bar{\mathbf{A}})$$

- Choose the most important neighbors

$$idx = top\_rank(\mathbf{S}(i, :), k)$$

- Process adjacent matrix and feature matrix

$$\mathbf{X}_i = \mathbf{X}_{idx,:}, \quad \mathbf{A}_i = \mathbf{A}_{idx,idx}$$

Subgraph Encoding

- Encode a context subgraph $\mathcal{G}_i = (\mathbf{X}_i, \mathbf{A}_i)$ of a central node i

$$\mathbf{H}_i = \mathcal{E}(\mathbf{X}_i, \mathbf{A}_i)$$

- Central node representation

$$\mathbf{h}_i = \mathcal{C}(\mathbf{H}_i)$$

- Subgraph-level representation

$$\mathbf{s}_i = \mathcal{R}(\mathbf{H}_i)$$

Contrastive Learning via Central Node and Context Sub-graph

- Positive sample
- Negative sample
  - Corruption function

$$\{\widetilde{\mathbf{s}}_1, \widetilde{\mathbf{s}}_2 ..., \widetilde{\mathbf{s}}_M\} \sim \mathcal{P}(\{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_m\})$$

- Margin loss function

$$\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} \mathbb{E}_{(\mathbf{X}, \mathbf{A})} (- \max(\sigma(\mathbf{h}_i \mathbf{s}_i) - \sigma(\mathbf{h}_i \widetilde{\mathbf{s}}_i) + \epsilon, 0))$$

Experiments

- Downstream task
  - Node classification
- Datasets

Table I. Dataset statistics

	Dataset	Type	Nodes	Edges	Degree	Features	Classes	Train / Val / Test
Small-scale	Cora	Citation network	2,708	5,429	4.0	1,433	7	0.05 / 0.18 / 0.37
	Citeseer	Citation network	3,327	4,732	2.8	3,703	6	0.04 / 0.15 / 0.30
	Pubmed	Citation network	19,717	44,338	4.5	500	3	0.003 / 0.03 / 0.05
Large-scale	PPI	Protein network	56,944	818,716	28.8	50	121	0.79 / 0.11 / 0.10
	Flickr	Social network	89,250	899,756	20.2	500	7	0.50 / 0.25 / 0.25
	Reddit	Social network	232,965	11,606,919	99.6	602	41	0.66 / 0.10 / 0.24

Node Classification

Table II. Performance comparison with different methods on node classification. The second column illustrates the data used by each algorithm in the training phase, where **X**, **A**, and **Y** denotes features, adjacency matrix, and labels, respectively. **OOM**: Out of memory.

Algorithm	Available data	Cora	Citeseer	Pubmed	PPI	Flickr	Reddit
Raw features	**X**	56.6 ± 0.4	57.8 ± 0.2	69.1 ± 0.2	42.5 ± 0.3	20.3 ± 0.2	58.5 ± 0.1
DeepWalk	**A**	67.2	43.2	65.3	52.9	27.9	32.4
Unsup-GraphSAGE	**X, A**	75.2 ± 1.5	59.4 ± 0.9	70.1 ± 1.4	46.5 ± 0.7	36.5 ± 1.0	90.8 ± 1.1
DGI	**X, A**	82.3 ± 0.6	71.8 ± 0.7	76.8 ± 0.6	63.8 ± 0.2	42.9 ± 0.1	94.0 ± 0.1
GMI	**X, A**	83.0 ± 0.3	73.0 ± 0.3	79.9 ± 0.2	65.0 ± 0.0	44.5 ± 0.2	95.0 ± 0.0
GCN	**X, A, Y**	81.4 ± 0.6	70.3 ± 0.7	76.8 ± 0.6	51.5 ± 0.6	48.7 ± 0.3	93.3 ± 0.1
GAT	**X, A, Y**	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3	**97.3 ± 0.2**	**OOM**	**OOM**
FastGCN	**X, A, Y**	78.0 ± 2.1	63.5 ± 1.8	74.4 ± 0.8	63.7 ± 0.6	48.1 ± 0.5	89.5 ± 1.2
GraphSAGE	**X, A, Y**	79.2 ± 1.5	71.2 ± 0.5	73.1 ± 1.4	51.3 ± 3.2	**50.1 ± 1.3**	92.1 ± 1.1
SUBG-CON	**X, A**	**83.5 ± 0.5**	**73.2 ± 0.2**	**81.0 ± 0.1**	**66.9 ± 0.2**	**48.8 ± 0.1**	**95.2 ± 0.0**

Node Classification

Table II. Performance comparison with different methods on node classification. The second column illustrates the data used by each algorithm in the training phase, where **X**, **A**, and **Y** denotes features, adjacency matrix, and labels, respectively. **OOM**: Out of memory.

Algorithm	Available data	Cora	Citeseer	Pubmed	PPI	Flickr	Reddit
Raw features	**X**	56.6 ± 0.4	57.8 ± 0.2	69.1 ± 0.2	42.5 ± 0.3	20.3 ± 0.2	58.5 ± 0.1
DeepWalk	**A**	67.2	43.2	65.3	52.9	27.9	32.4
Unsup-GraphSAGE	**X, A**	75.2 ± 1.5	59.4 ± 0.9	70.1 ± 1.4	46.5 ± 0.7	36.5 ± 1.0	90.8 ± 1.1
DGI	**X, A**	82.3 ± 0.6	71.8 ± 0.7	76.8 ± 0.6	63.8 ± 0.2	42.9 ± 0.1	94.0 ± 0.1
GMI	**X, A**	83.0 ± 0.3	73.0 ± 0.3	79.9 ± 0.2	65.0 ± 0.0	44.5 ± 0.2	95.0 ± 0.0
GCN	**X, A, Y**	81.4 ± 0.6	70.3 ± 0.7	76.8 ± 0.6	51.5 ± 0.6	48.7 ± 0.3	93.3 ± 0.1
GAT	**X, A, Y**	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3	**97.3 ± 0.2**	**OOM**	**OOM**
FastGCN	**X, A, Y**	78.0 ± 2.1	63.5 ± 1.8	74.4 ± 0.8	63.7 ± 0.6	48.1 ± 0.5	89.5 ± 1.2
GraphSAGE	**X, A, Y**	79.2 ± 1.5	71.2 ± 0.5	73.1 ± 1.4	51.3 ± 3.2	**50.1 ± 1.3**	92.1 ± 1.1
SUBG-CON	**X, A**	**83.5 ± 0.5**	**73.2 ± 0.2**	**81.0 ± 0.1**	**66.9 ± 0.2**	**48.8 ± 0.1**	**95.2 ± 0.0**

SUBG-CON outperforms all the unsupervised approaches.

Node Classification

Table II. Performance comparison with different methods on node classification. The second column illustrates the data used by each algorithm in the training phase, where $\mathbf{X}$, $\mathbf{A}$, and $\mathbf{Y}$ denotes features, adjacency matrix, and labels, respectively. **OOM**: Out of memory.

Algorithm	Available data	Cora	Citeseer	Pubmed	PPI	Flickr	Reddit
Raw features	$\mathbf{X}$	56.6 ± 0.4	57.8 ± 0.2	69.1 ± 0.2	42.5 ± 0.3	20.3 ± 0.2	58.5 ± 0.1
DeepWalk	$\mathbf{A}$	67.2	43.2	65.3	52.9	27.9	32.4
Unsup-GraphSAGE	$\mathbf{X}$, $\mathbf{A}$	75.2 ± 1.5	59.4 ± 0.9	70.1 ± 1.4	46.5 ± 0.7	36.5 ± 1.0	90.8 ± 1.1
DGI	$\mathbf{X}$, $\mathbf{A}$	82.3 ± 0.6	71.8 ± 0.7	76.8 ± 0.6	63.8 ± 0.2	42.9 ± 0.1	94.0 ± 0.1
GMI	$\mathbf{X}$, $\mathbf{A}$	83.0 ± 0.3	73.0 ± 0.3	79.9 ± 0.2	65.0 ± 0.0	44.5 ± 0.2	95.0 ± 0.0
GCN	$\mathbf{X}$, $\mathbf{A}$, $\mathbf{Y}$	81.4 ± 0.6	70.3 ± 0.7	76.8 ± 0.6	51.5 ± 0.6	48.7 ± 0.3	93.3 ± 0.1
GAT	$\mathbf{X}$, $\mathbf{A}$, $\mathbf{Y}$	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3	**97.3 ± 0.2**	**OOM**	**OOM**
FastGCN	$\mathbf{X}$, $\mathbf{A}$, $\mathbf{Y}$	78.0 ± 2.1	63.5 ± 1.8	74.4 ± 0.8	63.7 ± 0.6	48.1 ± 0.5	89.5 ± 1.2
GraphSAGE	$\mathbf{X}$, $\mathbf{A}$, $\mathbf{Y}$	79.2 ± 1.5	71.2 ± 0.5	73.1 ± 1.4	51.3 ± 3.2	**50.1 ± 1.3**	92.1 ± 1.1
SUBG-CON	$\mathbf{X}$, $\mathbf{A}$	**83.5 ± 0.5**	**73.2 ± 0.2**	**81.0 ± 0.1**	**66.9 ± 0.2**	**48.8 ± 0.1**	**95.2 ± 0.0**

SUBG-CON is competitive compared with supervised GNNs.

Design of Encoder

Table IV. Comparison with different graph neural network encoders.

Dataset	GCN	GCN+Skip	GAT	GIN
Cora	82.1	**83.5**	**83.5**	83.0
Citeseer	72.4	**73.2**	73.0	73.0
Pubmed	79.2	**81.1**	80.0	80.4
PPI	66.2	**66.9**	66.8	66.0
Flickr	**48.8**	48.2	48.7	48.3
Reddit	**95.2**	94.5	94.9	93.9

Simpler GNNs are suitable for subgraph encoding.

Choose of Objective

Table III. Studied Objective functions.

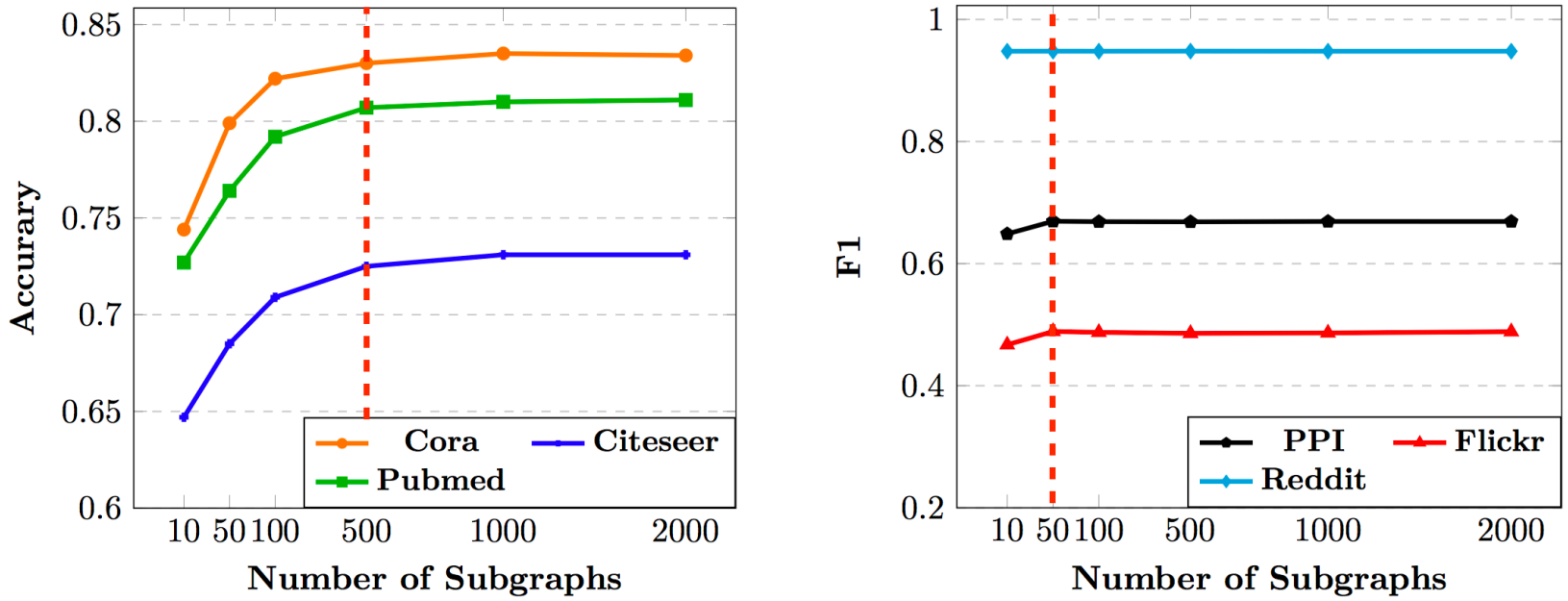
Name	Objective Function
Margin Loss	$-\max(\sigma(\mathbf{hs}) - \sigma(\mathbf{h\widetilde{s}}) + \epsilon, 0)$
Logistic Loss	$\log \sigma(\mathbf{hs}) + \log \sigma(-\mathbf{h\widetilde{s}})$
BPR Loss	$\log \sigma(\mathbf{hs} - \mathbf{h\widetilde{s}})$

Table V. Comparison with models trained with different objective functions.

	Cora	Citeseer	Pubmed	PPI	Flickr	Reddit
Margin	**83.5**	**73.2**	**81.0**	**66.9**	**48.8**	**95.2**
Logistic	82.4	72.2	79.8	66.8	48.5	95.0
BPR	81.7	72.0	79.9	66.8	48.6	94.8

Because subgraphs from the same original graph can overlap.

Train with A Few Subgraphs



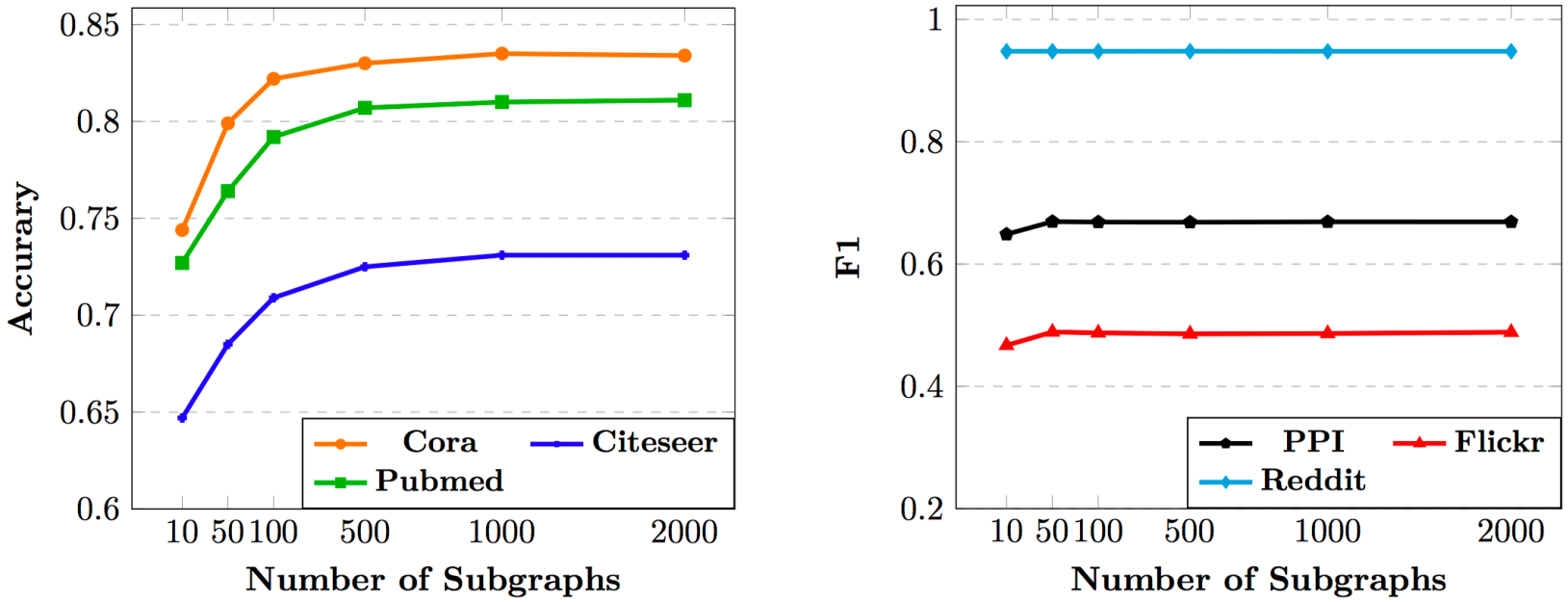
(a) Small-Scale Datasets (b) Large-Scale Datasets

Fig. 3. The effectiveness of training the encoder with different numbers of sampled subgraphs

Best value for #subgraph

Dataset	Small-scale	Large-scale
#Subgraph	500	50

Train with A Few Subgraphs



(a) Small-Scale Datasets (b) Large-Scale Datasets

Fig. 3. The effectiveness of training the encoder with different numbers of sampled subgraphs

A few subgraphs is enough to train the encoder.

WHY?

Train with A Few Subgraphs

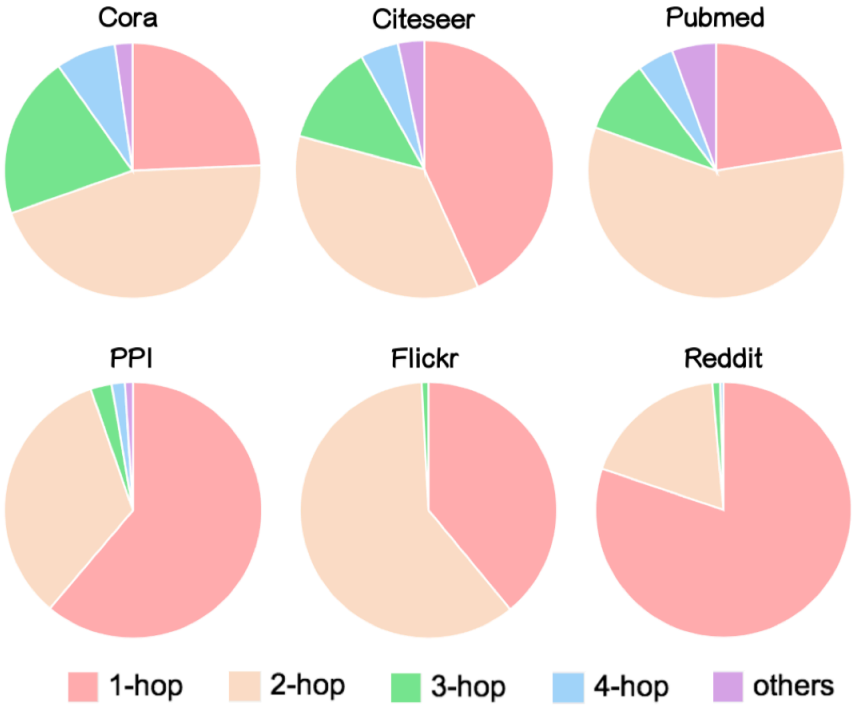


Fig. 4. Composition of context subgraphs for different datasets. The pie chart indicates the proportion of neighbors of different distances from central nodes in the context subgraphs.

Subgraphs have simple and similar structures

⇒ Faster convergence

Efficiency

- Lower training time
- Lower computation memory

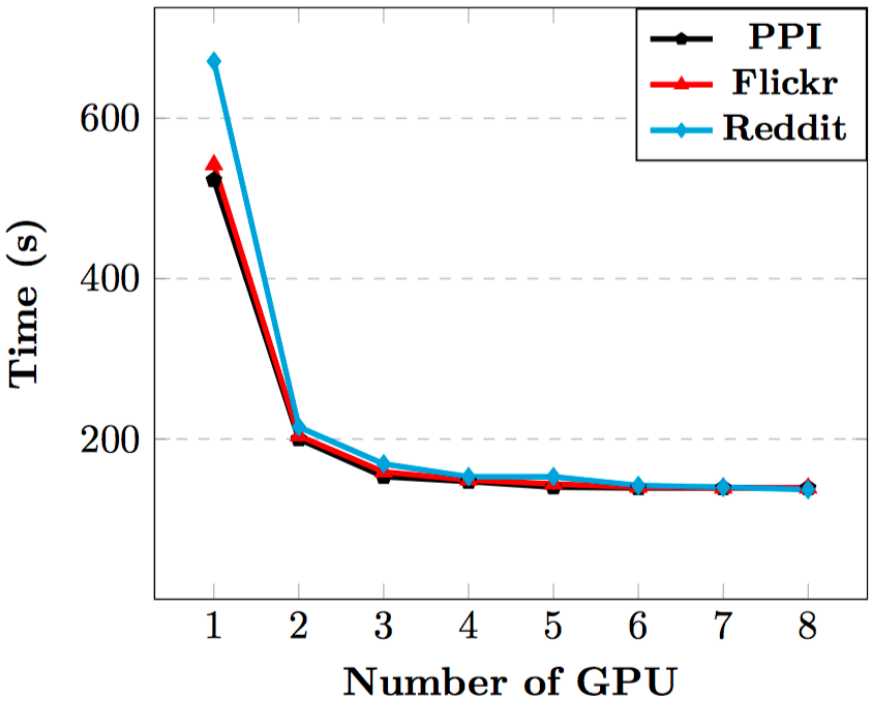
Table VI. Efficiency of SUBG-CON on three small-scale datasets. We train the encoder with 500 context subgraphs.

Dataset	Algorithm	Training Time	Memory
Cora	DGI	27s	3597MB
	GMI	104s	3927MB
	SUBG-CON	**14s**	**1586MB**
Citeseer	DGI	48s	4867MB
	GMI	410s	7605MB
	SUBG-CON	**12s**	**1163MB**
Pubmed	DGI	104s	10911MB
	GMI	1012s	12115MB
	SUBG-CON	**26s**	**975MB**

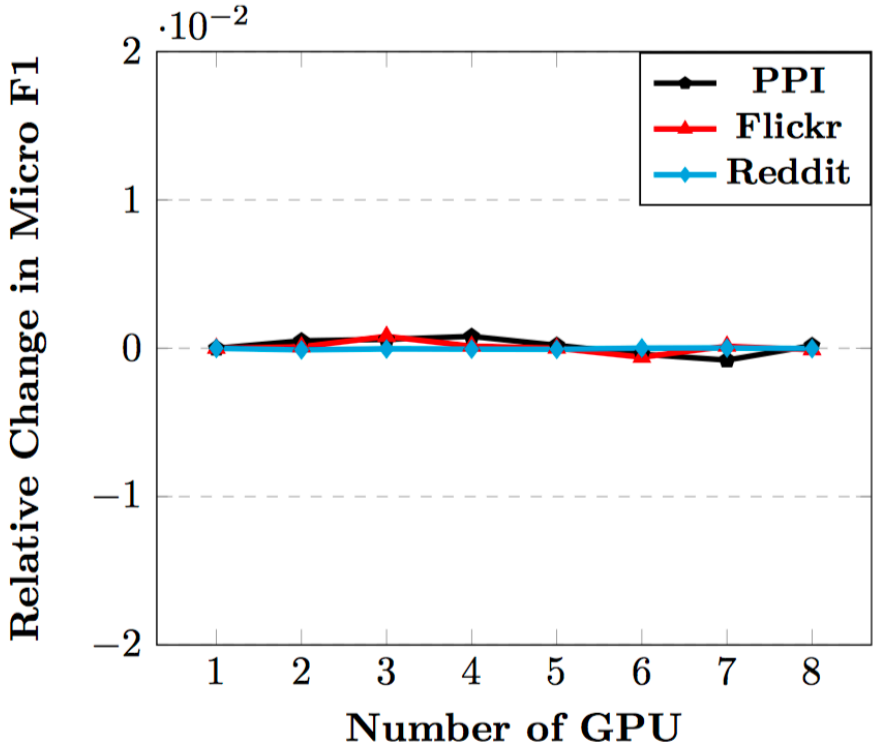
Table VII. Efficiency of SUBG-CON on three large-scale datasets. We train the encoder with 50 context subgraphs.

Dataset	Algorithm	Training Time	Memory
PPI	DGI	44s	10171MB
	GMI	561s	12101MB
	SUBG-CON	**3s**	**1349MB**
Flickr	DGI	518s	5028MB
	GMI	1247s	9768MB
	SUBG-CON	**12s**	**1903MB**
Reddit	DGI	4071s	8517MB
	GMI	9847s	12098MB
	SUBG-CON	**25s**	**3805MB**

Parallel Computation



(a) Training Time



(b) Performance

Efficiency

- Training with subgraphs
- Training with a few subgraphs
- Parallel computation

Subgraph size analysis

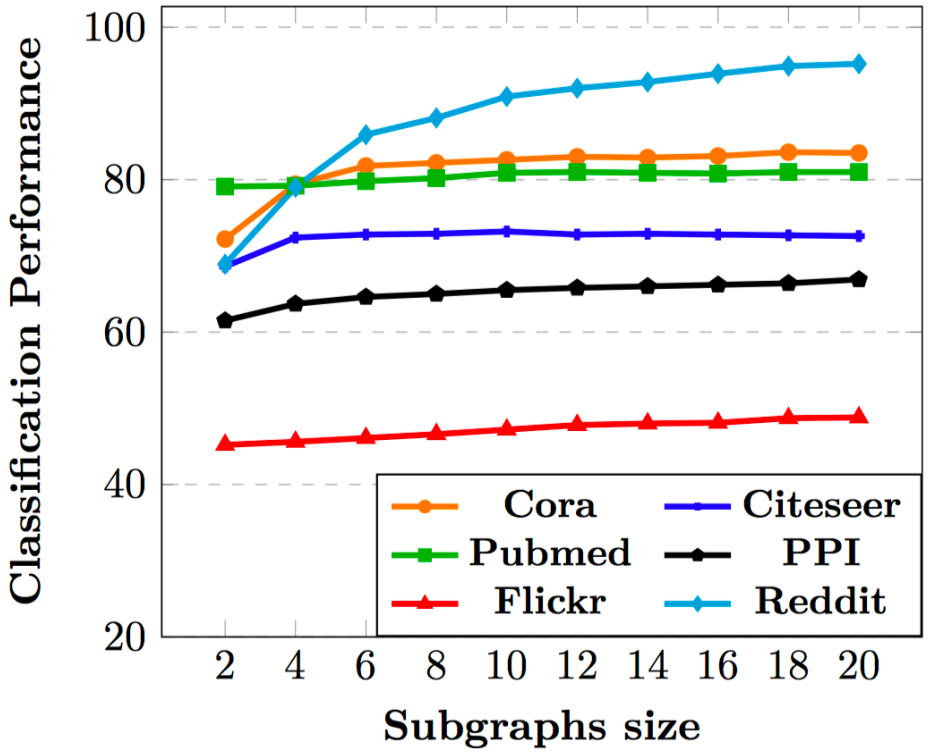


Fig. 6. Subgraph size analysis.

Better performance for larger subgraph sizes

Summary

- This paper
  - A novel self-supervised graph representation learning method via sub-graph contrast
  - Efficiency and scalability
- Future work
  - Pay more attention to capturing higher-order information in graph.

IEEE ICDM 2020

November 17-20, 2020

Sub-graph Contrast for
Scalable Self-Supervised Graph Representation Learning

Thanks for your attention!

Yizhu Jiao
yzjiao18@fudan.edu.cn