GraphSAGE

Domain	Graph
tag	Embedding generation, Inductive represenation learning
Conference / Journal	NeurIPs
Publish year	2017
정리 날짜	@2024년 1월 29일
AI summary	An inductive framework for generating node embeddings on large graphs. Unlike other node embedding methods, GraphSAGE can generate embeddings for unseen nodes by learning a function that aggregates information from a node's local neighborhood.
AI key info	GNN, Inductive Representation Learning, Node Embedding, Transductive Approaches, Inductive Capability, Graph Structure, Convolution Operator, GCN, GraphSAGE, Aggregator Function, Factorization-based Embedding Approaches, Random Walk, Graph-based Loss Function, Weight Matrices, Isomorphism, Weisfeiler-Lehman Isomorphism Test, Full Neighbor, Mean Aggregator, LSTM Aggregator, Pooling Aggregator, Clustering Coefficient, Local Graph Structure

Inductive Representation Learning on Large Graphs

Summary

Unseen node에 대해서 node embedding을 efficiently generate하는 방법 제시

개별 node에 대해 embedding 학습하지 않고, node feature 활용해 embedding 생성

Background & Motivation

Embedding of node

- Large graph의 node를 low-dim vector embedding으로 변환하는 것은 prediction과 graph analysis에서 매우 유용했음
  - Node의 neighborhood에 대한 High-dim info를 dense vector embedding으로 dimensionality reduction하는 것
- Transductive(변환하는) approaches
  - Each node에 대해 MF사용해 embedding optimize
  - Single, fixed graph에 대해서만 prediction해 generalize 불가능
- Real-world는 Inductive capability 필요
  - Unseen nodes에 대응하는 embedding이 빠르게 생성되어야
    - Unseen node를 다루고, evolving하는 graph, high-throughput에 필수적임
    - Generalization across graph 필요
  - Generalization은 새 node들을 기존의 최적화된 embedding과 align시켜야 하므로 어려움
  - Transductive를 Inductive로 만드려는 시도
    - Inductive setting하 작동 → new prediction생성 전 gradient descent해야 해 비싼 케이스
    - Graph structure을 convolution operator로 학습 → 지금까지는 fixed graph에서만
    - GCN을 inductive unsupervised learning에 적용하고 GCN을 단순 convolution을 넘어 trainable aggregtaion functions을 쓰는 것으로 generalize 해보자!
- GraphSAGE

- node와 이웃 간의 구조 + node feature의 분포를 모두 학습
- Node feature로 embedding function을 학습해 unseen node를 generalize
  - aggregator function이 node의 local neighborhood의 feature info를 모아 학습
  - inference시에, 학습된 aggreation function로 unseen node의 embedding 생성

Related work

- Factorization-based embedding approaches
  - MF기반 learning obective로 Random walk 기반으로 low-dim embedding 학습
  - 각 노드에 대해 node embedding을 학습하므로 비쌈
  - Generalize불가: objective function이 embedding의 orthogonal transformation에 불변

Methodology

GraphSAGE parameter learning →→ node embedding 만들기

Key idea: 어떻게 node의 이웃으로부터 feature info를 모을거냐

GraphSAGE parameter learning

- Graph-based Loss function

$$J_{\mathcal{G}}(\mathbf{z}_u) = -\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma(-\mathbf{z}_u^\top \mathbf{z}_{v_n})\right)$$

  - 인접 node는 비슷한 representation 갖도록 하고, 멀면 다르게 함
- output representation z_u, weight matrices 학습
  - node representation z_u가 각 노드당 unique embedding학습 아닌, local 이웃 이내의 feature로 unsupervised learning됨

GraphSAGE Embedding generation

- Algorithm
  - 매 iteration마다 node가 이웃으로부터 aggregated info 받아옴
    - 받아온 info 기존것과 vector로 concatenate하여 fully connected single layer 거침
    - 반복하며 node가 graph의 정보를 점차 모으게 됨
- Graph의 isomorphism(동형) 검증 알고리즘으로부터 착안함
  - Weisfeiler-Lehman isomorphism test
- Full neighbor을 사용하지 않고, sample fixed-size neighbor하여 계산

Aggregator architecture

Node의 neighbors은 natural ordering이 없으므로 symmetric 해야 함: 랜덤하게 잡아도 똑같게

세 가지 aggregator

- Mean aggregator
  - vector의 원소 단위로 mean 때려버림: GCN과 유사
    - GraphSAGE는 concatenate operation 한다는 차이
- LSTM aggregator
  - Larger expressive capability, but not symmetric
    - Random permutation으로 symmetric하게 하고자 함
- Pooling aggregator
  - elementwise로 max-pooling
  - different aspect of neighborhood capture잘 함
- 실험 결과 LSTM과 Pool이 비슷하게 성능 좋았음, but LSTM이 많이 느림

Theoretical analysis

- Clustering coefficient of node: 1-hop neighborhood내에 삼각형 형성 비율

- how clustered a node's local neighborhoood is 판단에 좋은 measure
- GraphSAGE embedding generation algorithm은 approximating clustering coefficient to an arbitrary degree of precision 가능함을 증명

Theorem 1. *Let* $\mathbf{x}_v \in U, \forall v \in \mathcal{V}$ *denote the feature inputs for Algorithm 1 on graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$*, where* U *is any compact subset of* $\mathbb{R}^d$*. Suppose that there exists a fixed positive constant* $C \in \mathbb{R}^+$ *such that* $\|\mathbf{x}_v - \mathbf{x}_{v'}\|_2 > C$ *for all pairs of nodes. Then we have that* $\forall \epsilon > 0$ *there exists a parameter setting* $\mathbf{\Theta}^*$ *for Algorithm 1 such that after* $K = 4$ *iterations*

$$|z_v - c_v| < \epsilon, \forall v \in \mathcal{V},$$

where $z_v \in \mathbb{R}$ *are final output values generated by Algorithm 1 and* c_v *are node clustering coefficients.*

- Local graph structure을 학습할 수 있음을 증명
  - node feature가 sampled 됐더라도

Questions