Литературный обзор по теме «Разработка алгоритмов обучения без учителя для графовых структур данных»

Андреева Полина

Введение

Графовое представление данных это естественный, понятный человеку с одной стороны, и удобный, подходящий для машинной обработки с другой, способ визуализации данных из различных областей: от генетики до банковского дела. И хотя большинство самых распространенных архитектур нейронных сетей (напримен CNN, RNN) ориентировано на векторные структуры, в последнее время появились новые подходы глубокого обучения для представления и моделирования графово-структурированных данных.

Большинство успехов в глубоком убучении достигнуто с применением одной из двух парадигм - обучения с учителем (контролируемое обучение) или обучение с подкреплением. В первом случае модель делает предположение, а результат сравнивается с истинным значением, после чего производится обратное распространение ошибки. Во втором подходе, агент взаимодействует со средой. Он делает действие, а среда подает сигналы подкрепления, поэтому такое обучение является частным случаем обучения с учителем, но учителем является среда или её модель. Но в обоих случаях пределы обучения определяются людьми, внешними надзирателями. Для истинного интеллекта необходимы более независимые стретегии обучения. В связи с чем возникает еще одна парадигма - неконтролируемое обучение, когда агент обучается с целью обучиться (например, младенцы изучают мир из любопытсва, через наблюдение, без подсказок). Более того, углубляясь в понимание работы человеческого интеллекта, можно заметить, что во взрослой жизни, люди не обучаются постоянно с нуля, а приспосабливаются к текущей ситуации, используя имеющиеся навыки и знания. По аналогии с чем в глубоком обучении придумана следующая парадигма, развивающая работу искусственного интеллекта - обучение с переносом. С точки зрения затраченных ресурсов, такой способ решения задач намного эффективней, так как не требует большого объема новых данных.

В данном обзоре сначала будет рассмотрен основной подход алгоритмов на графах. Затем - кратко перечислены основные методы и их идеи для обучения представлений графовых структур данных, основанных на нейронных сетях и нет. В следующем параграфе педставлен сравнительный анализ методов построения эмбедингов, а несколько таких методов рассматриваются более подробно. В конце описаны варианты улучшения алгоритмов обучения без учителя - масштабирование на большие графы и парадигма обучения с переносом.

Задачи на графах

Графы – структуры, состоящие из уникальных сущностей (узлов или вершин графа) и связей между ними (ребер графа), являются основной формой представления и хранения знаний и удобным иснтрументом для работы с большинством видов данных. Среди основных преимуществ графового представления данных это логическая строгость и масштабируемость на большие объемы информации. Последнее время было сделано много попыток расширить сверточные нейронные сети на графы. Проблема графовых структур данных в самом не-векторном представлении. В связи с чем, основным подходом в задачах на графах стало представление элементов графа в некотором высокоразмерном пространстве, в виде векторов, так называемых эмбедингов [2]. А далее такое представление в виде таблицы подается на вход классификатору. В таком виде задача решается как для любой другой обычной нейронной сети. Методы, использующие эмбединги, включают в себя Graph Neural Networks (GNNs) [2] и были применены к различных задачам относящимся к графам.

Задачи на графах можно разделить на следующие наиболее общие группы:

- Предсказание связи между вершинами
- Классификация вершин
- Классификация ребер
- Классификация графов

Очень много существующих методов работают в режиме обучения без учителя – то есть, без привязки к решаемой задаче, получают векторное представление узла в графе. А качество таких эмбедингов проверяют на классических задачах, чаще всего - классификация вершин.

Различия методов, использующие нейронные сети и не использующие их

Традиционные методы построения эмбедингов используют:

- Варианты сжатия, как например Лапласиан или матрицу смежности
- Спектральные техники [3]
- Нелинейное уменьшение размерности [4, 5]
- Маргинальный анализ Фишера анализирует снижение размерности множества точечных данных как эмбединги графа, используя их статистические геометрические свойства [6].

Однако, вышеперечисленные методы не могут быть применены к большим графам, так как они используют плотные матрицы. Для преодоления этого, были изобретены некоторые модификации / новые методы:

- Например, [7] использует стохастический градиентный спуск для быстрой декомпозиции матрицы смежности на собственные векторы.
- В методе HOPE [8] используется разреженный обобщенный SVD для построения эмбедингов графа из матрицы подобия, которая поддается разложению на две матрицы разреженной близости. HOPE является первым методом, использующим различные меры сходства
- VERSE [9] использует понятия схожести между вершинами и минимизируют расхождение между распределением схожести от одной вершины ко всем дргуим и схожести между конкретными вершинами
- ARCTE (Approximate Regularized Commute-Time Embedding) [10] использует алгоритмы personalized PageRank для поиска локальных сообщетсв в каждой графовой структуре пользователя.

Данные методы все еще требуют на вход полностью всю матрицу графа и рассматривают решение проблемы как один из линейных методов снижения размерности.

Нейронные графовые эмбединги:
Это в основном методы, основанные на случайных блужданиях

- DeepWalk [11] первый метод, обучающий скрытые представления в виде векторов в низко-размерном пространстве, используя ближайших соседей. Данный метод несколько раз запускает случайные блуждания фиксированной длины и создает матрицу $d-$ размерного представления вершины используя SkipGram [12].

- GraRep [13] исполльзует Singular Value Decomposition (SVD) на лог-преобразованной матрице вероятности перехода DeepWalk разных порядков а затем конкатенирует получившиеся представления.

- Struc2vec [14] перестраивает граф для отражения изоморфизма между узлами и поиска структурных сходств, а затем получает эмбединги, опираясь наDeepWalk

- LINE [15] предлагает эмбединги, которые используют более сложные понятия близости, однако, ограничиваясь только ближайшими соседями.

- Node2vec [16] вводит два гиперпараметра, чтобы регулировать генерацию случайных блужданий и, таким образом, адаптировать процесс обучения к имеющемуся графу в полууправляемом режиме.

- SEED [17] обучает автоэнкодер на случайных блужданиях типа WEAVE с последующим усреднением представлений подграфов. А для эмбеддингов вершин, такие блуждания строятся из каждой вершины.

- CraphSAGE [18] Вместо обучения отдельных эмбедингов для каждого узла, строится (обучается) функция, которая генерирует эмбединги путем выборки и агрегирования признаков из локальной окрестности узла

Сравнительный анализ

В работе [19] был рассмотрен подробный сравнительный анализ методов неконтролируемых представлений из графов. Для того, чтоб сравнивать методы, был построен объединительный фрйемворк: для каждого датасета строили вспомогательный, направленный и взвешенный контекстный граф $C(V, E')$, таким образом, что чем больше схожесть вершин, тем больше вес ребра. Для тех вершин у которых нет ребра, в контекстном графе ребро имеет вес -1. Для каждого ребра первая вершина называется исходной (или источником), вторая - контекстом. Так как каждая вершина может служить и исходной и контекстной для другой, то для каждой вершины строятся два представления Φ_i, Θ_i - исходное и контекстное соответственно. Эти Φ_i, Θ_i обучаются, минимизируя следующую функцию потерь

$$L = -\sum_{i,j} c_{i,j} \cdot f(\Phi_i, \Theta_i)$$

где f- монотонно возрастающая, c_{ij} это элемент (i, j) в матрице смежности C для контекстного графа. Вышеприведенная формула используется для методов DeepWalk [11], Node2vec [16], APP [20], LINE-2 [15]. А методы VERSE [9], LINE-1 [15], SDNE [21], unsupervised GraphSAGE [18] используют похожую функцию, но обучают только исходные представления:

$$L' = -\sum_{i,j} c_{i,j} \cdot f(\Phi_i, \Phi_i)$$

Методы HOPE [8] и NETMF [22], используют функции потерь матричной факторизации, основанные на их эквивалентности вышеуказанной функции.

Для каждого метода контекстный граф строится по-своему. В таблице 1 сводная информация по сравневаемым методам.

В ходе экспериментов учитывались несколько структурных характеристик графов для того, чтоб оценить вияние структуры графа на решение, а именно: диаметр графа, взаимность, транзитивность, спектральная разделимсоть, коэффициент локальной класстеризации. В качестве оценок выступали следущие метрики: для предсказания связей- ROC-AUC, для классификации вершин - Micro-F1 и Macro-F1. Для конструирования графов - evaluation measure. Для кластеризации графов - Normalized Mutual Information (NMI) по отношению к количеству кластеров.

Context Graph	Algorithm	Sym. C	Learnt Embeddings	Used Embeddings	Loss	Optimization
Random Walk based	DEEPWALK	✓	Φ, θ	Φ	$-\sum_{i,j} c_{i,j} \log \frac{\exp(\Phi_i \cdot \theta_j)}{\sum_{k \in V} \exp(\Phi_i \cdot \theta_k)}$	Hierarchical Softmax
Random Walk based	NODE2VEC	✓	Φ, θ	Φ	$-\sum_{i,j} c_{i,j} \log \frac{\exp(\Phi_i \cdot \theta_j)}{\sum_{k \in V} \exp(\Phi_i \cdot \theta_k)}$	Negative Sampling (NS)
PPR based	APP	✗	Φ, θ	Φ, θ	$-\sum_{i,j} c_{i,j} \log \frac{\exp(\Phi_i \cdot \theta_j)}{\sum_{k \in V} \exp(\Phi_i \cdot \theta_k)}$	NS
PPR based	VERSE	✗	Φ	Φ	$-\sum_{i,j} c_{i,j} \log \frac{\exp(\Phi_i \cdot \Phi_j)}{\sum_{k \in V} \exp(\Phi_i \cdot \Phi_k)}$	NS
Adjacency based	LINE-1	✓	Φ	Φ	$-\sum_{i,j} c_{i,j} \log \frac{1}{1+\exp(-\Phi_i \cdot \Phi_j)}$	NS
Adjacency based	LINE-2	✗	Φ, θ	Φ	$-\sum_{i,j} c_{i,j} \log \frac{\exp(\Phi_i \cdot \theta_j)}{\sum_{k \in V} \exp(\Phi_i \cdot \theta_k)}$	NS
Direct matrix	NETMF	✗	Φ, θ	Φ	$\lvert\lvert C - \Phi \cdot \theta \rvert\rvert_F^2$	Matrix Factorization (MF)
Direct matrix	HOPE	✗	Φ, θ	Φ, θ	$\lvert\lvert C - \Phi \cdot \theta \rvert\rvert_F^2$	MF
Adjacency based	SDNE	✗	Φ	Φ	see Equation (10)	Deep Autoencoders
Random Walk based	Unsupervised GRAPHSAGE	✓	Φ	Φ	$-\sum_{i,j} c_{i,j} \log \frac{\exp(\Phi_i \cdot \Phi_j)}{\sum_{k \in V} \exp(\Phi_i \cdot \Phi_k)}$	NS with Neighborhood Aggregation

Таблица 1 [19]

Сравнение проводилось решая задачи предсказания связей, классификации вершин, рекунструкции графов и кластеризации графов на следующих датасетах:

- Flickr, BlogCatalog, Youtube: социальные сети с пользователями в качестве вершин и дружбой между ними в качестве ненаправленных ребер с несколькими метками на узел.
- Twitter, Epinion: социальные сети без меток, направленный граф моделирует подписчиков и доверие между пользователями.
- DBLP-Ci, CoCit, Cora, PubMed: направленные графы, отображающие сети академического цитирования, вершины - статьи, ребра - цитирование между ними.
- DBLP- Au сеть сотрудничества авторов научных статьей DBLP Computer Science bibliography.

Основные результаты следующие:

1. Методы, учитывающие роль вершины как источника и контекста при изучении представлений, рекомендуются для прогнозирования связей в ориентированных графах.
2. Некоторые структурные свойства, такие как коэффициент кластеризации, транзитивность, взаимность и т. д., рекомендуется учитывать при выборе конкретного метода.
3. Простой классификатор, основанный на непосредственном соседстве, предлагает лучшую или сопоставимую производительность для ряда наборов данных.

4. Для задач предсказания связей на неориентированных графах методы на основе PPR (APP и VERSE) - являются наиболее эффективными методами во всех наборах данных.

5. В задачах предсказания связей, метод LINE, который непосредственно использует матрицу смежности в качестве матрицы контекста, превосходит методы случайного блуждания для неориентированных графов.

6. В задачах предсказания связей, для ориентированных графов с низкой взаимностью повторное представление контекста узла играет большую роль, и для задач предсказания направленных связей следует использовать методы кодирования и использования двух пространств внедрения для исходной и целевой контекстных представлений узлов.

7. В задачах предсказания связей более глубокие модели не имеют значительного преимущества перед мелкими.

8. Влияние структурных характеристик: для задачи предсказания связей, основнное на PPR построение показывает лучшие результаты и для ненаправленных и для направленных графов. Также, для направленных графов с низкой взаимностью, контекст основанный на Katz similarity показывает лучшие результаты. Смещенные блуждания с другой стороны имеют преимущества в предсказании связей для графов с высоким коэффициентом кластеризации, высокой транзитивностью и высокой взаимностью. Для низких коэффициентов кластеризации и транзитивности LINE-1 показывает намного более хороший результат для предсказания связей. Методы основанные на случайных блужданиях более надежные для задач классицикации вершин, а методы основанные на агрегаторах соседеней показывают лучшие результаты если есть большая схожесть между метками соседей.

9. Роль контекстного представления: контекстное представление следует явно использовать в предсказаниях направленных связей. Чем меньше взаимность (т.е. чем больше ассимитричность в роли вершин как источников и контекста) в направленных графах, тем более важно контекстное представление.

Сводная таблица 2 показывает как лучше использовать каждый отдельный метод.

Algorithm	Favourable Task	Favourable Label Properties	Favourable Graph Properties
DEEPWALK	NC	Robust for different label distributions	High Spectral separation
NODE2VEC	NC	Robust for different label distributions	High Clustering Coefficient, High Reciprocity
APP	LP	-	Directed Graphs, Low Spectral Separation
VERSE	LP	-	Undirected or directed with high reciprocity
LINE	LP, NC	Low Similarity among labels of neighboring nodes	Undirected, low clustering coefficient, low transitivity
NETMF	NC	Robust for different label distributions	Undirected
HOPE	LP	-	Directed Graphs with Low Reciprocity and low Clustering coefficient
GRAPHSAGE-GCN	LP, NC	High Label similarity among immediate neighbors	Undirected graphs with high clustering coefficient

Таблица 2 [19]

Также для некоторых конкретных задач, исходя из данного анализа, можно вывести рекомендации:

- Для ненаправленного графа, для задачи предсказания связей рекомендуются методы основанные на PPR такие как Verse и APP
- При предсказании связей в направленных графах почти всегда предпочтительно использовать APP и HOPE. Но в случае высокого значения взаимности и другие методы могут применены.
- Для направленных графов, для задачи реконструкции графов HOPE предпочитается APP
- Для задач классификации узлов, степень гомофилии должна быть подсчитана прежде чем выбирать подход. Подходы глубокого обучения, основанные на аггрегации подходят для высокой степени гомофилии, а для низкой - DeepWalk.

Подробное описание нескольких методов

Рассмотрим несколько методов неконтролируемого построения эмбедингов более подробно.

SEED

Метод:

Метод SEED [17] строит эмбединги графов в целом. Но его можно использовать и для представления эмбедингов вершин, для этого строятся случайные блуждания из каждой вершины и не усредняеются.

1. Сэмплирование.

   С помощью случайных блужданий генерируется s подграфов, используя WEAVE (random Walk with EArliest Visit timE). Число шагов блужданий k и s это гиперпараметры. Каждый граф представлен как цепочка вершин атрибутированных самым ранним визитом.

   Каждый подграф WAVE выглядит следующим образом:

   $$X = [x^{(0)}, x^{(1)}, ..., x^{(k)}]$$

   Каждый вектор-столбец представляет одну вершину в виде: атрибуты вершины x_a + наиболее раннее время визита в ходе p-го блуждания x_t:

   $$x^{(p)} = [x_a^{(p)}, x_t^{(p)}]$$

2. Обучение автоэнкодера

   Обучить MLP автоэнкодер для того чтобы он представлял каждый подграф в низкоразмерном пространстве z. В качестве тренировочного набора: s подграфов, сгенерированных на прошлом этапе. Минимизировать ошибку реконструкции $\hat{X}$ (столбец $\hat{X}$ — это x, восстановленный декодером по z):

   $$\mathbf{z} = f(X; \theta_e)$$

   $$\hat{X} = g(\mathbf{z}; \theta_d)$$

   $$L = ||X - \hat{X}||_2^2$$

   Итог: матрица z это низкоразмерное представление графа, каждый столбец z_i это низкоразмерное представление подграфа.

3. Эмбединг графа

   Для того, чтобы перейти от матрицы к векторному представлению (эмбеддингу) для графа в целом, нужно «обобщить» эмбеддинги сэмплов. Самый простой способ сделать это — усреднить

$$\hat{\mu}_G = \frac{1}{s}\sum_{i=1}^{s}\mathbf{z}_i$$

   Другой, более сложный и более общий случай — отобразить полученные представления z в некоторое новое пространство и усреднить полученные представления:

$$\hat{\mu}_G = \frac{1}{s}\sum_{i=1}^{s}\phi(\mathbf{z}_i)$$

На рисунке 1 обозначены этапы данного метода.

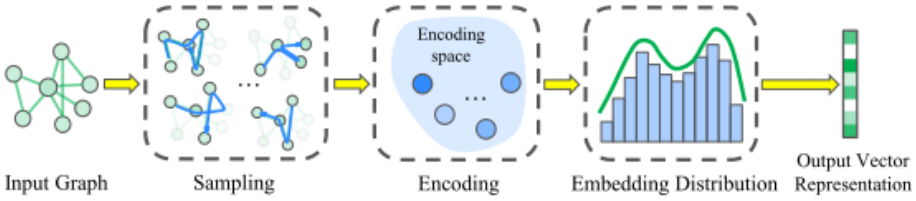

Рисунок 1 [17]

Какая задача решается: классификация и кластеризации графов.
Функция потерь: $L = ||X - \hat{X}||_2^2$
На каких наборах данных ставятся эксперименты:

- Deezer User-User Friendship Networks (Deezer): социальная сеть, признаки каждой вершины закодированы в 84 размерные multi-hot векторы.
- Mutagenic Aromatic and Heteroaromatic Nitro Compounds (MUTAG): биоинформатический датасет, два класса соединений, атрибут вершины - тип атома, атрибут ребра - тип связи.
- NCI1 - химические соединения, каждый граф относится к каждому соединению.

- PROTEINS содержит 3 класса
- COLLAB - графы, относящиеся к ученым и научным группам, метка по которой производится классификация - область исследования.
- IMDB-BINARY - кино-сеть, каждая вршина это актеры, а связь между ними существует, когда появлялись они в одном фильме.
- IMDB-MULTI расширенный предыдущий.

Какими метриками оценивается качество:

- Accuracy (ACC) для классификации.
- Normalized Mutual Information (NMI) для задач кластеризации.

  И ACC и NMI позтивные метрики (чем выше значение метрики, тем лучше результат).

С какими методами сравнивается:

- Graph Sample and Aggregate (GraphSAGE) (в неконтролируемом варианте) [18]
- Graph Matching Network (GMN) [23]
- Graph Isomorphism Network (GIN) [24]

Результат, преимущества:

SEED превосходит сравниваемые методы в задачах классификации и кластеризации. Поскольку GraphSAGE в основном фокусируется на агрегации информации о функциях от соседних узлов, для GraphSAGE может быть трудно извлечь эффективную структурную информацию в неконтролируемом режиме. В неконтролируемом режиме SEED способен дифференцировать структурные различия при более тонкой детализации и собирать богатую информацию об атрибутах, что приводит к высококачественному представлению графов с лучшей производительностью в последующих задачах. Для наборов данных NCI и PROTININS признаки узлов приносят небольшое улучшение в неконтролируемом режиме. Одной из возможных причин может быть то, что информация об особенностях узла в этих случаях имеет высокую корреляцию со структурной информацией.

GraphSAGE

Метод:

Шаги метода GraphSAGE (SAmple and aggreGatE) показаны на рисунке 2.

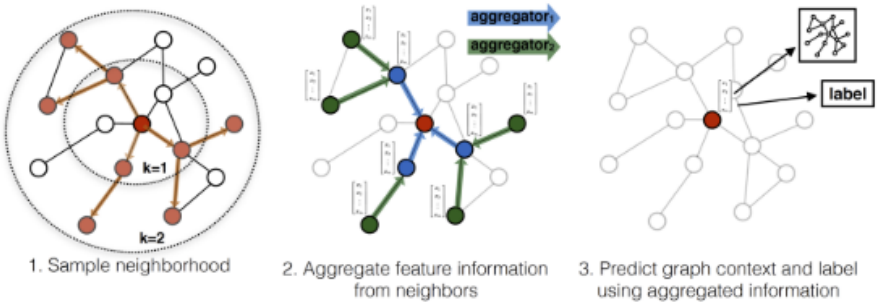

Рисунок 2 [18]

Основная идея - обучение на основе сбора информацию по признакам от соседей узла. Этапы:

1. Генерация эмбедингов (т.е., forward propagation)

   Вершина сначала собирает все представления ближайшего соседства в один вектор с помощью функции $AGGREGATE_K, \forall k \in \{1, ..K\}$(на данном этапе предполагается, что параметры всех K аггрегаторов уже обучены), затем конкатенирует с предствлением самой вершины в данный момент и этот конечный вектор подается полносвязному слою и получается вектор, который будет использоваться в качестве представления данной вершины на следующем слое.

2. Обучение параметров GrpaphSAGE

   Для случая обучения без учителя, функция потерь применяется на выходных представлениях $\mathbf{z}_u \forall u \in V$, а веса матриц $\mathbf{W}^k, \forall k \in \{1, .., K\}$ и параметры функций агрегаторов настраиваются методом градиентного спусука. Основанная на графах функция потерь поощряет близлежащие вершины иметь похожие представления, а разрозненные узлы заставляет быть оч рзличными:

$$L_G(\mathbf{z}_u) = -log(\sigma(\mathbf{z}_u^T \mathbf{z}_v)) - Q \cdot \mathsf{E}_{v_n \sim P_n(v)} log(\sigma(-\mathbf{z}_u^T z_{v_n}))$$

где P_n- распределение негативных сэмплов, $Q-$ количество негативных сэмплов.

Варианты агрегаторов:

- Mean aggregator. Функции активации на вход подается усредненное значение представлений всех вершин соседей и самой вершины.
- LSTM aggregator. В данном алгоритме LSTM адаптируется для работы с неупорядоченным множеством, применением LSTM к случайной перестановке соседей узла.
- Pooling aggregator. Выбирается максимальное значение от применения фукнции активации по всем вершинам.

Какая задача решается: классификация вершин.
Функция потерь:

$$L_G(\mathbf{z}_u) = -log(\sigma(\mathbf{z}_u^T \mathbf{z}_v)) - Q \cdot \mathsf{E}_{v_n \sim P_n(v)} log(\sigma(-\mathbf{z}_u^T z_{v_n}))$$

На каких наборах данных ставятся эксперименты:

- Web of Science citation datase: расширающиеся графы статей, основанные на цитированных данных (предсказание предмета статьи).
- Посты Reddit (предсказание категории поста).
- Датасет protein-protein взаимодействия (предсказание функци протеинов).

Какими метриками оценивается качество:
Micro-averaged F1 score
С какими методами сравнивается:

- Рандомный классификатор
- Классификатор логистической регрессии, основанныq на признаках (игнорируя структуру графа)
- Алгоритм DeepWalk [28] как типичный представитель подхода, основанного на факторизации.
- Конкатенация двух предыдущих

Результат, преимущества:

В результате экспериментов было обнаружено, что GraphSAGE значительно превосходит все базовые показатели, а обучаемые агрегаторы обеспечивают значительный выигрыш по сравнению с подходом GCN. Например, неконтролируемый вариант GraphSAGE-pool превосходит конкатенацию эмбедингов DeepWalk и необработанных функций на 13,8% по данным цитирования и 29,1% по данным Reddit, тогда как контролируемая версия обеспечивает прирост 19,7% и 37,2% соответственно. Интересно, что агрегатор на основе LSTM демонстрирует высокую производительность, несмотря на то, что он предназначен для последовательных данных, а не неупорядоченных наборов. Так же можно отметить, что производительность неконтролируемой GraphSAGE достаточно конкурентоспособна с полностью контролируемой версией, что указывает на то, что можно достичь высокой производительности без точной настройки под конкретные задачи.

Сравнение агрегаторов: LSTM and pooling агрегаторы показывают лучшие результаты. Но значительного различия между этими двумя агрегаторами нет. Однако, алгоритм, основанный на LSTM намного медленне.

VERSE

Метод:

Предполагается, что признаки, извлеченные с помощью более универсального понятия сходства, нежели чем у локального соседства, позволят достичь гибкости для решения разнообразных задач интеллектуального анализа данных на большом разнообразии графов. В связи с чем целью метода VERSE (VERtex Similarity Embeddings) является построение эмбедингов с помощью универсальных мер сходства.

Эмбединг для каждой вершины $v \in V$ должен отражать распределение сходства $sim_G : V \times V \to R$. Требуется, чтобы схожесть от любой вершины v до всех других вершин была интерпретирована как распределение $\sum_{u \in V} sim_G(v, u) = 1 \; \forall v \in V$.

Соответствующая мера схожести вершины к вершине в пространстве эмбедингов это:

$sim_E : V \times V \to R$.

Минимизируется расхождение Кульбака — Лейблера (KL) между распределениями sim_G и sim_E в пространстве эмбедингов:

$$\sum_{v \in V} KL(sim_G(v, \cdot)||sim_E(v, \cdot))$$

Распределние схожести в пространстве эмбеддингов - это нормализованное с помощью softmax преобразования скалярное произведение W_u и W_v^T (скалярное произведение векторов максимально, когда они совпадают) :

$$sim_E(v, \cdot) = \frac{exp(W_v W^T)}{\sum_{i=1}^{n} exp(W_v \cdot W_i)}$$

Тогда функция потерь (исходя из определения расхождения Кульбака — Лейблера) преобразовывается и становится эквиалентной минимизации кросс - энтропии:

$$L = -\sum_{v \in V} sim_G(v, \cdot) log(sim_E(v, \cdot))$$

Для оптимизации используется метод Noise Contrastive Estimation (NCE). NCE обучает бинарный классификатор, для различения выборки узлов, полученные из эмпирического распределения подобия sim_G, и выборки, генерируемой распределением шума Q. Вспомогательная случайная переменная D равна 1, если узел взят из эмпирического распределения и 0 для выборки, взятой из распределения шума. Имея вершины u из некоторого распределения $\mathcal{P}$ и v из распределения $sim_G(u, \cdot)$ строится $s \ll n$ вершин $\widetilde{v}$ из $Q(u)$ и испольузтся логистическая регрессия для минимизации отриацательного логарифмическое правдоподобия:

$$L_{NCE} = \sum_{u \sim \mathcal{P}, v \sim sim_G(u, \cdot)} [log P_{r_W}(D = 1 | sim_E(u, v)) + \\ + s\mathbf{E}_{\widetilde{v} \sim Q(u)} log P_{r_W}(D = 0 | sim_E(u, \widetilde{v}))]$$

где $P_{r_W}-$ вычисленная сигмоида скалярного произведения W_u и W_v.

Вид sim_G зависит от выбранной меры сходства:

- **Personalized PageRank**. Учитывая начальное распределение узлов s, фактор демфирования α и нормализованную матрица смежности A, вектор Personalized PageRank π_s определяется рекурсивным уравнением:

  $$\pi_s = \alpha \cdot s + (1 - \alpha)\pi_s A$$

  Стационарное распределение случайных блужданий с вероятностью перезапуска α сходится к PPR.

- **SimRank** является мерой структурной взаимосвязи между двумя узлами, основанной на предположении, что два узла похожи, если они связаны с другими подобными узлами; SimRank определяется рекурсивно следующим образом:

$$sim_G^{SR}(u,v) = \frac{C}{|I(u)||I(v)|} \sum_{i=1}^{|I(u)|} \sum_{j=1}^{|I(v)|} sim_G^{SR}(I_i(u), I_j(v))$$

  где $I(v)-$ количество соседей по выходящим из вершины v ребрам. А число $C \in [0,1]$ геометрически обесценивает важность более дальнего узла

- **Adjacency similarity** нормализованная матрица смежности является непосредственной мерой подобия. Это сходство учитывает только непосредственные соседи вершины. Имея значение $Out(u)$, равное количеству выходящих из вершины u ребер, формальное значение схожести выражается слдующим образом:

$$sim_G^{ADJ}(u,v) = \begin{cases} \frac{1}{Out(u)} & \text{if } (u,v) \in E \\ 0 & \text{otherwise} \end{cases}$$

Итак, общий алгоритм:

1. Имея функции подобия sim_G и размерность d пространства эмбедингов, инициализируется матрица эмбедингов W из нормального распределения с мат ожиданием равным 0 и дисперсией $\frac{1}{d}$.

2. Оптимизируется функция потерь методом градиентного спуска, используя алгоритм NCE: выбирается вершина из позитивного распеделения $\mathcal{P}$, выбираеся sim_G и строится s неагтивных примеров.

3. Выбирается $\mathcal{P}$, Q таким образом, чтобы оин были равномерно-распределенными $\mathcal{U}(1,n)$

Какая задача решается: задачи предсказания связей, классификации вершин, кластеризации вершин, реконструкции графа.

Функция потерь:

$$L_{NCE} = \sum_{u \sim \mathcal{P}, v \sim sim_G(u,\cdot)} [logP_{r_W}(D=1|sim_E(u,v)) + \\ + s\mathbf{E}_{\widetilde{v} \sim Q(u)} logP_{r_W}(D=0|sim_E(u,\widetilde{v}))]$$

На каких наборах данных ставятся эксперименты:

- BlogCatalog - сеть социальных взаимодействий среди блогеров на сайте BlogCatalog. Метки вершин означают темы, на которые пишет автор.
- CoCit - сеть статец, цитируемых другими статьями. Метки означают конференции, на которых статьи были опубликованы.
- CoAuthor - соавторская сеть среди авторов.
- VK - извлечены два состояния сети в ноябре 2016 и мае 2017. Использовался гендер для классификации пользователей и срана для кластеризации пользователей.
- YouTube - метки означаю группы зрителей по жанрам видео.
- Orkut - сеть социальных взаимодействий между пользователями сайта Orkut. Метки означают сообщества пользователя. Для данного метода были извлечены 50 наибольших сообщества.

Какими метриками оценивается качество:

- Предсказание связей: Adamic-Adar, Preferential attachment, Katz и Jaccard coefficient.
- Классификация вершин: Macro-F1
- Кластеризация вершин: Normalized Mutual Information (NMI)
- Реконструкция графа: относительная точность, померенная как количестов верных вершин в соседстве вершины в пространстве эмбедингов.

С какими методами сравнивается:
HOPE [8], GraRep [13], LINE [15], DeepWalk [11], Node2vec [16], Logistic regression, Louvain community detection.

Результат, преимущества:
Предсказание связей: Из проводимых экспериментов видно, что VERSE с произведением векторов Адамара лучшее представление ребер. VERSE стабильно превосходит другие методы в протестированных наборах данных. Кроме того, контролируемый гиперпараметром вариант hsVERSE обгоняет Node2vec по всем наборам данных.

Классификация вершин: VERSE дает прогнозы, сопоставимые или превосходящие прогнозы других методов, в то время как он масштабируется для больших сетей, таких как Orkut. LINE превосходит VERSE только в датасете VK, где пол пользователей лучше определяется с помощью прямого соседства. Вариант с гиперпараметром, hs-VERSE такой же по качеству как и Node2vec на датасетаз CoCit и VK; В крупных наборах данных YouTube и Orkut метод hsVERSE превосходит неконтролируемые альтернативы, а Node2vec исполбузет много памяти.

Кластеризация вершин: на датасете CoAuthor метод VERSE имеет сравнимые с DeepWalk результаты; А на датасете VK, VERSE превосходит все остальные методы.

Реконструкция графа: лучшим методом оказался вариант является hsVERSE, который достигает полученного результата, когда создается экземпляр сходства смежности.

ARCTE

Метод:

В классификации пользователей социальных сетей возникает проблема, когда пользователи информацию о себе выкладывают довольно запутанную, или когда ифнормаця уже устарела, или слишком мала (твиттер). А многие пользователи вообще ничего не выкладывают о себе. Идея использующаяся в данном методе для преодоления такой проблемы - плотные сообщества, вознкающие от того, что люди люди подписываются и дружат с теми, с кем интересы совпадают.

Данный метод решает проблему полуконтролируемой классификации пользователей по множественным меткам в два этапа перед использованием непосредственно самого классификатора:

1. Предлагаемый алгоритм Approximate Regularized Commute-Time Embedding (ARCTE) обнаруживает сообщества, сконцентрированные вокуг пользователей, учитывая пользователей, не связанных напрямую, но схожимх по интересам. иИсправленный алгоритм personalized PageRank используются для поиска локальностей в каждой графовой структуре пользователя. А затем пользователи проецируются в скрытое пространство с помощью этих сообещств.

2. Затем работает контролируемое взвешивание признаков сообществ для того, чтобы повысить важность сообществ, с высоким уровнем прогнозирования.

Основные этапы алгоритма изображены на рисунке 3.

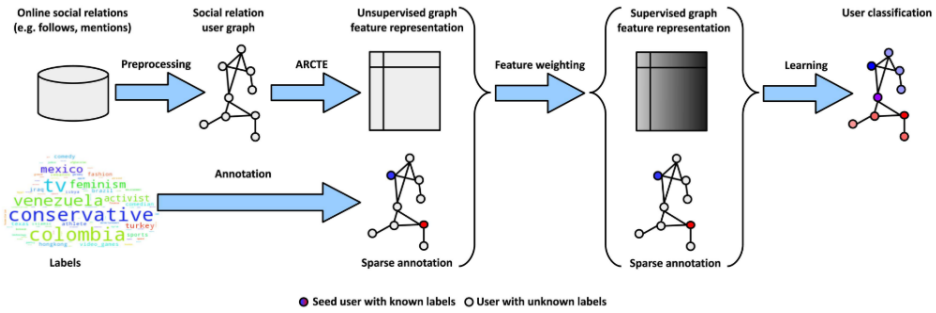

Рисунок 3 [10]

Описание этапов:

1. Этап неконтролируемого построения эмбедингов подразумевает формирование сообществ вокруг вершины. Данные сообщества содержат саму веришу, все ближайшие соседи и другие веришны, не связанные напрямую с данной, но тем не менее схожие с ней. Для поиска последних - строится вектор k, где каждое значение соответствует близости от каждой вершины к данной, а затем усекается, беря во внимание только самые близкие. Подсчет вектора k начинается с инициализации нулем и происходит случайными блужданиями с перезапуском, итеративно обновляясь:

$$k_{pr}^{(t+1)} = k_{pr}^{(t)} + r^{(t)} I_u$$

$$r^{(t)} = \rho e_v - k_{pr}^{(t)}(I - (1-\rho)W)$$

где $r^{(t)}$ называется вектором остаточных вероятностей на шаге времени t. $\rho-$ вероятность перезапуска случайного блуждания, I_u- нулевая матрица с единственным ненулевым элементом на диагонали на $u-$ ой строке. e_v- это распределение с вероятнстью сконцентрированной на позиции v. $W = D^{-1}A-$ матрица вероятностей перехода марковской цепи. $D-$ матрица степеней, $A-$ матрица смежности.

Другой вариант подсчета вектора близостей - через вектор кумулятивных разностей PageRank:

$$ck_{\delta pr}^{(t+1)} = ck_{\delta pr}^{(t)} + k_{pr}^{(t+1)} - k_{pr}^{(t)}$$

или:

$$ck_{\delta pr}^{(t+1)} = ck_{\delta pr}^{(t)} + (1-\rho)r^{(t-1)}W_u$$

Вершина u входит в вектор k, если $r(u)/d(u) > \varepsilon$, где $\varepsilon-$ выбранный заранее порог.

2. Контролируемая часть данного метода подращумеваем взвешивание признаков путем умножения каждого признака j на свой вес w_j. Вес w_j считается по формуле:

$$w_j = ivf(j) \times \chi^2(j)$$

где первый множитель учитывает то, что большие сообщества подразумевают более слабое участие вершин внутри сообщества и считается обраным часотным взвешиванием вершин внутри одного сообщества. А второй множитель отвечает за то, чтобы сообщества с несколькими метками имели большее влияние на предсказание и считается через $\chi^2(j)$ статистику.

Какая задача решается: классификации пользователей.
На каких наборах данных ставятся эксперименты:

- SNOW2014 Graph - упоминания и ретвиты в твиттере.
- ASU-Flickr (ASU-FR) - вершины графа представляют собой пользователей платформы для размещения изображений и видео. Пользователи могут подписываться друг на друга и подписываться на группы по интересам.
- ASU-YouTube (ASU-YT) - вершины предсталяют собой пользователей YouTibe. Пользователи также могут подписываться дргу на друга и группы по интересам.
- Insight Resource Multiview (IRMV) -IRMV-PoliticsUK. - неориентированный граф с политическими интересами пользователя, извлечены ретвиты и упоминания.

Какими метриками оценивается качество:
Micro и macro-averages метрики F1.
С какими методами сравнивается:

- Laplacian Eigenmaps (LapEig) [26]
- Replicator Eigenmaps (RepEig) [27]

- Random Walk Modularity Maximization (RWModMax) [25]
- DeepWalk [11]
- LINE [15]
- Louvain [28]
- Edge Clustering (EdgeCluster) [29]
- Multiple Resolution Overlapping Communities (MROC) [30]
- Cluster Affiliation Model For Big Networks (BigClam) [31]
- Order Statistics Local Optimization Method (OSLOM) [31]
- Base Communities (BaseComm)

Результат, преимущества:

Во-первых, было показано, что чем меньше параметры ρ, ε, тем точнее результат но тем и больше время работы.

Во-вторых, ARCTE превосходит все методы на всех датасетах, кроме ASU-Flickr, на котором результаты все еще сравними с победителями (MROC, BaseComm) в рамках $F1-macro$. Если же сравнивать спользуя $F1-micro$, то ARCTE показывает лучшие результаты против асболютно всех методов.

Некоторые выводы: методы ARCTE, MROC и BaseComm были значительно эффективнее, чем Edge-Cluster, BigClam и OSLOM. Возможно, причина в том, что первые три метода фокусируются на структуре с высоким разрешением в графе, тогда как последние три предназначены для создания мезоскопического представления графа. Более высокая производительность ARCTE по сравнению со всеми низкоранговыми матричными методами представления также может быть объяснена улучшением, вызванным контролируемым взвешиванием сообщества.

Машстабирование

В сверочных нейронных сетях информцация для выходного узла собирается из соседних узлов на предыдущем слое. А для них, в свою очередь, собирается информации для их соседей с предпредыдущего слоя. Таким образом происходит т.н. "neighbor explosion"и чем глубже слои, тем медленнее и сложнее обучение. Для преодоления этой проблемы существует несколько методов, но как правило, они все равно сталкиваются с проблемой масштабирования на большие графы. Рассмотрим возможные варианты решения проблемы масштабирования:

1. Для масштабирования сверточных графовых сетей на большие графы, были предложены методы выборки слоев [18, 34–37] для эффективного обучения мини-батчей. Все они выполняют три меташага: 1. Строят полный GCN на полном графе. 2. Выбираются узлы или ребер каждого слоя для формирования мини-пакетов. 3. Распространите вперед и назад среди отобранных GCN. Шаги (2) и (3) выполняются итеративно для обновления весов посредством стохастического градиентного спуска.

2. Алгоритм выборки слоев GraphSAGE [18] выполняет равномерную выборку узлов на соседних предыдущего уровнях. Он применяет предопределенное ограничение на размер выборки, чтобы ограничить сложность вычисления мини-батча.

3. В статье [34] авторы расширют сэмплер слоев [18], вводя показатель важности каждого соседа. Предполагается, что алгоритм приводит к меньшей потере информации из-за взвешенной агрегации.

4. S-GCN [35] дополнительно ограничивает размер окрестности, требуя только два узла на предыдущем уровне. Идея состоит в том, чтобы использовать исторические активации в предыдущем слое, чтобы избежать избыточной переоценки.

5. FastGCN [33] выполняет выборку с другой точки зрения: вместо отслеживания межуровневых соединений выборка вершин выполняется независимо для каждого уровня. Выбираются важные вершины для уменьшения дисперсии, что приводит к постоянному размеру выборки во всех слоях. Тем не менее, мини-батчи потенциально становятся слишком разреженными для достижения высокой точности.

6. FastGCN улучшен в статье [37] за счет дополнительной выборки нейронной сети. Это обеспечивает высокую точность, поскольку выборка обусловлена выбранными в следующем слое узлами. Значительные накладные расходы могут возникнуть из-за дорогостоящего алгоритма выборки и дополнительных параметров выборки, которые необходимо изучить.

7. Вместо слоев выборки, работы [38,39] строят мини-батчи из подграфов. Кроме того, они представляют методы для масштабирования такого обучения на многоядерных платформах с общей памятью. Обе работы не учитывают смещение из-за неравной вероятности появления каждого узла / ребра в мини-батче. [38] предлагает специальный алгоритм выборки графа для обеспечения связи между вершинами мини-батчей. ClusterGCN [39] предлагает обучение мини-батчей на основе кластеризации графов. Во время предварительной обработки тренировочный граф разбивается на плотно связанные кластеры. Во время обучения кластеры выбираются случайным образом для формирования мини-батчей, а соединения внутри кластера остаются неизменными.

8. Другое направление исследований направлено на улучшение возможностей модели. Обращая внимание на графы, архитектуры [40–42] лучше учитывают признаки соседей путем динамической регулировки веса ребер.

9. В работе [43] объединен PageRank с GCN, чтобы обеспечить эффективное распространение информации на много "скачков"от вершины.

10. Для разработки более глубоких моделей «skip-connection» позаимствовано у CNN [37,44] в контексте GCN. В частности, JK-net [45] демонстрирует значительное улучшение точности для GCN с более чем двумя слоями. Однако, JK-net [45] следует той же стратегии выборки, что и GraphSAGE ([18]). Таким образом, его стоимость обучения все еще высока из-за "neighbor explosion".

Метод GraphSAINT [46], который превосходит многие современные методы (vanilla GCN [58], GraphSAGE [18], FastGCN [33], S-GCN [35], AS-GCN [37], ClusterGCN [39], использует следующие идеи:

1. Вместо того, чтобы строить сначала GCN на полном обучающем графе, а затем сэмплировать сквозь все слои, сначала сэмплирует обучающий граф и только потом строит полный GCN на подграфе.

2. Интуитивно кажется, что узлы более высокго влияния друг на друга, должны иметь бОльшую вероятность оказаться в одном подграфе. Это позволяет выбранным узлам поддерживать друг друга, не выходя за пределы мини-пакета. Но к сожалению, такая стртегия ведет к неидентичной вероятности выборки узлов что ведет к смещению в оценке мини-батча. Для решения этой проблемы для данного метода разработана техника нормалиации, такая что обучение признаков не дает предпочтение узлам более часто выбираемым.

Перед началом обучения, граф проходит легкую предобработку с помощью сэмплера SAMPLE. Предварительная обработка оценивает вероятность того, что узел $v \in V$ и ребро $e \in E$ будут отобраны с помощью SAMPLE. Такая вероятность позже используется для нормализации агрегации соседей подграфа и функции потерь мини-батча. Обучение происходит интеративным обновлением веса с помощью SGD.Каждая итерация начинается с независимо отобранного подграфа G_s (количество узлов в подграфе намного меньше чем в целом графе). Затем строится полный GCN на подграфе G_s для генерации эмбединга и подсчета функции потерь для каждой вершины из подграфа.

Идеальный SAMPLE отвечает двум требованиям: похожие узлы должны быть в одном подграфе и ребра должны иметь ничтожно малую вероятность попасть в подграф. Для первого требования идеальный SAMPLE будет рассматривать совместную информацию о связях между узлами так же как и сами атрибуты. Хотя итоговый алгоритм может быть высоко затратный по вычислениям. Для простоты «влияние» определяется с точки зрения связности графов и сэмплеры строятся на основе топологии. Второе требование приводит к лучшему обобщению, так как позволяет нейронной сети исследовать все пространство признаков и меток.

Обучение с переносом

Бывают случаи, когда размеченных данных очень мало, тогда используется парадигма "обучения с переносом". Она позволяет натренировать модель распознавать «базовые» признаки неразмеченных данных, а затем, присоединяя несколько последних слоёв и заменяя функцию потерь непосредственно для решения задачи обучения с учителем, модели подаются имеющиеся размеченные данные. Рассмотрим несколько методов обучения с переносом.

Обучение с переносом внутренней геометрии на графово - структурированных данных [53]

Какую проблему решает: Методы CNN и RNN ориентированы на матричные структуры данных. А данный подход улучшает глубокое обучение данных с графовой структурой с помощью обучения с переносом. Передавая внутреннюю геометрическую информацию, полученную в исходной области, этот метод позволяет построить модель для новой, но связанной задачи в целевой области без сбора новых данных и без обучения новой модели с нуля.

Основная идея: Обучить сверточную нейронную сеть на исходных данных, а затем последний слой переобучить на целевой задаче для перенастройки весов.

Метод по шагам:

- Строится граф из входных данных. Есть два варианта построения графов: co-occurrence graph estimation (CoGE) [54] и supervised graph estimation (SGE) [55]. CoGE напрямую определяет близость элементов данных основываясь на частоте смежности. SGE автоматически запоминает признаки сходства элементов.

- Строится лапласиан полученного графа. Определяется операция свертки.

- Применяется сверточная нейронная сеть к графам. Модель SCNN (спектральная сверточная нейронная сеть) обучается на информации, полученной на предыдущем шаге. Обучение определят веса каждого слоя путем минимизации функции потерь, специфичной для каждой задачи.

- Выделение признаков для переноса.

- Обучение переносом в спектральном домене. Строится модель для целевой задачи копированием сверточных и субдискретизирующих слоев, которые содержат признаки обученные для исходного домена и исходной задачи. А затем, последний, полносвязный слой модели переобучается на новом, целевом домене для настройки весов.

Результаты: передача знаний между доменами более эффективна, когда представление графа в исходном и целевом домене более схожи.

Будущее развитие: применить подход к различным наборам данных в разных областях.

SR2LR [47]

Задача: предсказание связей между вершинами.

Основная идея:

Данные в исходном и целевом доменах описываются марковскими логическими сетями. Предикаты описывают отношения между сущностями. Предполагается, что хорошая модель исходного домена содержит дизъюнкты двух типов - короткодействующие (short-range clauses), которые касаются свойств единственной сущности и дальнодействующие (long-range clauses), относящийся к свойствам нескольких сущностей. Возможные отображения дизъюнктов первого типа в целевой домен могут быть вычислены на доступных целевых данных непосредственно. Поэтому метод основан на том, чтобы использовать короткодействующие дизъюнкты с целью найти отображения между отношениями в двух доменах, а затем эти отображения использовать для передачи дальнодействующих дизъюнктов. Отсюда и название метода SR2LR - short-range to long-range.

Сравнение с другими методами:

- Одна из областей, где обучение с перносом особенно эффективно работает - это статичтическое относительное обучение (statistical relational learning SRL). В SRL обучаются вероятностные модели на основе сложных реляционных данных [48], [49]. Примеры обучения (они называются мега-примеры) обычно огромные, содержат сотни сущностей и связей различной длины. Из-за этого, как правило, алгоритмы SRL имеют длительное время обучения и часто чувствительны к локальным максимумам.

- Для решения проблем точности и скорости обучения и эффективен подход обучения с переносом [50] [51]. Но эти подходы требуют большого количества данных, как минимум один целевой мега-пример, состоящий из большого количества сущностей. А метод SR2LR справляется с проблемой и решает задачу в случае нембольшого количества сущностей, в экстремальном варианте - всего с одним.

- Structure-mapping engine (SME) [52] тоже переносит исходные знания в целевой домен. Метод производит отображение предикатов на основе синтаксического структурного критерия, называемого системностью, и не учитывает точность результирующих выводов в

целевых данных. В отличие от SR2LR, который оценивает отображения на основе того, производят ли они эмпирически адекватные дизъюнкты в целевой области.

Будущее развитие: новые способы отображения исходных знаний, такие как, отображение предикатов разной арности друг в друга, отображение конъюкций двух предикатов в исходном домене в один предикат в целевом домене и наоборот.

AdaGCN [56]

Алгоритмы доменной адаптации (domain adaptation) справляются с задачами обучения с переносом в ситуациях, когда распределение вероятностей в исходном и целевом домене различаются.

Задача: Классификация узлов. Использование частично помеченной исходной сети для того, чтоб облегчить классификацию узлов в другой, полностью не помеченной сети.

Основная идея: метод состоит из двух компнент - компонента полуконтролируемого обучения (найти границу классов с помощью сверточных сетей) и компонента состязательной доменной адаптации (найти представления вершин, инвариантные относительно доменов с помощью состязательного обучения)

1 компонента: представляем k - ый сверточный слой в исправленном виде относительно обычного GCN:

$$\mathbf{H}_g^k = \sigma\left(\hat{\mathbf{A}}^{n_I}\mathbf{H}_g^{k-1}\mathbf{W}_g^k\right),$$

где $\hat{A}$ – сумма матрицы смежности графа и диагональной матрицы степеней вершин графа, H_g^k – k-ый сверточный слой, W_g^k– матрица весов, n_I – параметр сглаживания. С помощью установления подходящего данного параметра, можно контролировать силу сглаживания свертки графа для облегчения переноса знаний и классификации, избегая переобучения.

Далее, после представления вершин исходного и целевого домена в виде эмбедингов, эти данные передаются в классификатор для прогнозирования меток. Классификатор может быть и однослойными классификатором логистической регрессии и многослойным перцептроном.

2 компонента: доменная адаптация моделирутеся как игра двух игроков, так же как и в GAN. Сеть для генерации эмбедингов вершин играет роль генератора, а играющий роль дискриминатора доменный

критик оптимизируется для задача классификаций эмбедингов вершин на два класса: вершины из исходной сети и вершины из целевой.

После состязательного обучения, могут быть получены инвариантные представления сети и информация о классах может быть передана из исходного в целевой домен. Итого, для обучения инвариантных узлов решается следующая minimax проблема:

$$\min_{\theta_g} \max_{\theta_d} \{\mathcal{L}_d - \gamma \mathcal{L}_{grad}\},$$

где θ_g, θ_d параметры обучения модели , $\mathcal{L}_d$ функция потерь доменного критика по отношению к θ_d, а $\mathcal{L}_{grad}$ это штрафной градиент для параметра θ_d, γ это штрафной коэффициент, который должен быть устанолвен нулем для генератора.

Сравнение с другими методами:

- Существующие эмбединговые методы [11], [15], [16] сначала изучают компактные представления узлов для сохранения информации о структуре сети, а затем обучают классификатор с помощью изученных представлений для классификации узлов. Такие методы не позволяют решить следующие проблемы:

  - значительное расхождение между исходным и целевым доменом и отсутсвие общих свойств.

  - отсутвие межсетевых ребер для распространения знаний из исходной в целевую сеть.

  - если в исходном домене помечена только малая часть вершин.

- Методы полуконтролируемого обучения основанные на графах [57], [58] показали высокую эффективность для классификации вершин на единственной сети даже в случае когда помеченных вершин не очень много. Методы GCN [58], GraphSAGE [18], GAT [40] показали высокую производительность при классификации вершин, посредством интеграции топологии графов, свойств вершин и наблюдаемых меток вршин в сквозную среду обучения. Но эти методы построены для обучения задания в единственном домене и имеют проблемы в обобщении на другие домены, которые могут иметь существенно отличающийся набор атрибутов.

- Несколько методов [59], [60] предложены для использования оношений между сетями для повышения эффективности. Они обучают эмбединги для нескольких сетей одновременно, но они сильно

зависят от существования межсетевых соединений. Как уже было сказано выше, они не справятся с ситуацией отсутвия этих связей.

- Обычный метод доменной адаптации использует знания исходных доменов для того, чтоб решить такую же задачу в целевом домене [61], [62]. Практически все методы в основном ориентированы на векторные данные, такие как картинки и текст, а не графовые структуры данных.

Будущее развитие: Исследование передачи знаний из множества исходных сетей в одну целевую сеть и изучение условной адаптации состязательного домена для лучшего согласования распределения мультимодальных данных.

AS-MAML [63]

Вариант переноса с обучением, когда требуемый результат получают основываясь только на нескольких/одном тренировочной примере соответственно. Этот тип крайне важен в реальном мире, когда невозможно для каждой задачи собрать большое количество помеченных тренировочных данных для каждого класса и в случаях, когда классы постоянно добавляются.

Задача: Классификация графов.

Дан граф $G = (G_1, y_1), (G_2, y_2), ..., (G_n, y_n)$, где $G_i = (V_i, E_i, X_i)$. В зависимости от метки y_i он разбивается на тренировочное и тествое множество $\{(G^{train}, y^{train})\}$ и $\{(G^{test}, y^{test})\}$. При этом y^{train}, y^{test} не должны иметь общих классов. На этапе обучения на каждом шаге выбирается задача $\mathcal{T}$ и каждая задача содержит поддерживающие данные $\mathcal{D}_{sup} = \{(G_i^{train}, \mathbf{y}_i^{train})\}_{i=1}^{s}$ и данные запроса $\mathcal{D}_{que} = \{(G_i^{text}, \mathbf{y}_i^{test})\}_{i=1}^{q}$. Имея помеченные данные в $\mathcal{D}_{sup}$, необходимо предсказать метки в $\mathcal{D}_{que}$.

Основная идея: Передача знаний от обученной задачи классификации графов к новым задачам. Этот метод является объединением методов GNN [2] в качестве основы и MAML [64] для автоматического подбора оптимальных гиперпараметров.

Этап 1: применение GNN. Цель GNN это представить граф в виде эмбедингов. Это делается в предположении теоремы Банаха о неподвижной точке, так как каждый эмбединг зависит от эмбедингов свох соседей. Существуют различные методы GNN, в зависимсоти от того в каком виде представлена функция для построения эмбединга. В методе AS-MAML используется агрегатор среднего значения, как в методе GraphSAGE [**?**]:

$$\mathbf{h}_v^l = \sigma\left(\mathbf{W} \cdot mean(\{\mathbf{h}_v^{l-1}\} \cup \{\mathbf{h}_u^{l-1}, \forall u \in \mathcal{N}(v))\}\right),$$

где $\mathbf{h}_v^l$ - это l - ый слой представления вершины v, σ - сигмоидная функция, $\mathbf{W}$ параметры агрегатора, $\mathcal{N}(v)$ содержит соседей вершины v.

А затем использовать обычную сверточную нейронную сеть.

Этап 2: применение MAML для реализации быстрой адапатации. [65] предложил основанный на обучении с подкреплением (Reinforcement Learning - RL) шаговый контроллер для проведения метаобучения для предсказания связей между вершинами. Но выбраннная функция потерь является слишком неоптимальной, чтобы рассматриваться как вознаграждение за преодоление переобучения. В связи с этим и разработан новый шаговый контроллер для ускорения обучения и преодоления переобучения. Он также управляется RL, но оптимальный шаг адаптации выбирается исходя из показателя ANI (Average Node Information) и функции потерь в качестве входных данных и точности классификации в качестве вознаграждения:

Сравнение с другими методами:

- Существует несколько методов классификации вершин [66], [67], [68], [69], основанные на GNN и ускоряющие передачу знаний и методы предсказания связей [65] между вершинами. Но они не могут быть расширены на задачи классификации графов.

- В [70] предложена классификация графов, основанная на спектральных изерениях, но этот метод предполагат, что тестовые классы принадлжат тому же множуству классов, что и тренировочные данные.

Будущее развитие: Данный метод может быть расширен на другие задачи классификации графов, как например распознавание действий по скелету и анализ подграфов для социальных сетей.

Выводы

Было рассмотрено множество различных методов, использующих разные подходы к обучению представления графовых структур данных. Основным подходом остается построение эмбедингов на основе случайных блужданий, хотя и другие методы показывают улучшения. При решении задач и выборе метода, необходимо учитывать структурные свойства графов, размер графа, тип задач.

Чаще всего, для задач на грфах используются датасеты социальных сетей или сетей цитирования, и решают задачу классификации вершин. А для оценки качества методов считают $micro-, macro-F1$ для задач классификации и NMI для задач кластеризации.

Основное будущее развитие всех методов это масштабирование на большие графы, а также улучшения методов, направленные на улучшение качества и скорости работы алгоритмов.

Список литературы

[1] S. Pan, Q. Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(10):1345–1359.

[2] Z. Wu, S. Pan, F. Chen, G. Long,C. Zhang, S. Yu Philip. 2019. A comprehensive survey on graph neural networks. arXiv:1901.00596.

[3] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In NIPS. 585–591

[4] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. science 290, 5500 (2000), 2323–2326.

[5] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. science 290, 5500 (2000), 2319– 2323

[6] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. 2007. Graph embedding and extensions: A general framework for dimensionality reduction. TPAMI 29, 1 (2007)

[7] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. 2013. Distributed large-scale natural graph factorization. In WWW. ACM, 37–48.

[8] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In KDD. 1105–1114

[9] A. Tsitsulin, D. Mottin, P. Karras, and E. Müller, "VERSE: Versatile graph embeddings from similarity measures," Web Conf. 2018 - Proc. World Wide Web Conf. WWW 2018, pp. 539–548, 2018, doi: 10.1145/3178876.3186120.

[10] G. Rizos, S. Papadopoulos, and Y. Kompatsiaris, Multilabel user classification using the community structure of online networks, vol. 12, no. 3. 2017.

[11] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in KDD, 2014, pp. 701–710.

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations ofWords and Phrases and their Composition- ality. In NIPS. 3111–3119

[13] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning Graph Repre- sentations with Global Structural Information. In CIKM. 891–900

[14] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In KDD. ACM, 385–394

[15] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: large-scale information network embedding," in WWW, 2015, pp. 1067–1077.

[16] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in KDD, 2016, pp. 855–864.

[17] L. Wang et al., "INDUCTIVE AND UNSUPERVISED REPRESENTATION LEARNING ON GRAPH STRUCTURED OBJECTS," 2016.

[18] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," Adv. Neural Inf. Process. Syst., vol. 2017-Decem, no. Nips, pp. 1025–1035, 2017.

[19] M. Khosla, V. Setty, and A. Anand, "A Comparative Study for Unsupervised Network Representation Learning," IEEE Trans. Knowl. Data Eng., pp. 1–1, 2019, doi: 10.1109/tkde.2019.2951398.

[20] C. Zhou, Y. Liu, X. Liu, Z. Liu, and J. Gao. Scalable graph embedding for asymmetric proximity. In AAAI, pages 2942–2948, 2017.

[21] D.Wang, P. Cui, andW. Zhu. Structural deep network embedding. In SIGKDD, pages 1225–1234. ACM, 2016.

[22] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In WSDM, pages 459–467, 2018.

[23] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching net- works for learning the similarity of graph structured objects. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), Proceedings ofInternational Conference on Machine Learning, volume 97, pp. 3835–3845, 09–15 Jun 2019

[24] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In Proceedings ofInternational Conference on Learning Representations, 2019

[25] Devooght R, Mantrach A, Kivima¨ki I, Bersini H, Jaimes A, Saerens M. Random walks based modularity: application to semi-supervised learning. In: Proceedings of the 23rd international conference on World wide web. Seoul, Republic of Korea: International World Wide Web Conferences Steering Committee; 2014. p. 213–224.

[26] Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation. 2003; 15(6):1373–1396.

[27] Smith LM, Lerman K, Garcia-Cardona C, Percus AG, Ghosh R. Spectral clustering with epidemic diffu- sion. Physical Review E. 2013; 88(4):042813.

[28] Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communitie

[29] Tang L, Liu H. Scalable learning of collective behavior based on sparse social dimensions. In: Proceed- ings of the 18th ACM conference on Information and knowledge management. Hong Kong, China: ACM; 2009. p. 1107–1116. 21.

[30] Wang X, Tang L, Liu H, Wang L. Learning with multi-resolution overlapping communities. Knowledge and information systems. 2013; 36(2):517–535

[31] Yang J, Leskovec J. Overlapping community detection at scale: a nonnegative matrix factorization approach. In: Proceedings of the sixth ACM international conference on Web search and data mining. Rome, Italy: ACM; 2013. p. 587–596. 56.

[32] Lancichinetti A, Radicchi F, Ramasco JJ, Fortunato S. Finding statistically significant communities in networks. PloS one. 2011; 6(4):e18961.

[33] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: Fast learning with graph convolutional networks via importance sampling. In International Conference on Learning Representations (ICLR), 2018b

[34] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Proceedings ofthe 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 18, 2018a. ISBN

[35] Jianfei Chen, Jun Zhu, and Le Song. Stochastic training of graph convolutional networks with variance reduction. In ICML, pp. 941–949, 2018a

[36] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '18, pp. 1416–1424, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5552-0.

[37] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. In Advances in Neural Information Processing Systems, pp. 4558–4567, 2018.

[38] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor K. Prasanna. Accurate, efficient and scalable graph embedding. CoRR, abs/1810.11899, 2018. URL http://arxiv.org/abs/1810.11899

[39] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An ef- ficient algorithm for training deep and large graph convolutional networks. CoRR, abs/1905.07953, 2019. URL http://arxiv.org/abs/1905.07953

[40] [12] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Li'o, and Y. Bengio, "Graph attention networks," CoRR, 2017. Velickovic et al. (2017);

[41] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. arXiv preprint arXiv:1803.07294, 2018

[42] Haonan Lu, Seth H. Huang, Tian Ye, and Xiuyan Guo. Graph star net for generalized multi-task learning. CoRR, abs/1906.12330, 2019. URL http://arxiv.org/abs/1906.12330

[43] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Gunnemann. Personalized embedding propa- gation: Combining neural networks on graphs with personalized pagerank. CoRR, abs/1810.05997, 2018. URL http://arxiv.org/abs/1810.05997

[44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385

[45] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. arXiv preprint arXiv:1806.03536, 2018

[46] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "GraphSAINT: Graph Sampling Based Inductive Learning Method," no. 2018, 2019, [Online]. Available: http://arxiv.org/abs/1907.04931..

[47] Mihalkova, Lilyana and Mooney, Raymond J. 2009. Transfer learning from minimal target data by mapping across relational domains. *Twenty-First International Joint Conference on Artificial Intelligence.*

[48] L. Getoor, B. Taskar, editors. Introduction to Statistical Relational Learning. MIT Press, Cambridge, MA, 2007.

[49] M. Richardson and P. Domingos. Markov logic networks. Machine Learning, 62:107–136, 2006.

[50] L. Mihalkova, T. Huynh, and R. J. Mooney. Mapping and revising Markov logic networks for transfer learning. (AAAI-07).

[51] J. Davis, P. Domingos. Deep transfer via second-order markov logic. *In Proceedings of the AAAI Workshop on Transfer Learning For Complex Tasks.* 2008

[52] Kenneth D. Forbus , Dan Oblinger. Making SME greedy and pragmatic. (CogSci-90).

[53] J. Lee, H. Kim, J. Lee, and S. Yoon, "Transfer learning for deep learningon graph-structured data." in AAAI, 2017, pp. 2154–2160

[54] Sonawane, S., and Kulkarni, P. 2014. Graph based representation and analysis of text document: A survey of techniques. *International Journal of Computer Applications* 96(19).

[55] Henaff, M.; Bruna, J.; and LeCun, Y. 2015. Deep convolutional networks on graph - structured data. arXiv:1506.05163.

[56] Quanyu Dai, Xiao Shen, Xiao-Ming Wu, Dan Wang. 2019. Network Transfer Learning via Adversarial Domain Adaptation with Graph Convolution. arXiv:1909.01541(2019)

[57] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," in ICML, 2016, pp. 40–48.

[58] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in ICLR, 2017.

[59] J. Ni, S. Chang, X. Liu, W. Cheng, H. Chen, D. Xu, and X. Zhang, "Co-regularized deep multi-network embedding," in WWW, 2018, pp. 469–478.

[60] L. Xu, X. Wei, J. Cao, and P. S. Yu, "Embedding of embedding (EOE): joint embedding for coupled heterogeneous networks," in WSDM, 2017, pp. 741–749.

[61] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.

[62] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," Neurocomputing, vol. 312, pp. 135–153, 2018

[63] Ma, Ning, et al. "Few-Shot Graph Classification with Model Agnostic Meta-Learning."arXiv preprint arXiv:2003.08246 (2020).

[64] Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks."*Proceedings of the 34th International Conference on Machine Learning-Volume 70.* JMLR. org, 2017

[65] Zhengxiao Du, Xiaowei Wang, Hongxia Yang, Jingren Zhou, and Jie Tang. Sequential scenario- specific meta learner for online recommendation. In KDD'19, page 2895–2904, 2019.

[66] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In ICLR, 2018.

[67] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D. Yoo. Edge-labeling graph neural net- work for few-shot learning. In CVPR, June 2019.

[68] Lu Liu, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. Learning to propagate for graph meta-learning. In NeurIPS, 2019.

[69] Huaxiu Yao, Xian Wu, Zhiqiang Tao, Yaliang Li, Bolin Ding, Ruirui Li, and Zhenhui Li. Au- tomated relational meta-learning. In ICLR, 2020.

[70] Jatin Chauhan, Deepak Nathani, and Manohar Kaul. Few-shot learning on graphs via super- classes based on graph spectral measures. In ICLR, 2020.