

Обработка графов

Виктор Китов

victorkitov.github.io



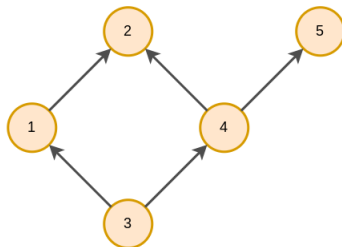
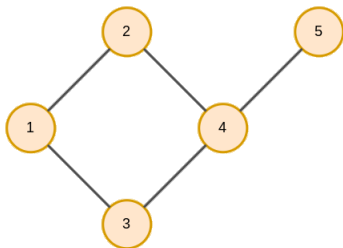
Содержание

- 1 Введение
- 2 Усложнения графов
- 3 Примеры мультиграфов
- 4 Инвариантность к перенумерации вершин
- 5 Графовые нейросети
- 6 Эмбединги вершин по геометрии графа

Граф и др. типы данных

Граф: (V, E) .

- рёбра направленные / ненаправленные
- могут задаваться веса связей W (обычно ≥ 0)



Социальные сети



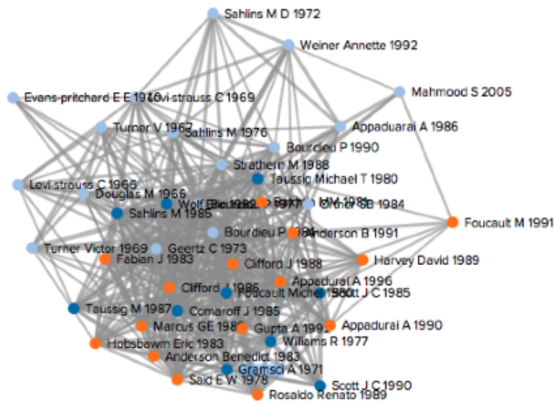
Социальная сеть:

- узлы: люди
- связи: состояние дружбы, сообщения, репосты.

Задачи на социальных сетях

- Предсказание свойств узла (node prediction):
 - не указанные атрибуты: возраст, пол, образование, профессия
 - интересы
 - является ли аккаунт ботом?
- Предсказание свойств связей:
 - классификация отношений: друзья, семья, коллеги, одноклассники
 - определение силы связи
 - оценка направления влияния (выявление лидеров мнений)
- Восстановление связей (link prediction):
 - рекомендации друзей
- Обнаружение сообществ (community prediction):
обнаружение сильно связанных подграфов, отвечающих
 - коллегам по работе, выпускникам одного ВУЗа и т.д.

Граф цитирований научных работ



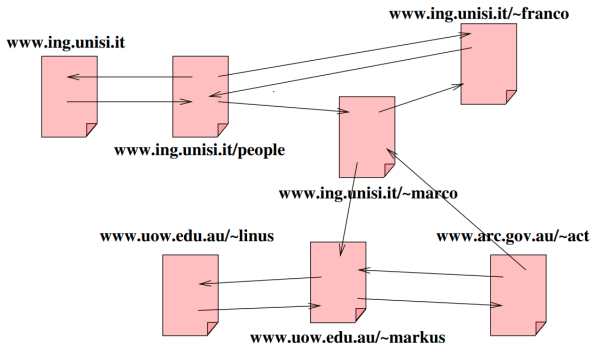
Граф цитирований:

- узлы: статьи
- связи: цитирования др. работ.

Задачи на графе цитирований

- Предсказание свойств узла:
 - тематика работы
 - ключевые слова
 - авторитетность работы
- Восстановление связей:
 - рекомендации для полного обзора литературы
- Обнаружение сообществ:
 - направления исследований, научные школы

Граф веб-страниц



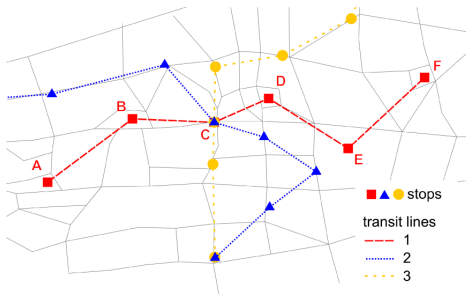
Граф веб-страниц:

- узлы: веб-страницы
- связи: гиперссылки, факт просмотра одним пользователем.

Задачи на графе веб-страниц

- Предсказание свойств узла:
 - важность страницы при ранжировании
 - тематика страницы
- Восстановление связей:
 - рекомендация страниц похожей тематики
- Обнаружение сообществ:
 - построение семантической карты интернета
 - выявление рубрик для автоматической рубрикации страниц

Транспортный/логистический граф



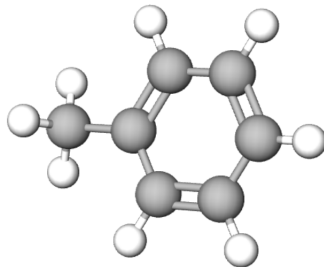
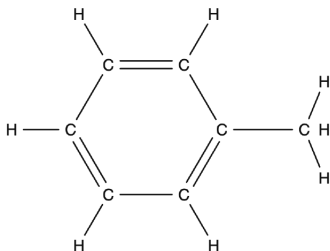
Транспортная сеть:

- узлы: локации (перекрёстки, остановки, станции, склады)
- связи: перемещения между локациями.

Задачи на транспортной сети

- Предсказание свойств узла:
 - предсказание загруженности станции/склада
 - прогноз времени ожидания в узле
- Предсказание связей:
 - прогноз времени прохождения по ребру
 - оценка пропускной способности
- Восстановление связей:
 - рекомендация новых маршрутов/рейсов

Молекула как граф



Молекула как граф:

- узлы: атомы
- связи: химические связи

Молекула как граф

Классификация графа:

- предсказывать химические свойства молекулы
- если лекарство: лечебный эффект

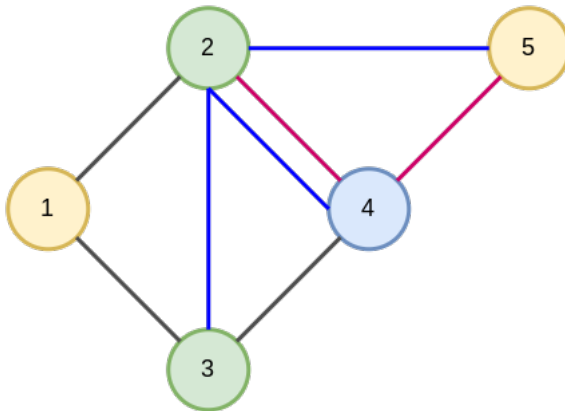
Генерация графа:

- создание вещества с заданными свойствами
 - легкий и прочный,
- создание лекарств с желаемым эффектом
 - лечит определённую болезнь
 - совместимо с другими лекарствами
 - обладает минимумом побочных эффектов

Содержание

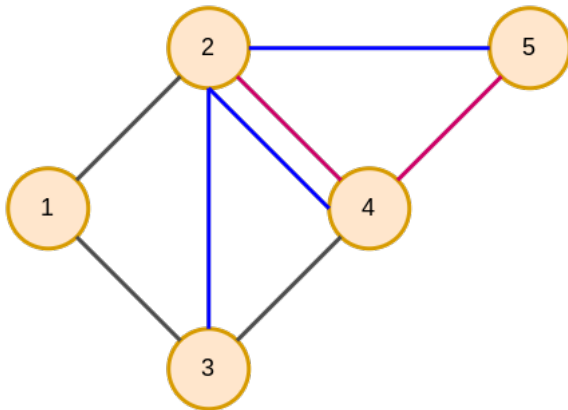
- 1 Введение
- 2 Усложнения графов**
- 3 Примеры мультиграфов
- 4 Инвариантность к перенумерации вершин
- 5 Графовые нейросети
- 6 Эмбединги вершин по геометрии графа

Гетерогенный граф (узлы разных типов)



Граф знаний: люди, города, страны, события, компании и т.д.

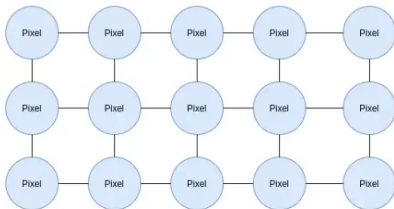
Мультиграф (рёбра разных типов)



В соц. сети пользователи добавили в друзья, писали друг другу сообщения, ставили друг другу реакции.

Изображения и текст как мультиграф

- **Изображения:** $\uparrow\downarrow\leftarrow\rightarrow$ сосед.
- **Текст:** $\leftarrow\rightarrow$ сосед, соседи в рамках одного предложения.

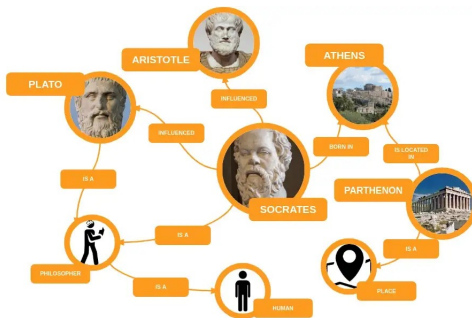


- Используя мультиграфовые сети, можно строить мультимодальные системы.

Содержание

- 1 Введение
- 2 Усложнения графов
- 3 Примеры мультиграфов**
- 4 Инвариантность к перенумерации вершин
- 5 Графовые нейросети
- 6 Эмбединги вершин по геометрии графа

Граф знаний



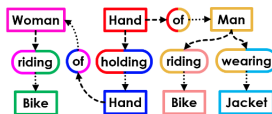
Граф знаний:

- узлы: объекты мира, сущности, понятия.
- связи: отношения между ними

Граф знаний

- Предсказание свойств узла:
 - человек: время жизни, должность, ...
 - город: уровень жизни, развитие культуры, ...
- Восстановление связей:
 - выявление взаимосвязей между сущностями, новые знания (knowledge discovery)

Граф сцены (scene graph)¹



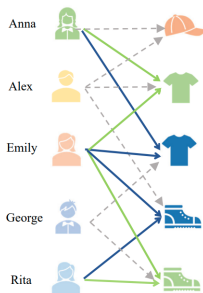
Граф сцены (scene graph):

- узлы: объекты
- связи: типы взаимосвязей объектов

Классификация сцены по графу (семейная, работа, панорама).

¹<https://www.arxiv-vanity.com/papers/2001.02314/>

Взаимодействие пользователей и товаров



Взаимодействия пользователей и товаров (user-item graph)²

- узлы: пользователи, товары
- связи: их взаимодействие (купил товар, просмотрел видео, лайкнул описание)

²Stacked Mixed-Order Graph Convolutional Networks for Collaborative Filtering (2020).

Граф взаимодействия пользователей и товаров

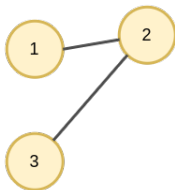
- Предсказание свойств узла:
 - категоризация пользователей и товаров
- Предсказание свойств связей:
 - прогноз интенсивности пользования сервисом
- Восстановление связей:
 - рекомендательная система

Содержание

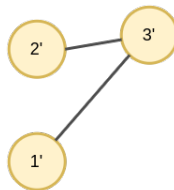
- 1 Введение
- 2 Усложнения графов
- 3 Примеры мультиграфов
- 4 Инвариантность к перенумерации вершин**
- 5 Графовые нейросети
- 6 Эмбединги вершин по геометрии графа

Перенумеровка вершин

а) исходный граф



б) эквивалентный граф



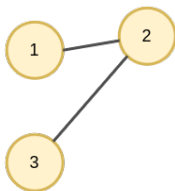
Матрицы данных H , смежности A , перестановок P :

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad H = \begin{pmatrix} 1 & 10 & 100 \\ 2 & 20 & 200 \\ 3 & 30 & 300 \\ 4 & 40 & 400 \end{pmatrix} \quad P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

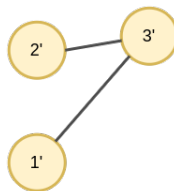
Прогнозирование на графе - инвариантность к перестановкам.
На отдельных вершинах - эквивариантность к перестановкам.

Перенумеровка матрицы смежности

а) исходный граф



б) эквивалентный граф

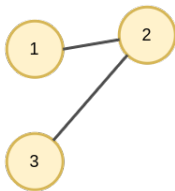


$$A' = P^T \cdot A \cdot P$$

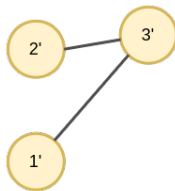
$$= \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

Перенумеровка матрицы данных

а) исходный граф

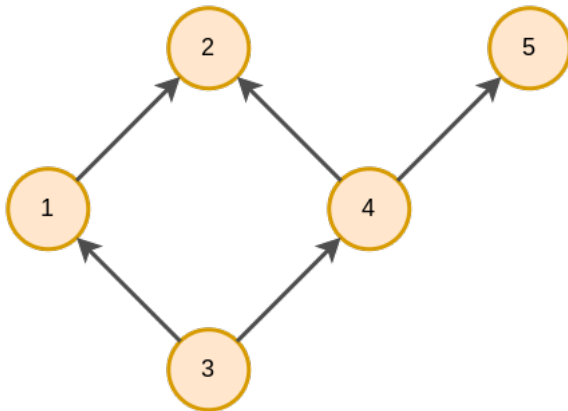


б) эквивалентный граф



$$H' = H \cdot P = \begin{pmatrix} 1 & 10 & 100 \\ 2 & 20 & 200 \\ 3 & 30 & 300 \\ 4 & 40 & 400 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 100 & 1 & 10 \\ 200 & 2 & 20 \\ 300 & 3 & 30 \\ 400 & 4 & 40 \end{pmatrix}$$

Направленный граф



Пример: цитирование статей, ссылки в веб-страницах.

Инструменты и датасеты

- Библиотеки:

- PyTorch Geometric (популярнее)
- Deep Graph Library (эффективнее)



PyG



- Коллекция графовых датасетов - Open Graph Benchmark



Содержание

- 1 Введение
- 2 Усложнения графов
- 3 Примеры мультиграфов
- 4 Инвариантность к перенумерации вершин
- 5 Графовые нейросети**
- 6 Эмбединги вершин по геометрии графа

Кодирование вершин

$$v_i \rightarrow \mathbf{h}_i \in \mathbb{R}^D, \quad i = 1, 2, \dots, N.$$

- Эмбединг инициализируется:
 - признаками вершины
 - или случайно, если признаков нет
- Далее эмбединг уточняется по геометрии графа (алгоритм передачи сообщений)
- Инвариантность прогноз для графа целиком:

$$\hat{y}(H, A) \equiv \hat{y}(HP, P^T AP), \quad P - \text{матрица перестановок}$$

- Эквивариантность прогноза для отдельных вершин:

$$\hat{y}(H, A) \cdot P = \hat{y}(HP, P^T AP)$$

Задачи на графе целиком

Регрессия ($\mathbf{i} = [1, 1, \dots, 1]^T \in \mathbb{R}^D$):

$$\hat{y}(G) = w_0 + \mathbf{w}^T \left(\frac{1}{N} \sum_{n=1}^N \mathbf{h}_n \right) = w_0 + \mathbf{w}^T \cdot H \cdot \mathbf{i} / N$$

Бинарная классификация:

$$\hat{y}(G) = \sigma \left(w_0 + \mathbf{w}^T \left(\frac{1}{N} \sum_{n=1}^N \mathbf{h}_n \right) \right) = \sigma(w_0 + \mathbf{w}^T \cdot H \cdot \mathbf{i} / N)$$

Многоклассовая классификация:

$$\begin{pmatrix} p(\hat{y} = 1|G) \\ p(\hat{y} = 2|G) \\ \dots \\ p(\hat{y} = C|G) \end{pmatrix} = \text{SoftMax}(\mathbf{w}_0 + W \cdot H \cdot \mathbf{i} / N),$$

Зависят от $\frac{1}{N} \sum_{n=1}^N \mathbf{h}_n \Rightarrow$ инвариантность (альт-вы среднему?)

Задачи на отдельной вершине

Регрессия:

$$\hat{y}_i = w_0 + \mathbf{w}^T \mathbf{h}_i$$

Бинарная классификация:

$$\hat{y}_i = \sigma(w_0 + \mathbf{w}^T \mathbf{h}_i)$$

Многоклассовая классификация:

$$\begin{pmatrix} p(\hat{y}_i = 1|G) \\ p(\hat{y}_i = 2|G) \\ \dots \\ p(\hat{y}_i = C|G) \end{pmatrix} = \text{SoftMax}(\mathbf{w}_0 + W\mathbf{h}_i),$$

- $\sigma(u) = 1/(1 + e^{-u})$ - сигмоида, $\mathbf{i} = [1, 1, \dots, 1]^T \in \mathbb{R}^D$.
- зависит только от $\mathbf{h}_i \Rightarrow$ эквивариантность.

Восстановление связей

$$p((v_i, v_j)\text{-connected} \mid \mathbf{h}_i, \mathbf{h}_j) = \sigma(\mathbf{h}_i^T \mathbf{h}_j)$$

$$p((v_i, v_j)\text{-connected} \mid \mathbf{h}_i, \mathbf{h}_j) > t \implies \text{connect } v_i, v_j$$

Если $\{\mathbf{h}_i\}_i$ настраиваются для решения других задач, то

$$p((v_i, v_j)\text{-connected} \mid \mathbf{h}_i, \mathbf{h}_j) = \sigma(w_0 + \mathbf{h}_i^T W \mathbf{h}_j), \quad W = W^T$$

Симметричные W можно перебирать, подбирая нижнетреугольную L :

$$W = W^T = L \cdot L^T \quad (\text{разложение Холецкого})$$

Более общий вариант:

$$p((v_i, v_j)\text{-connected} \mid \mathbf{h}_i, \mathbf{h}_j) = \sigma(\text{MLP}_W(\mathbf{h}_i, \mathbf{h}_j) + \text{MLP}_W(\mathbf{h}_j, \mathbf{h}_i))$$

Алгоритм передачи сообщений³

ВХОД:

- ненаправленный граф $G = (V, E)$
- начальные эмбединги вершин $\{\mathbf{h}_n^0\}$
- функции $\text{Aggregate}(\cdot)$ и $\text{Update}(\cdot)$

АЛГОРИТМ:

для $k = 1, 2, \dots K$:

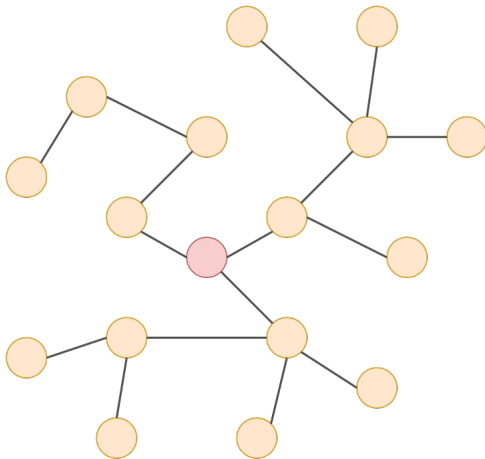
- $\mathbf{z}_n^k = \text{Aggregate}\left(\{\mathbf{h}_m^k\}_{m \in \mathcal{N}(n)}\right)$
- $\mathbf{h}_n^{k+1} = \text{Update}(\mathbf{h}_n^k, \mathbf{z}_n^k)$

ВЫХОД:

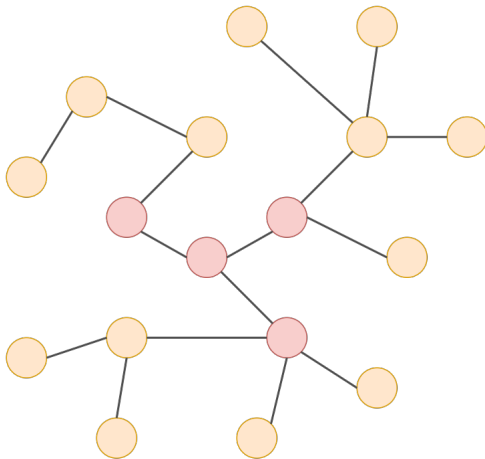
- итоговые эмбединги вершин \mathbf{h}_n^K , $n = 1, 2, \dots N$.

³ [Message passing neural network.](#)

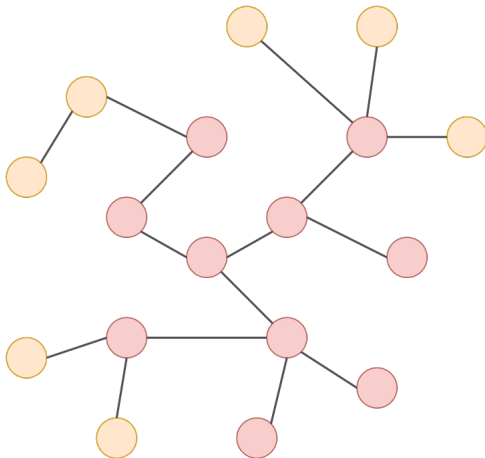
Алгоритм передачи сообщений (инициализация)



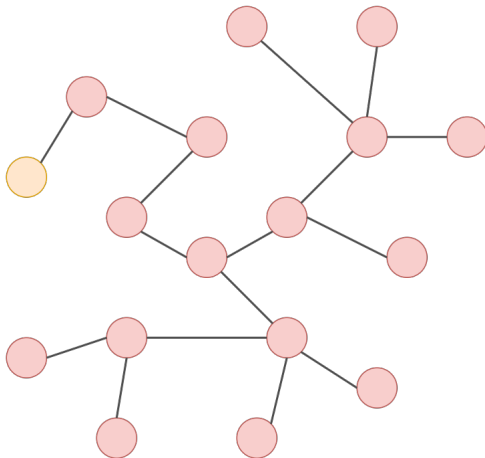
Алгоритм передачи сообщений (итерация 1)



Алгоритм передачи сообщений (итерация 2)



Алгоритм передачи сообщений (итерация 3)



Функция Aggregate

$$\text{Aggregate}(\cdot) = \sum_{m \in \mathcal{N}(n)} \mathbf{h}_m^k$$

$$\text{Aggregate}(\cdot) = \frac{1}{|\mathcal{N}(n)|} \sum_{m \in \mathcal{N}(n)} \mathbf{h}_m^k$$

$$\text{Aggregate}(\cdot) = \frac{1}{\sqrt{|\mathcal{N}(n)|}} \sum_{m \in \mathcal{N}(n)} \frac{1}{\sqrt{|\mathcal{N}(m)|}} \mathbf{h}_m^k$$

- Усреднение более устойчиво, зато сумма даёт информацию о #соседей.
- Третий вариант-учитывает #соседей для исходной и соседней вершины.

Функция Aggregate

$$\text{Aggregate}(\cdot) = \text{maximum} \left(\left\{ \mathbf{h}_m^k \right\}_{m \in \mathcal{N}(n)} \right)$$

Предотвращает сглаживание эмбедингов в рез-те многократных усреднений.

$$\text{Aggregate}(\cdot) = \text{MLP}_w \left(\sum_{m \in \mathcal{N}(n)} \text{MLP}_v \left(\mathbf{h}_m^k \right) \right)$$

Наиболее общий вид дифференцируемой ф-ции, инвариантной к перенумеровкам вершин.

Агрегация со вниманием (attention)

- 1 вычисляется внимание, с которым узел v_n должен смотреть на каждого своего соседа v_m , используя некоторую функцию $s_{nm} = g(\mathbf{h}_n^k, \mathbf{h}_m^k)$;
- 2 нормировка:

$$\{a_{nm}\}_{m \in \mathcal{N}(n)} = \text{SoftMax} \left(\{s_{nm}\}_{m \in \mathcal{N}(n)} \right)$$

- 3 взвешенная агрегация:

$$\mathbf{z}_n^k = \sum_{m \in \mathcal{N}(n)} a_{nm} \mathbf{h}_m^k$$

Примеры внимания

$$a_{mn} = \frac{\exp(\mathbf{h}_n^T W \mathbf{h}_m)}{\sum_{m' \in \mathcal{N}(n)} \exp(\mathbf{h}_n^T W \mathbf{h}_{m'})}$$

$$a_{mn} = \frac{\exp(\text{MLP}(\mathbf{h}_n, \mathbf{h}_m))}{\sum_{m' \in \mathcal{N}(n)} \exp(\text{MLP}(\mathbf{h}_n, \mathbf{h}_{m'}))}$$

Функция Update

$$\mathbf{h}_n^{k+1} = \text{Update}(\mathbf{h}_n^k, \mathbf{z}_n^k) = f(W\mathbf{h}^k + V\mathbf{z}_n^k + \mathbf{b}),$$

$f(\cdot)$ - нелинейность

Часто полагают $V = W$:

$$\mathbf{h}_n^{k+1} = f\left(W \sum_{m \in \mathcal{N}(n) \cup n} \mathbf{h}_m^k + \mathbf{b}\right)$$

Борьба со сглаженными эмбедингами

- За счёт многократных усреднений эмбединги сглаживаются.
- Способы борьбы:
 - ResNet блоки:

$$\mathbf{h}_n^{k+1} = \text{Update}(\mathbf{h}_n^k, \mathbf{z}_n^k) + \mathbf{h}_n^k$$

- использование в конце эмбедингов со всех слоёв:

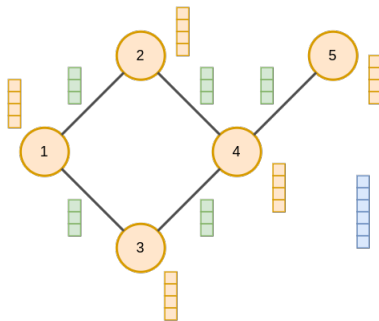
$$\mathbf{h}_n^{\text{final}} = [\mathbf{h}_n^1; \mathbf{h}_n^1; \dots \mathbf{h}_n^K]$$

Эмбединги

- Помимо эмбедингов вершин $v_i \rightarrow \mathbf{h}_i$ могут быть эмбединги рёбер

$$(ij) \rightarrow \mathbf{e}_{ij}$$

и эмбединг графа \mathbf{g} :



Обозначения:

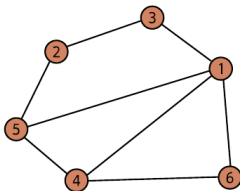


Задачи на рёбрах

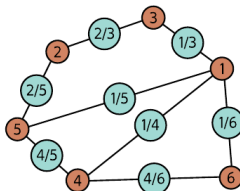
На рёбрах можно решать те же задачи, перейдя от графа к соотв. графу рёбер (edge graph)

- рёбра \rightarrow рёбра
- эмбединги рёбер \rightarrow эмбединги узлов

a) Original graph



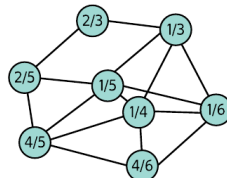
b)



Add new node
at each original edge

c)

Edge graph



Connect new nodes if
original edges shared node

Учёт эмбедингов рёбер и всего графа

ВХОД:

- ненаправленный граф $G = (V, E)$
- начальные эмбединги вершин $\{\mathbf{h}_n^0\}$
- начальные эмбединги рёбер $\{\mathbf{e}_{nm}^0\}$
- начальный эмбединг графа \mathbf{g}^0
- Функции $\text{Aggregate}(\cdot)$ и $\text{Update}(\cdot)$

АЛГОРИТМ:

для $k = 1, 2, \dots, K$:

- $\mathbf{e}_{nm}^{k+1} = \text{Update}_{\text{edge}}(\mathbf{e}_{nm}^k, \mathbf{h}_n^k, \mathbf{h}_m^k, \mathbf{g}^k)$
- $\mathbf{z}_n^{k+1} = \text{Aggregate}_{\text{node}}(\{\mathbf{e}_{nm}^k\}_{m \in \mathcal{N}(n)})$
- $\mathbf{h}_n^{k+1} = \text{Update}_{\text{node}}(\mathbf{h}_n^k, \mathbf{z}_n^{k+1}, \mathbf{g}^k)$
- $\mathbf{g}^{k+1} = \text{Update}(\mathbf{g}^k, \{\mathbf{h}_n^{k+1}\}, \{\mathbf{e}_{nm}^{k+1}\})$

ВЫХОД:

- итоговые эмбединги $\{\mathbf{h}_n^K\}, \{\mathbf{e}_{nm}^K\}, \mathbf{g}^K$

Инвариантность к перенумерациям

Итерации алгоритма обмена сообщениями:

$$H^1 = F(X, A, W^1)$$

$$H^2 = F(H^1, A, W^2)$$

...

$$H^K = F(H^K, A, W^K)$$

- A - матрица смежности вершин,
- $H^i = [h_1^i, \dots, h_N^i] \in \mathbb{R}^{D \times N}$ - эмбединги вершин.

Должно быть выполнено условие эквивариантности к перенумеровкам⁴, заданным матрицей перестановок P .

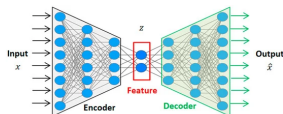
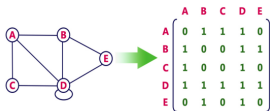
⁴Запишите шаг передачи сообщений в матричном виде и покажите, что эквивариантность выполнена, учитывая правила $PP^T = P^T P = I$, $H' = HP$, $A' = P^T AP$.

Содержание

- 1 Введение
- 2 Усложнения графов
- 3 Примеры мультиграфов
- 4 Инвариантность к перенумерации вершин
- 5 Графовые нейросети
- 6 Эмбединги вершин по геометрии графа

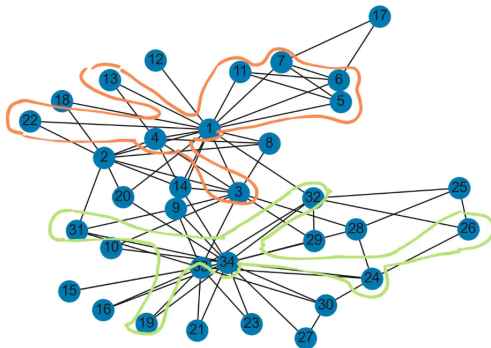
Эмбединги через автокодировщик⁵

- Предыдущие подходы простые.
- Structural Deep Network Embedding (SDNE): ↑ сложность генерации эмбедингов за счёт многослойного автокодировщика.



⁵<https://www.kdd.org/kdd2016/papers/files/rfp0191-wangAemb.pdf>

DeepWalk⁶



- Для вершин генерируем пути обхода небольшой длины.
- Скользим окном фикс. ширины по сгенерированному пути.
- По центр. v_t предсказываем соседей окна (SkipGram).

⁶<https://arxiv.org/pdf/1403.6652.pdf>

DeepWalk

$$\frac{1}{T} \sum_{t=1}^T \sum_{-c \leq i \leq c, i \neq 0} \ln p(v_{t+i}|v_t) \rightarrow \max_{\theta}$$

$$p(v_{t+i}|v_t) = \frac{\exp\left(\tilde{\phi}_{v_t}^T \phi_{v_{t+i}}\right)}{\sum_{v=1}^N \exp\left(\tilde{\phi}_{v_t}^T \phi_v\right)}$$

- Знаменатель вычисляется за $O(N)$.
 - используем Hierarchical SoftMax или Negative Sampling.
- Эмбединги: $v \rightarrow \phi_v$ либо $\tilde{\phi}_v$, либо $[\phi_v; \tilde{\phi}_v]$.

Node2vec⁷

- Node2vec - развитие DeepWalk.
- В разных задачах от эмбединга нужны разные признаки:
 - глобальные (характеристика сообщества, которому узел принадлежит)
 - блуждание - поиском в глубину
 - локальные (структурная роль вершины [переход между сообществами, хаб в центре, точка на краю])
 - блуждание - поиск в ширину

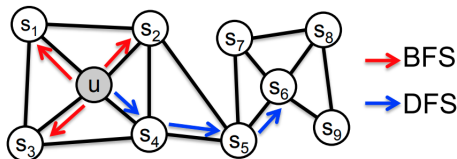


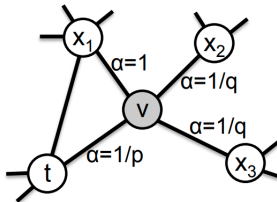
Figure 1: BFS and DFS search strategies from node u ($k = 3$).

⁷<https://arxiv.org/pdf/1607.00653.pdf>

Node2vec

$$p(v \rightarrow x|v, t) = \begin{cases} \gamma \cdot \frac{1}{p} & \text{если } x = t, \\ \gamma \cdot 1 & \text{если } d(t, x) = 1, \\ \gamma \cdot \frac{1}{q} & \text{если } d(t, x) = 2, \end{cases}$$

- обход в ширину (малое p)
 - или обход в глубину (малое q)
- Иллюстрация генерации α для узла v (только что перешли из t)
 - d_{tx} - мин. путь от t до x



Node2vec

На примере совстречаемости персонажей Les Miserables запуск node2vec (с последующей кластеризацией эмбедингов цветом)

- $p = 1, q = 0.5$: обнаружение сообществ
- $p = 1, q = 2$: определение структурных ролей персонажей [эпизодические, связующие, остальные]

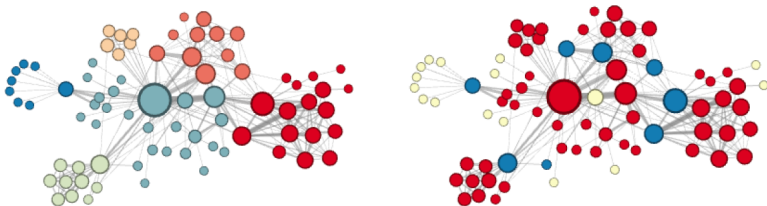


Figure 3: Complementary visualizations of Les Misérables co-occurrence network generated by *node2vec* with label colors reflecting homophily (top) and structural equivalence (bottom).

Дополнительная информация

- Простая обзорная статья.
- Обзор графовых нейросетей.
- Другой обзор графовых нейросетей.
- Книга Graph Representation Learning (2020).