Обработка графов

Виктор Китов
victorkitov.github.io

Содержание

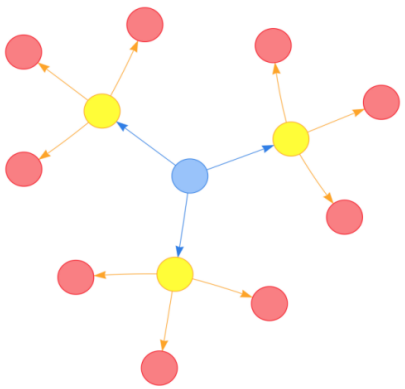
Граф и др. типы данных

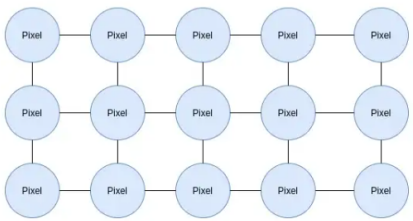
Граф: (V, E).

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

Нет понятий левый/правый/верхний/нижний сосед

- как в изображениях и текстах





Социальные сети

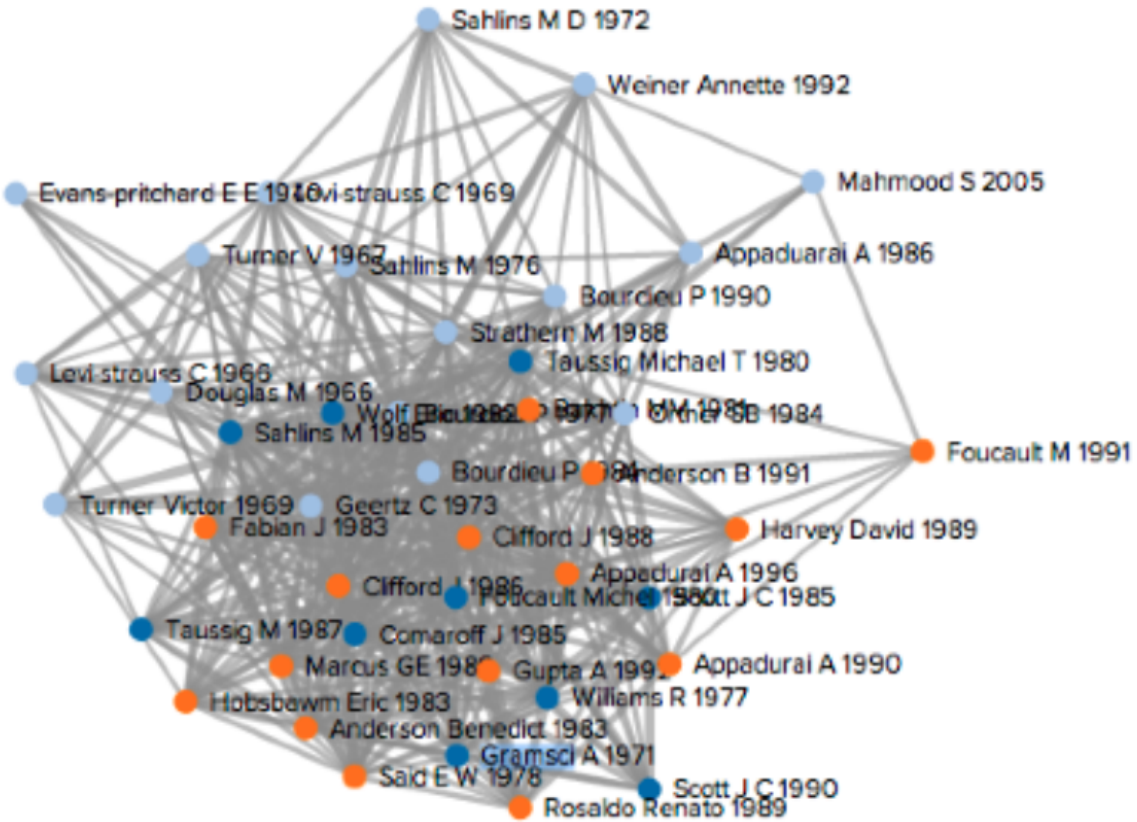
Социальная сеть:

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

Задачи на социальных сетях

- Предсказание свойств узла (node prediction):
  - не указанные аттрибуты: возраст, пол, образование, профессия
  - интересы
  - является ли аккаунт ботом?
- Предсказание 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



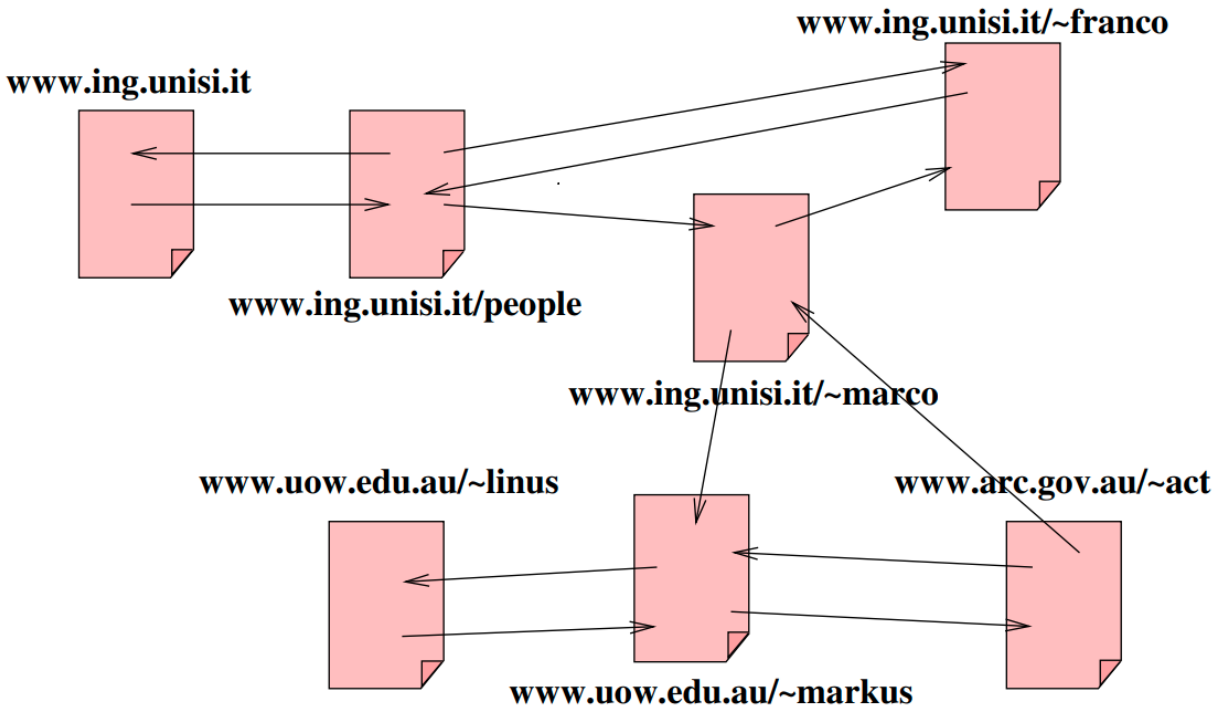
Граф цитирований:

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

Задачи на графе цитирований

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

Граф веб-страниц



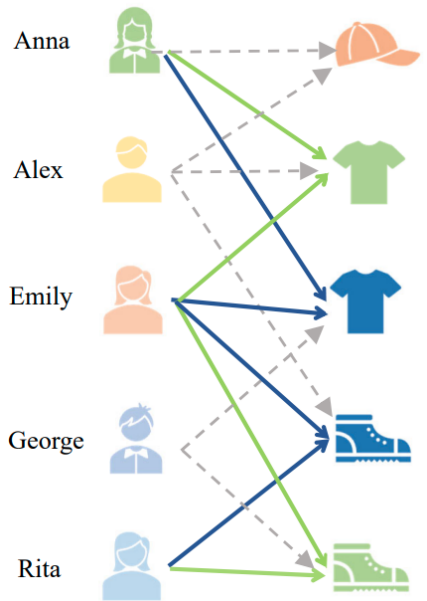
Граф веб-страниц:

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

Задачи на графе веб-страниц

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

Взаимодействие пользователей и товаров



Взаимодействия пользователей и товаров (user-item graph)[1]

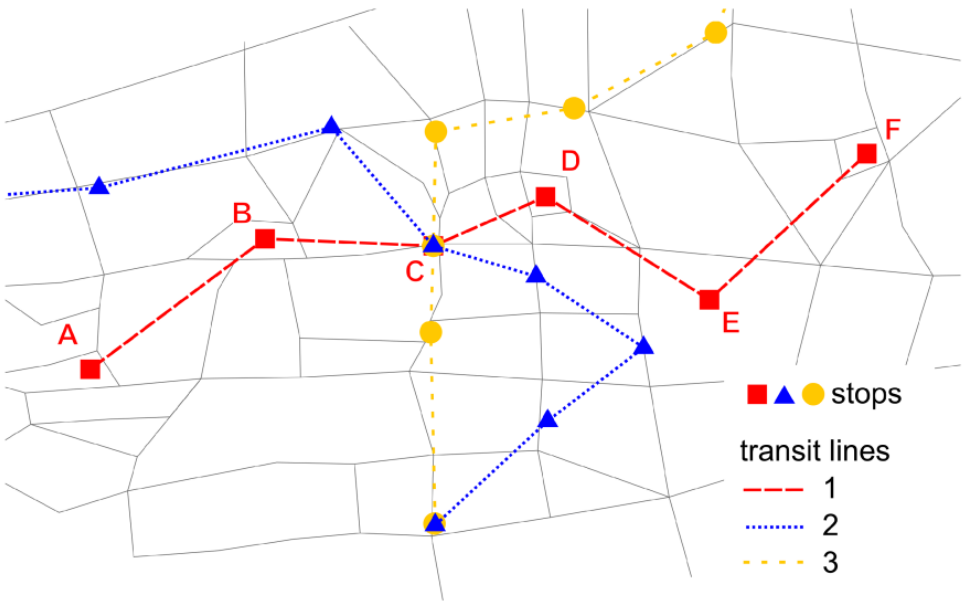
- узлы: пользователи, товары
- связи: их взаимодействие (купил товар, просмотрел видео, лайкнул описание)

[1] Stacked Mixed-Order Graph Convolutional Networks for Collaborative Filtering (2020).

Граф взаимодействия пользователей и товаров

- Предсказание свойств узла:
  - категоризация пользователей и товаров
- Предсказание связей:
  - рекомендательная система

Транспортная сеть



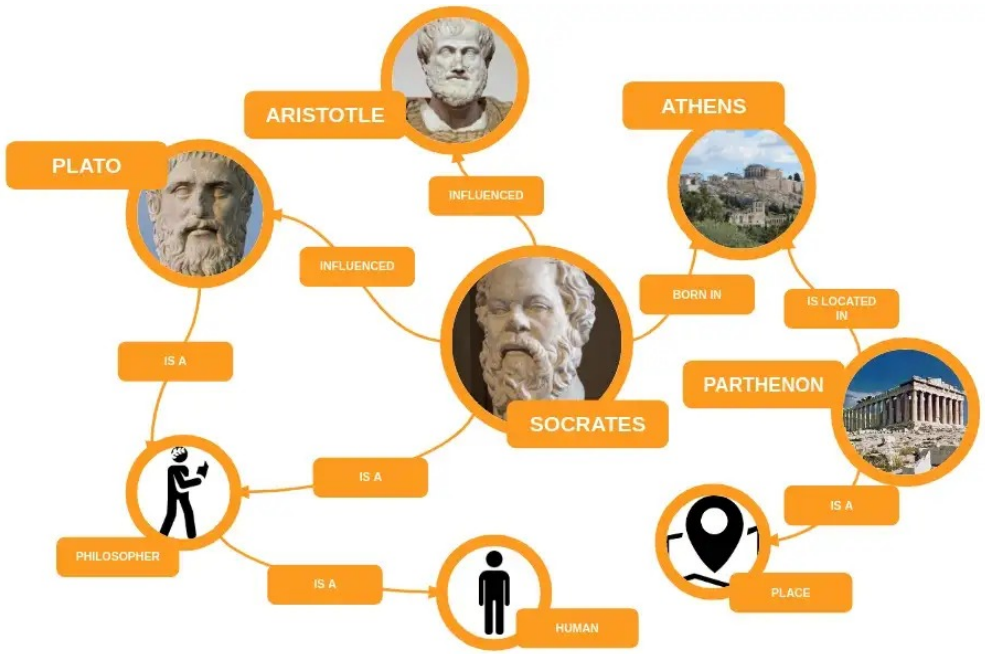
Транспортная сеть:

- узлы: локации
- связи: переезд из одной локации в другую.

Задачи на транспортной сети

- Предсказание свойств узла:
  - определение типов застройки (офисы/жилые дома/торговые центры)
- Предсказание связей:
  - рекомендация новых прямых маршрутов, дорог, авиарейсов.
- Обнаружение сообществ:
  - выявление самодостаточных регионов (например для расположения рекламы)

Граф знаний



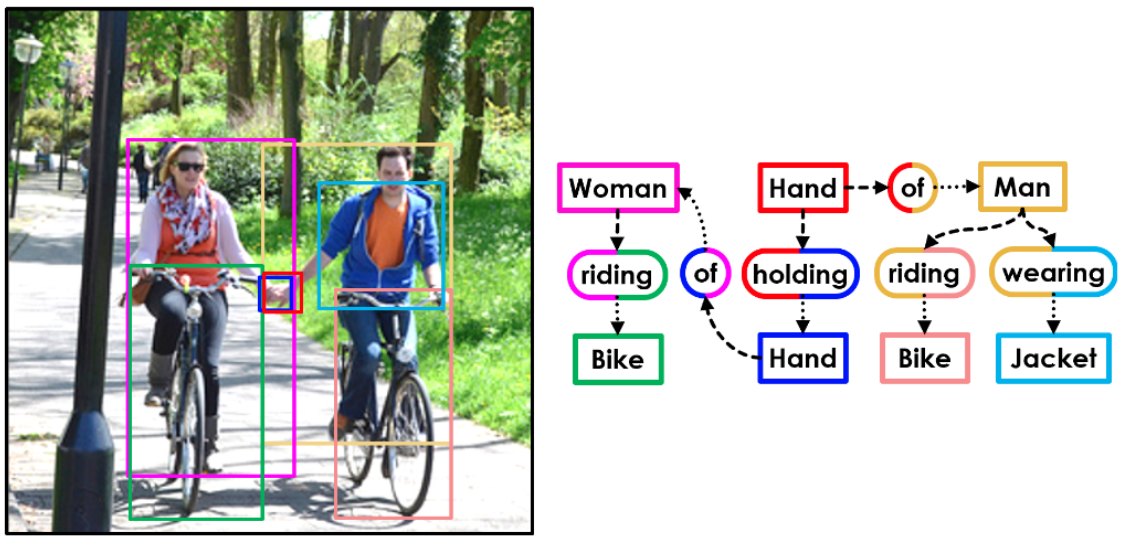
Граф знаний:

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

Граф знаний

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

Граф сцены (scene graph)[2]



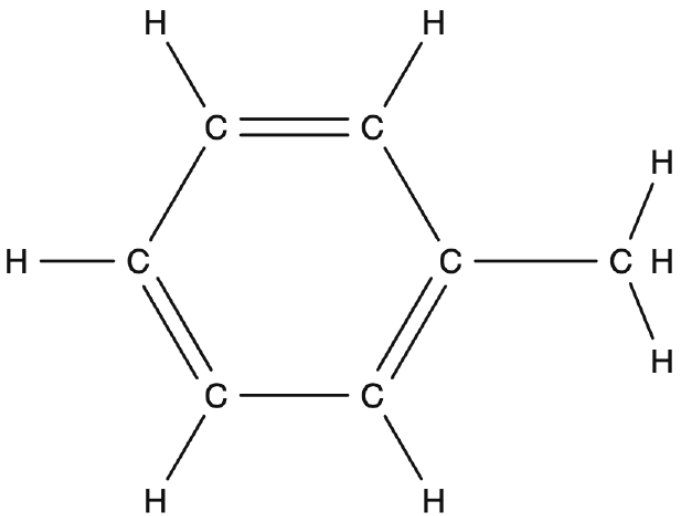
Граф сцены (scene graph):

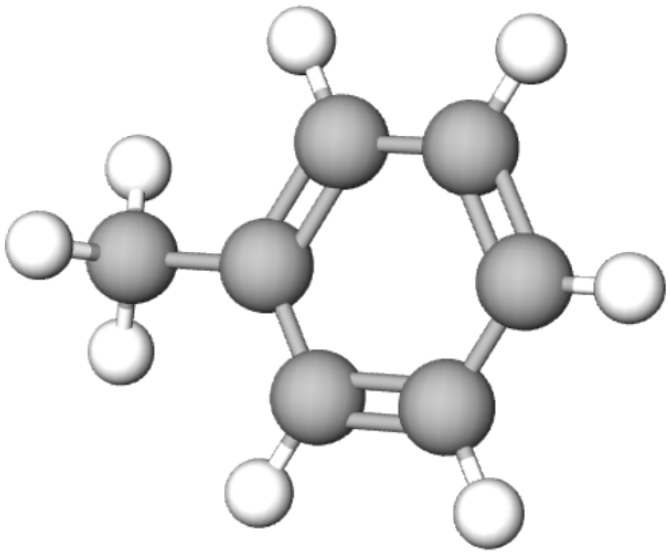
- узлы: объекты
- связи: типы взаимосвязей объектов

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

[2] https://www.arxiv-vanity.com/papers/2001.02314/

Молекула как граф





Молекула как граф:

- узлы: атомы
- связи: химические связи

Молекула как граф

Классификация графа:

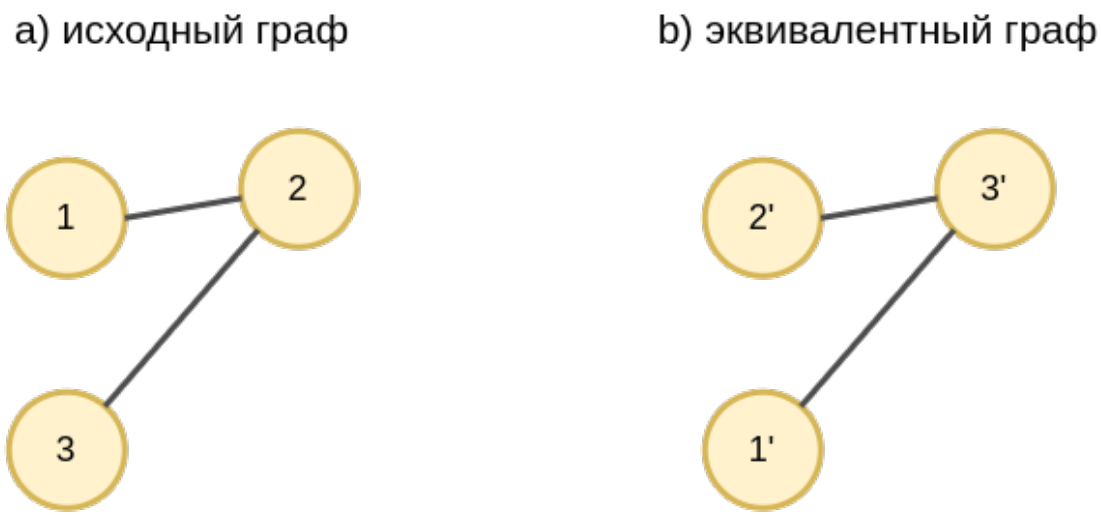
- предсказывать химические свойства молекулы
- если лекарство: лечебный эффект

Генерация графа:

- придумывание вещества с заданными свойствами
- придумывание лекарства с желаемым эффектом

Содержание

Перенумеровка вершин



Матрица данных H, матрица смежности A, матрица перестановок P:

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

Перенумеровка матрицы данных

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

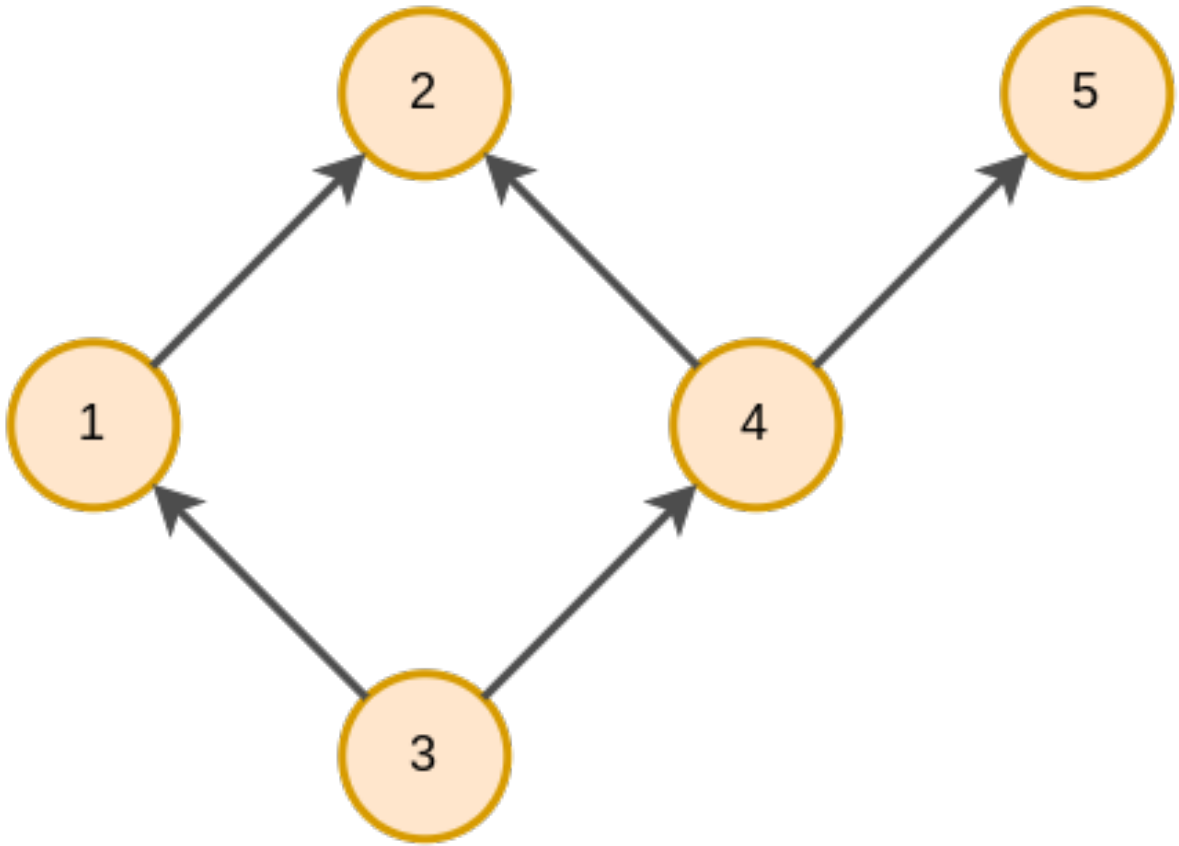
$$H' = H \cdot P = \begin{pmatrix} 1 & 10 & 100 \\ 2 & 20 & 200 \\ 3 & 30 & 300 \\ 4 & 40 & 400 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 100 & 1 & 10 \\ 200 & 2 & 20 \\ 300 & 3 & 30 \\ 400 & 4 & 40 \end{pmatrix}$$

Перенумеровка матрицы смежности

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

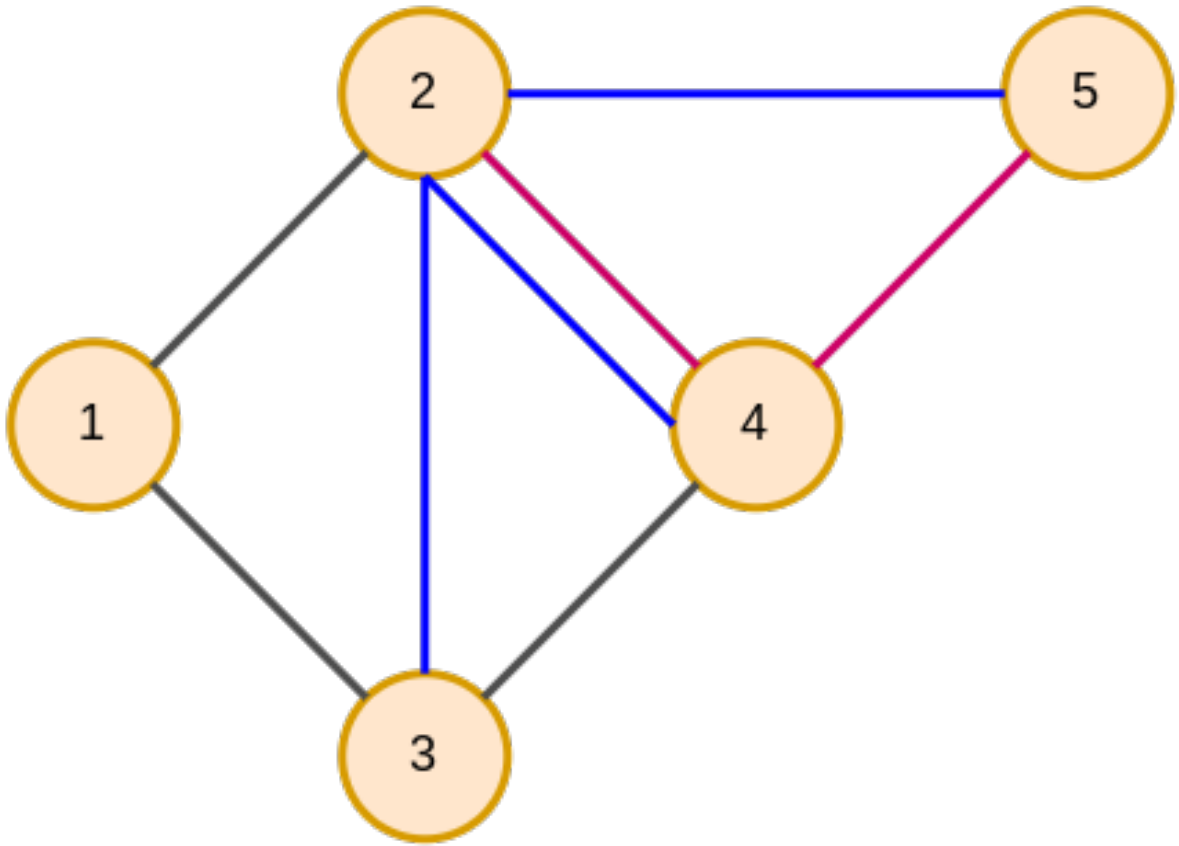
$$\begin{aligned} A' &= P^T \cdot A \cdot P \\ &= \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \end{aligned}$$

Направленный граф



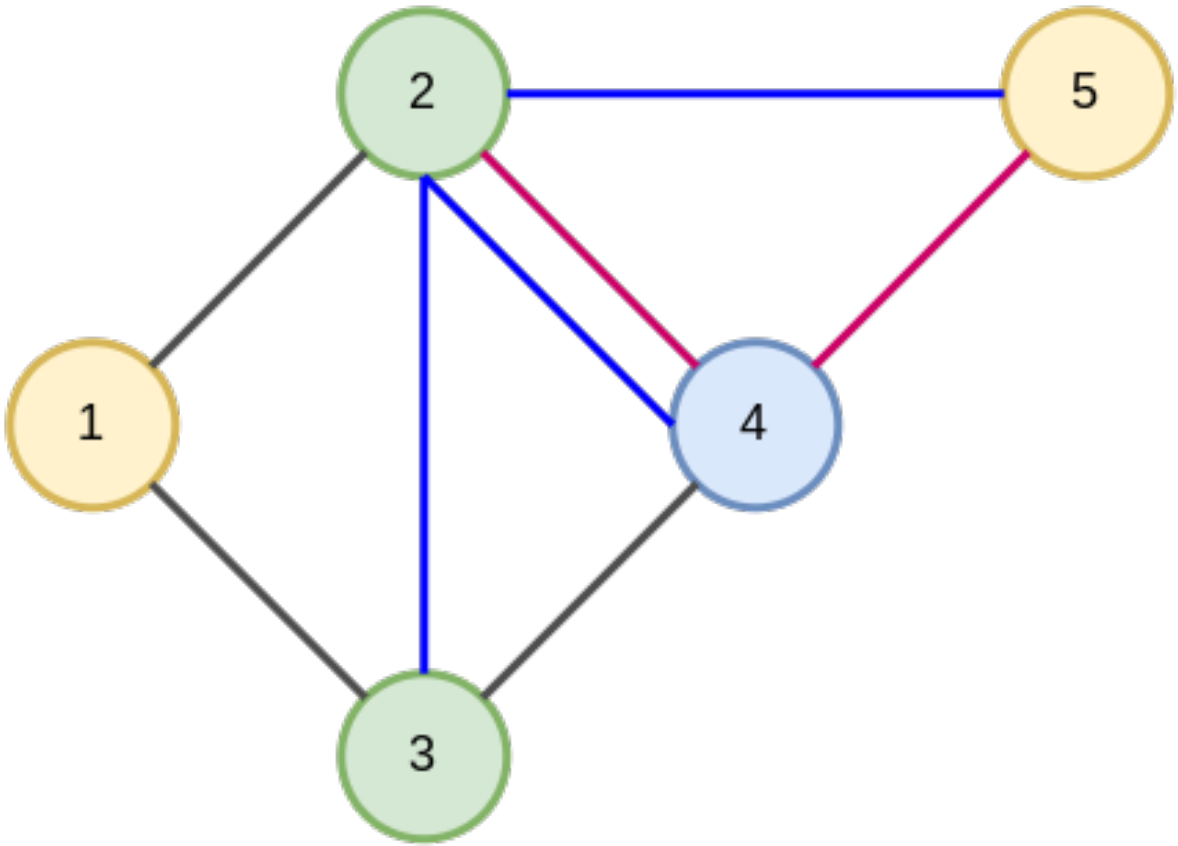
Пример: цитирование статей, ссылки в веб-страницах.

Мультиграф (рёбра разных типов)



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

Гетерогенный граф (узлы разных типов)



Граф знаний: люди, города, страны, события, компании...

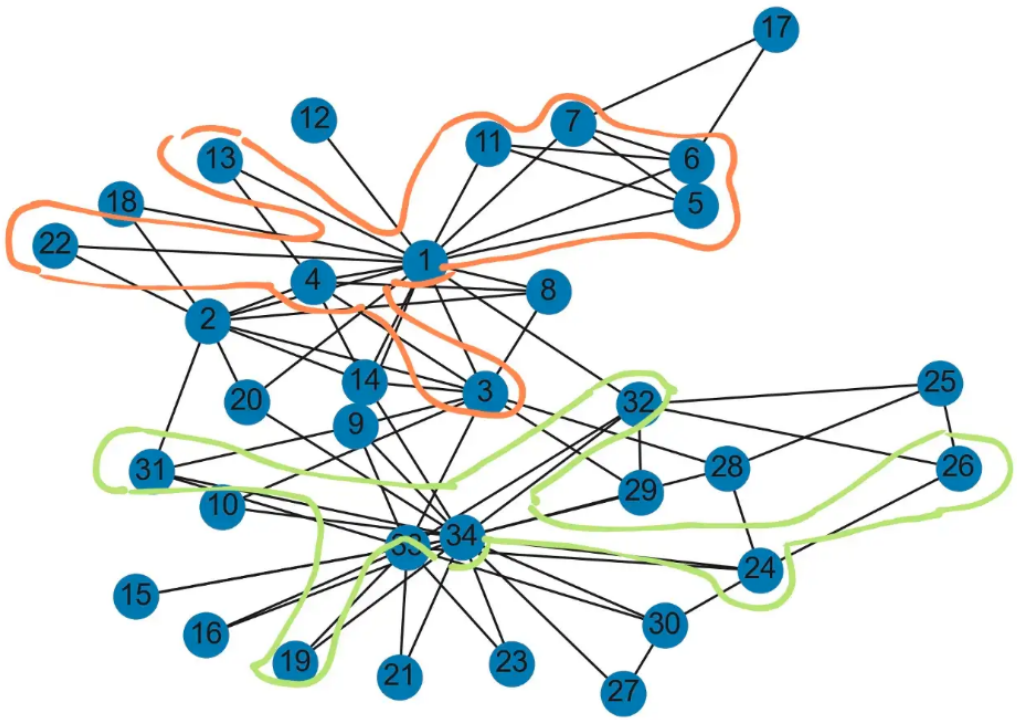
Инструменты

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

- Коллекция графовых датасетов - Open Graph Benchmark

Содержание

DeepWalk[3]



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

[3] https://arxiv.org/pdf/1403.6652.pdf

DeepWalk

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

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

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

Node2vec[4]

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

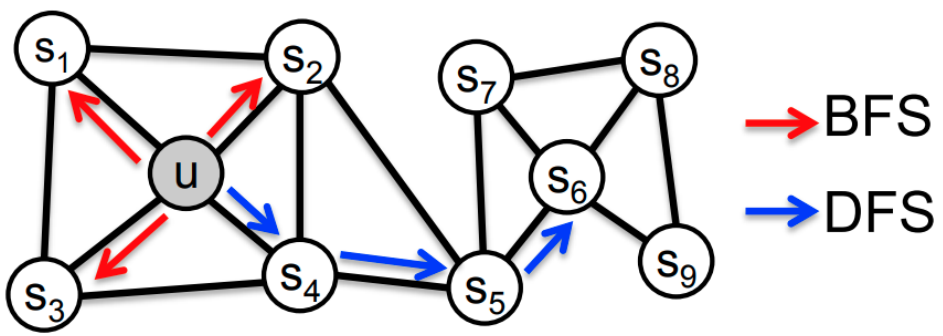


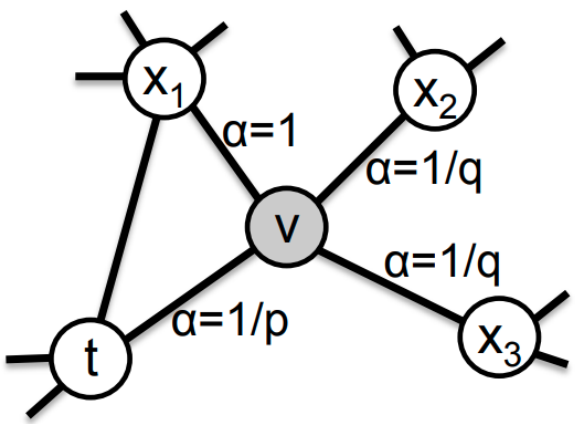
Figure 1: BFS and DFS search strategies from node u ($k = 3$).

[4] https://arxiv.org/pdf/1607.00653.pdf

Node2vec

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

- 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 эмбеддингов цветом)

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

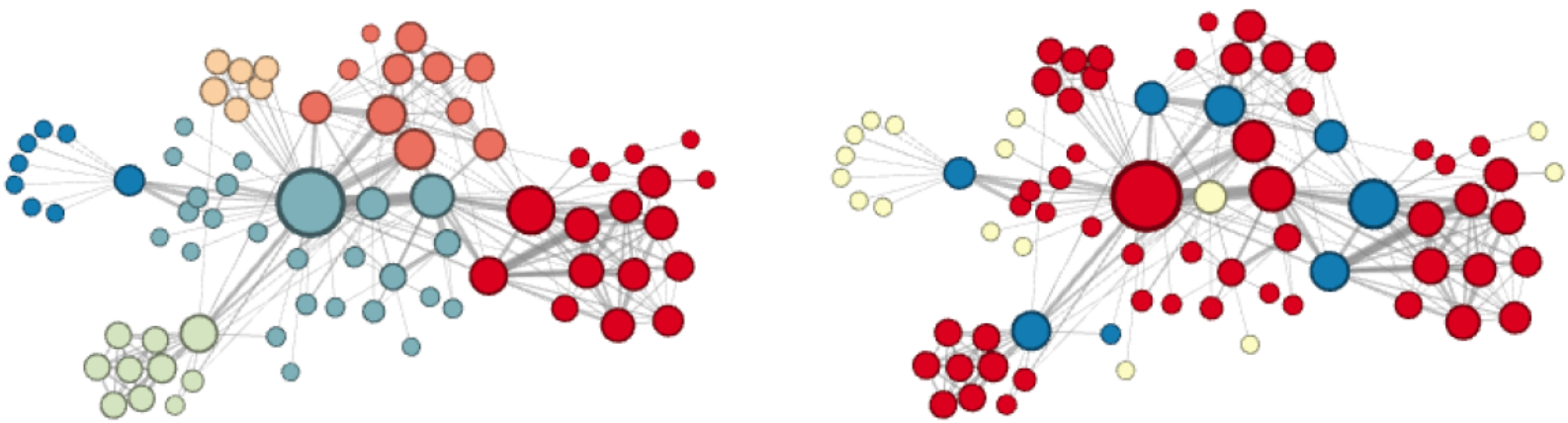
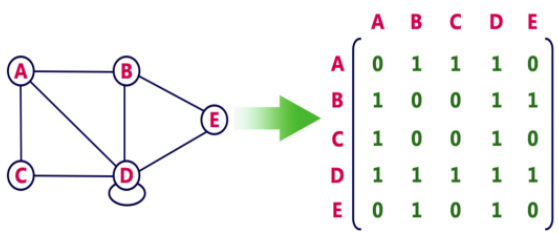
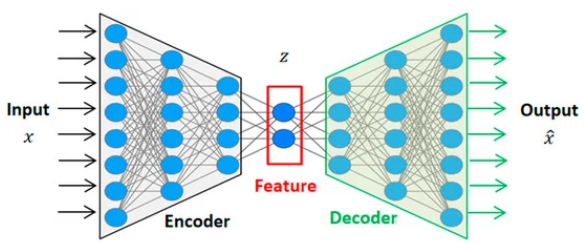


Figure 3: Complementary visualizations of Les Misérables coappearance network generated by *node2vec* with label colors reflecting homophily (top) and structural equivalence (bottom).

Эмбеддинги через автокодировщик[5]

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





[5] https://www.kdd.org/kdd2016/papers/files/rfp0191-wangAemb.pdf

Содержание

Кодирование вершин

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

Эмбеддинг уточняется по расположению на графе.

- (алгоритм передачи сообщений)

Прогноз должен быть инвариантен к перенумерации

$$\hat{y}(H) \equiv \hat{y}(H \cdot P)$$

где P - матрица перестановок вершин.

Задачи на графе целиком

Регрессия:

$$\hat{y}(G) = w_0 + \mathbf{w}^T \cdot H \cdot \mathbf{i}/N$$

Бинарная классификация:

$$\hat{y}(G) = \sigma(w_0 + \mathbf{w}^T \cdot H \cdot \mathbf{i}/N)$$

Многоклассовая классификация:

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

- $\sigma(u) = 1/(1+e^{-u})$ - сигмоида, $\mathbf{i} = [1,1,...1]^T \in \mathbb{R}^D$.
- параметры: $\mathbf{w}_0 \in \mathbb{R}^C$, $W \in \mathbb{R}^{C \times D}$

Задачи на отдельной вершине

Регрессия:

$$\hat{y}(G) = w_0 + \mathbf{w}^T \cdot H$$

Бинарная классификация:

$$\hat{y}(G) = \sigma(w_0 + \mathbf{w}^T \cdot H)$$

Многоклассовая классификация:

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

- $\sigma(u) = 1/(1+e^{-u})$ - сигмоида, $\mathbf{i} = [1, 1, ...1]^T \in \mathbb{R}^D$.
- параметры: $\mathbf{w}_0 \in \mathbb{R}^C$, $W \in \mathbb{R}^{C \times D}$

Восстановление связей

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

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

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

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

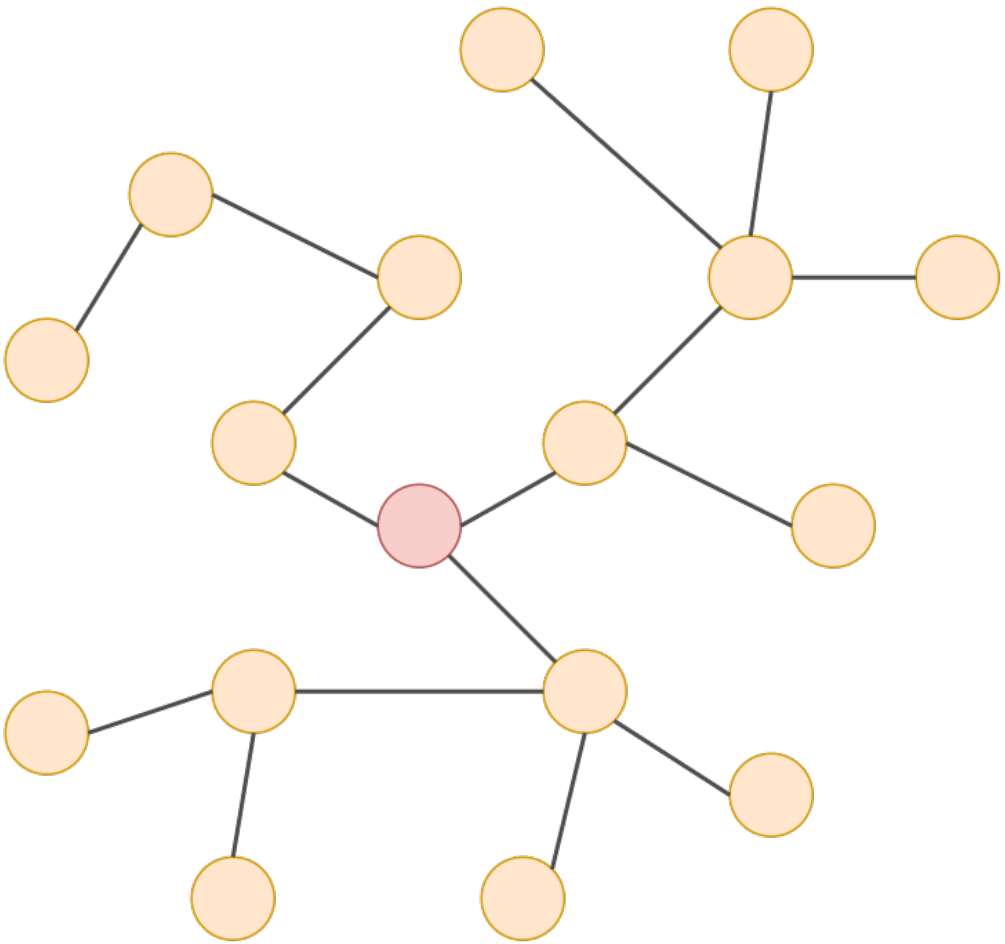
Симметричные W можно перебирать, подбирая нижнетреугольную L:

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

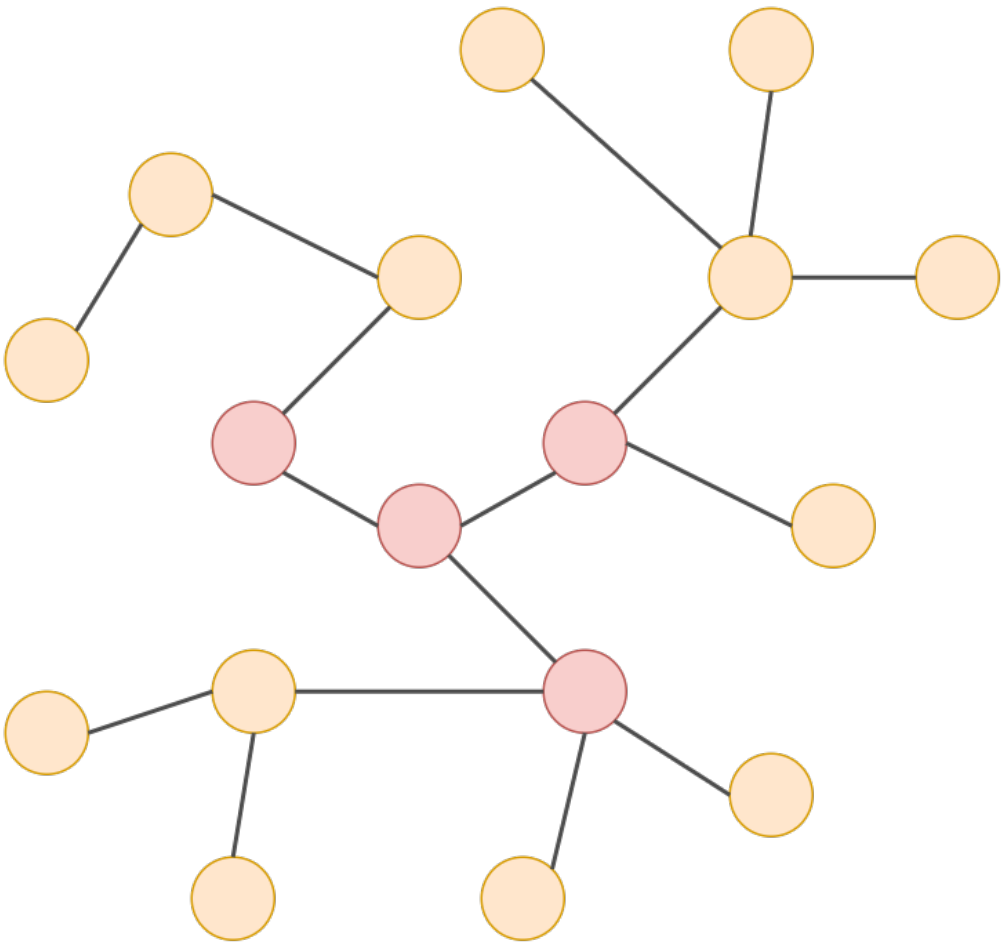
Более общий вариант:

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

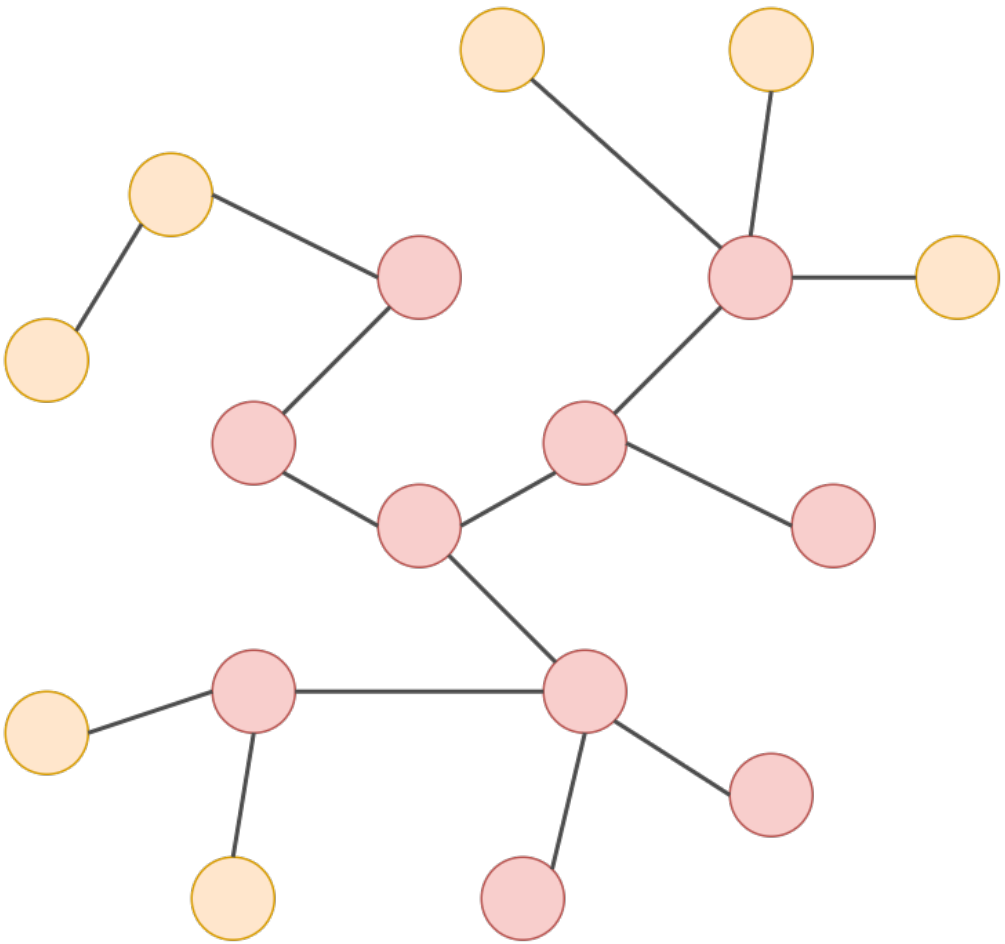
Алгоритм передачи сообщений



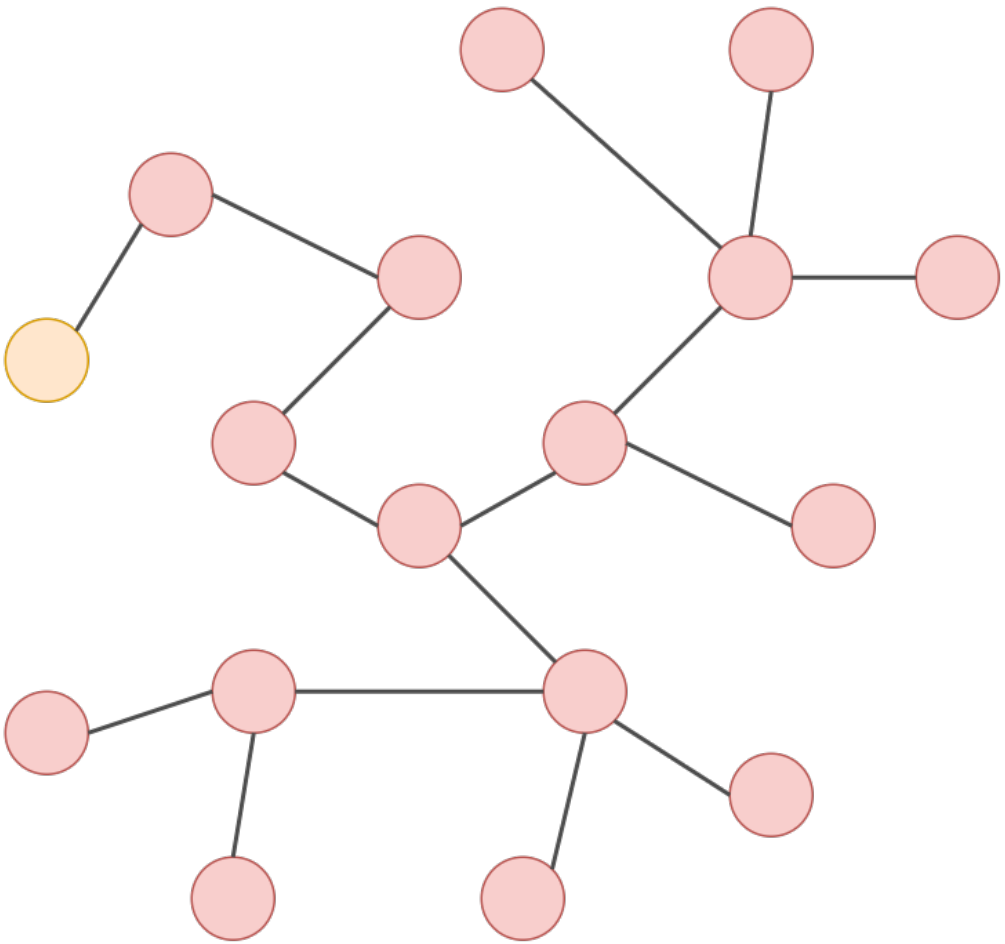
Алгоритм передачи сообщений



Алгоритм передачи сообщений



Алгоритм передачи сообщений



Алгоритм передачи сообщений[6]

ВХОД:

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

АЛГОРИТМ:

для $k = 1, 2, ...K$:

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

ВЫХОД:

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

[6] Message passing neural network.

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

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

Предотвращает сглаживание эмбеддингов в рез-те многократных усреднений.

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

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:

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

Примеры внимания

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

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

Функция Update

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

$f(\cdot)$ - 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 эмбеддингами

- За счёт многократных усреднений эмбеддинги слгаживаются.
- Способы борьбы:
  - ResNet блоки:

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

  - использование в конце эмбеддингов со всех слоёв:

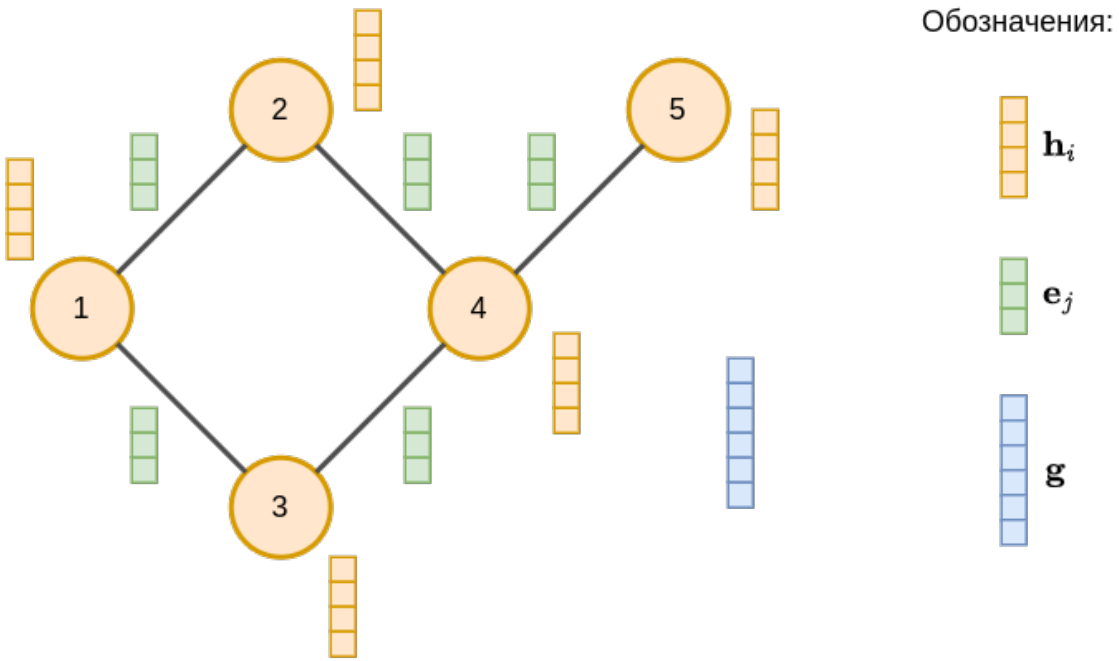
$$\mathbf{h}_n^{\text{final}} = [\mathbf{h}_n^1; \mathbf{h}_n^1; ...\mathbf{h}_n^K]$$

Эмбеддинги

- Помимо эмбеддингов вершин $v_i \to \mathbf{h}_i$ могут быть эмбеддинги рёбер

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

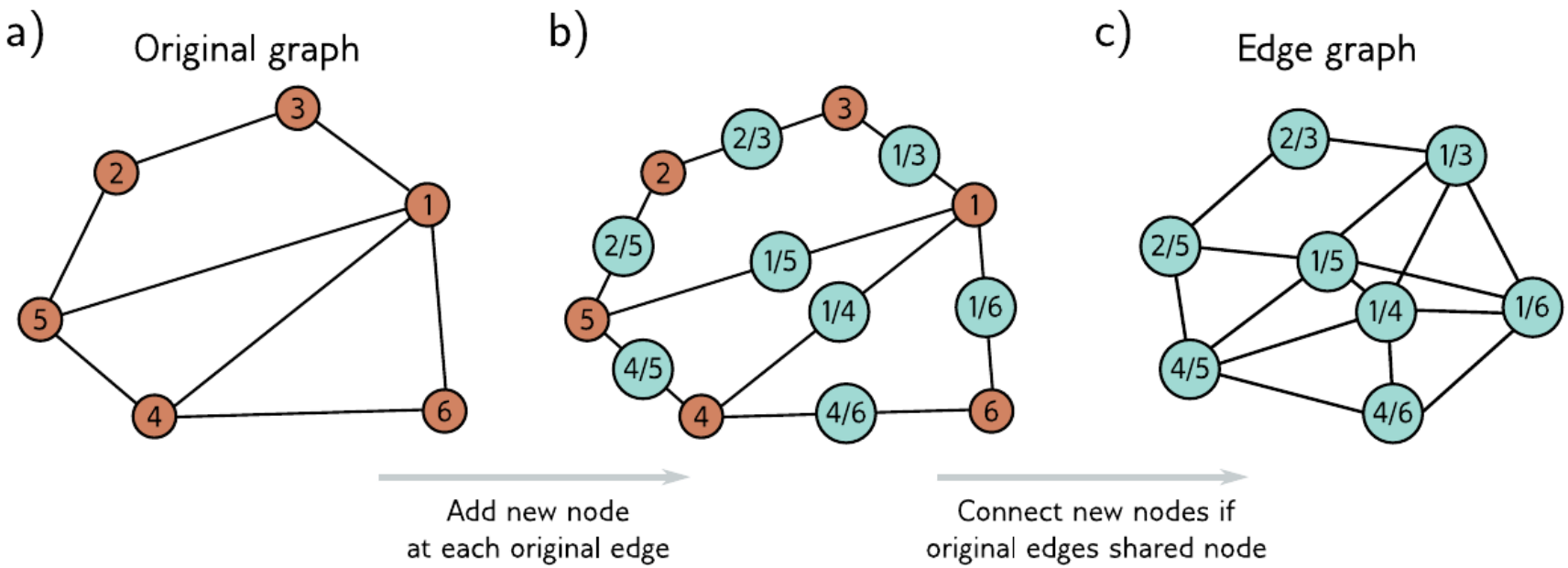
и эмбеддинг графа $\mathbf{g}$:



Задачи на рёбрах

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

- рёбра -> рёбра
- эмбеддинги рёбер -> эмбеддинги узлов



Учёт эмбеддингов рёбер и всего графа

ВХОД:

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

АЛГОРИТМ:

для $k = 1, 2, ...K$:

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

ВЫХОД:

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

Инвариантность к перенумерациям

Итерации алгоритма обмена сообщениями:

$$H^1 = F\left(X, A, W^1\right)$$
$$H^2 = F\left(H^1, A, W^2\right)$$
$$\dots$$
$$H^K = F\left(H^K, A, W^K\right)$$

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

Должно быть выполнено условие эквивариантности к перенумеровкам[7], заданным матрицей перестановок P.

[7] Запишите шаг передачи сообщений в матричном виде и покажите, что эквивариантность выполнена, учитывая правила $PP^T = P^TP = I$, $H' = HP$, $A' = P^TAP$.

Дополнительная информация

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