

Графы в машинном обучении

Романов Владимир БПМИ192

Национальный исследовательский университет
«Высшая школа экономики» (Москва)

23 ноября 2021 г.

Vertex embeddings

Определение

First-order proximity — непосредственная близость вершин

Second-order proximity — близость вершин по структуре

Рассмотрим embedding, которые описывают локальные свойства

Мы хотим, чтобы соседи имели близкий вектор, а далекие разные

Формулировка напоминает аналогичную задачу для текста

Идея: попытаемся применить SkipGram для этой задачи

DeepWalk (Постановка)

Обозначения

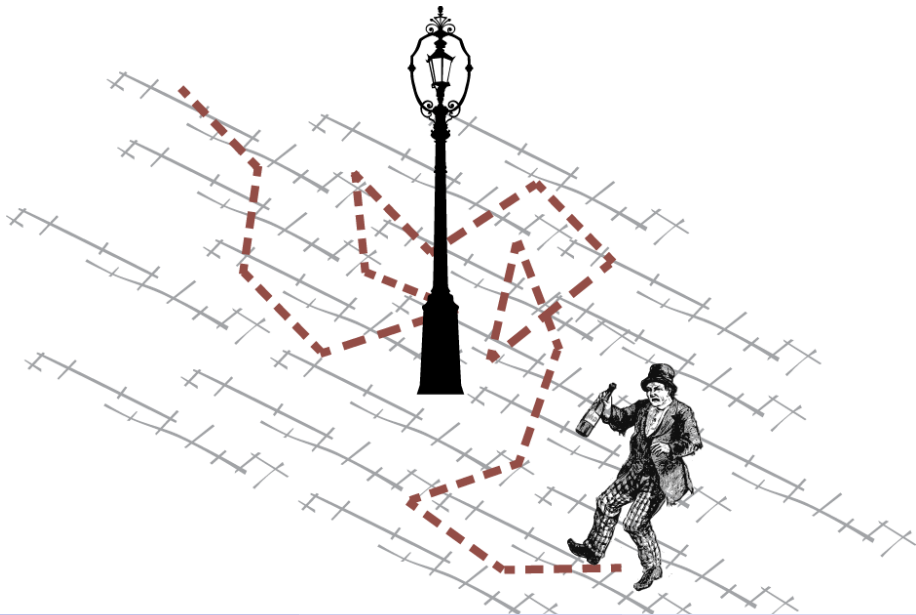
Embedding: $\Phi : V \rightarrow \mathbb{R}^d$

Loss: $\sum_{u \in N(v)} -\ln \Pr(u \mid \Phi(v))$

Алгоритм

- 1 Переберем вершину $s \in V$
- 2 Рассмотрим случайный путь \mathcal{W}_s размера t
- 3 Прорелаксируем SkipGram пройдясь по \mathcal{W}_s окном w
- 4 Повторим процесс γ раз

DeepWalk (in a nutshell)



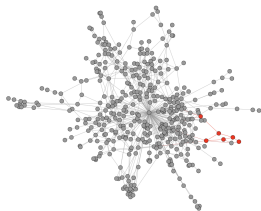
DeepWalk (Hierarchical Softmax)

Проблема: делать Softmax долго

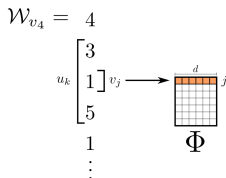
Решение: дерево отрезков (или negative sampling)

$$Pr(u | \Phi(v)) = \prod_{b_i \in B} Pr(b_i | \Phi(v))$$

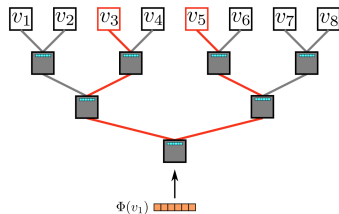
Итоговый алгоритм:



(a) Random walk generation.



(b) Representation mapping.



(c) Hierarchical Softmax.

Node2Vec (search strategy)

Обновление

$$\text{Loss: } \sum_{u \in N_S(v)} -\ln \Pr(u \mid \Phi(v))$$

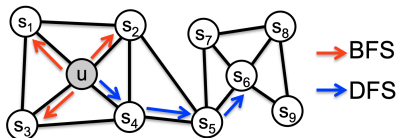


Figure 1: BFS and DFS search strategies from node u ($k = 3$).

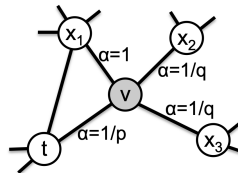


Figure 2: Illustration of the random walk procedure in *node2vec*. The walk just transitioned from t to v and is now evaluating its next step out of node v . Edge labels indicate search biases α .

Примеры кластеризации с разными параметрами q



Рис.: $p = 1, q = 0.5$

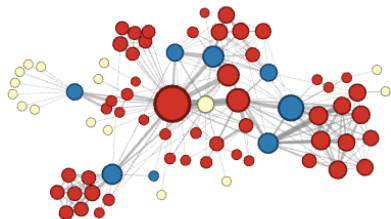
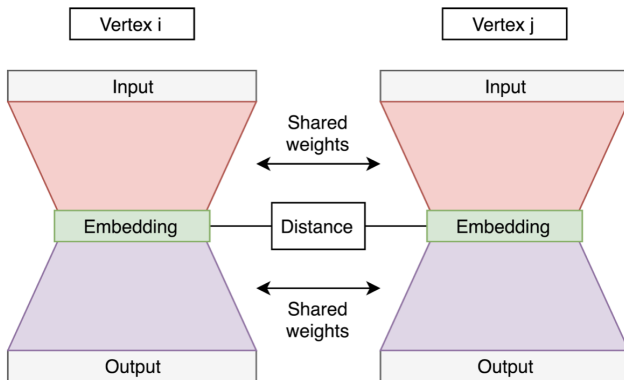


Рис.: $p = 1, q = 2$

Structural Deep Network Embedding (SDNE)

Хотим в embedding также «хранить» знания о структуре (second-order proximity)



Какие задачи мы можем решить?

Возможные формулы оценки:

Operator	Symbol	Definition
Average	\boxplus	$[f(u) \boxplus f(v)]_i = \frac{f_i(u) + f_i(v)}{2}$
Hadamard	\boxdot	$[f(u) \boxdot f(v)]_i = f_i(u) * f_i(v)$
Weighted-L1	$\ \cdot\ _{\bar{1}}$	$\ f(u) \cdot f(v)\ _{\bar{1}i} = f_i(u) - f_i(v) $
Weighted-L2	$\ \cdot\ _{\bar{2}}$	$\ f(u) \cdot f(v)\ _{\bar{2}i} = f_i(u) - f_i(v) ^2$

Примеры задач:

- node-classification
- node-clustering
- graph-visualization
- link-prediction

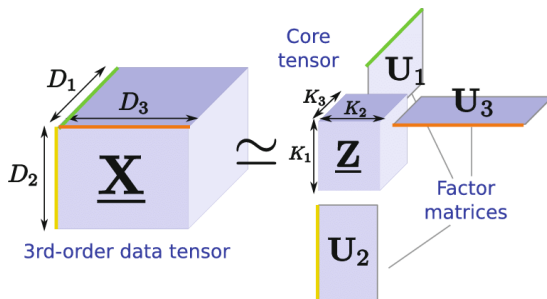
Усложнение link-prediction

Постановка:

- Есть граф (V, E)
- Каждое ребро принадлежит одному из классов (R)
- Хотим найти $\phi(v_1, r, v_2)$, которая будет предсказывать, есть ли ребро (v_1, v_2) класса r

Мы хотим предсказать трехмерный бинарный тензор $\mathbb{R}^{|V| \times |E| \times |V|}$

TuckER (decomposition)



$$X_{ijk} = \sum_{\alpha=1}^{K_1} \sum_{\beta=1}^{K_2} \sum_{\gamma=1}^{K_3} Z_{\alpha\beta\gamma} U_{i\alpha}^{(1)} U_{j\beta}^{(2)} U_{k\gamma}^{(3)}$$

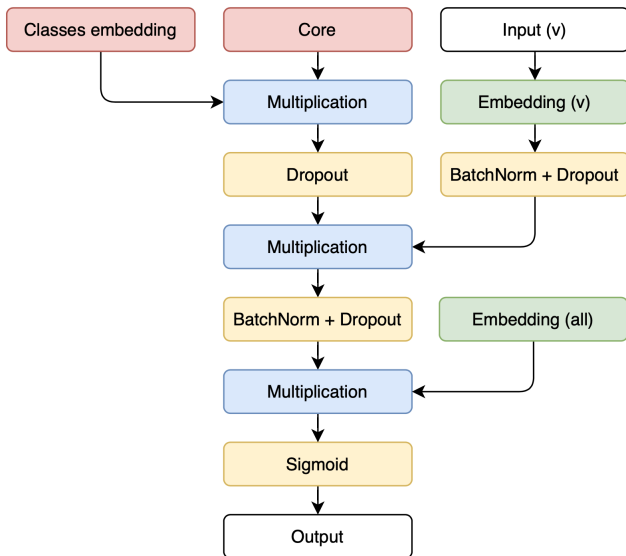
TuckER (model)

Попытаемся воспользоваться полученными embeddings для вершин

Первая идея: обучить модель для каждого класса

Вторая идея: обучим для классов свои embeddings, параллельно обучая ядро свертки

TuckER (model)



- DeepWalk: <https://arxiv.org/pdf/1403.6652.pdf>
- Node2Vec: <https://arxiv.org/pdf/1607.00653.pdf>
- SDNE: <https://www.kdd.org/kdd2016/papers/files/rfp0191-wangAemb.pdf>
- TuckER: <https://arxiv.org/pdf/1901.09590.pdf>