

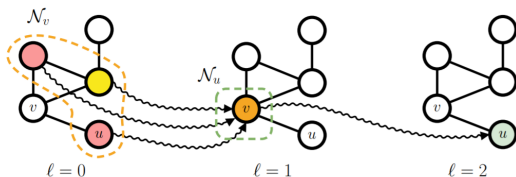
# Свёрточные нейронные сети на графах

Александр Колодезный БПМИ192

Национальный исследовательский университет  
«Высшая школа экономики» (Москва)

23 ноября 2021 г.

# Graph Convolutional Network



Вид одного свёрточного слоя

$$\mathbf{h}_v^{\ell+1} = \phi^{\ell+1} \left( \mathbf{h}_v^{\ell}, \Psi(\{\psi^{\ell+1}(\mathbf{h}_u^{\ell}) \mid u \in \mathcal{N}_v\}) \right)$$

- ▶  $\Psi$  — permutation-invariant функция
- ▶  $\phi^{l+1}$  и  $\psi^{l+1}$  — некоторый функция на  $l$ -ом слое

# Graph Convolutional Network

Model	Neighborhood Aggregation $\mathbf{h}_v^{\ell+1}$
NN4G [88]	$\sigma\left(\mathbf{w}^{\ell+1^T} \mathbf{x}_v + \sum_{i=0}^{\ell} \sum_{c_k \in \mathcal{C}} \sum_{u \in \mathcal{N}_v^{c_k}} w_{c_k}^i * \mathbf{h}_u^i\right)$
GNN [104]	$\sum_{u \in \mathcal{N}_v} MLP^{\ell+1}\left(\mathbf{x}_u, \mathbf{x}_v, \mathbf{a}_{uv}, \mathbf{h}_u^{\ell}\right)$
GraphESN [44]	$\sigma\left(\mathbf{W}^{\ell+1} \mathbf{x}_u + \hat{\mathbf{W}}^{\ell+1}[\mathbf{h}_{u_1}^{\ell}, \dots, \mathbf{h}_{u_{\mathcal{N}_v}}^{\ell}]\right)$
GCN [72]	$\sigma\left(\mathbf{W}^{\ell+1} \sum_{u \in \mathcal{N}(v)} \mathbf{L}_{vu} \mathbf{h}_u^{\ell}\right)$
GAT [120]	$\sigma\left(\sum_{u \in \mathcal{N}_v} \alpha_{uv}^{\ell+1} * \mathbf{W}^{\ell+1} \mathbf{h}_u\right)$
ECC [111]	$\sigma\left(\frac{1}{ \mathcal{N}_v } \sum_{u \in \mathcal{N}_v} MLP^{\ell+1}(\mathbf{a}_{uv})^T \mathbf{h}_u^{\ell}\right)$
R-GCN [105]	$\sigma\left(\sum_{c_k \in \mathcal{C}} \sum_{u \in \mathcal{N}_v^{c_k}} \frac{1}{ \mathcal{N}_v^{c_k} } \mathbf{W}_{c_k}^{\ell+1} \mathbf{h}_u^{\ell} + \mathbf{W}^{\ell+1} \mathbf{h}_v^{\ell}\right)$
GraphSAGE [54]	$\sigma\left(\mathbf{W}^{\ell+1}\left(\frac{1}{ \mathcal{N}_v }[\mathbf{h}_v^{\ell}, \sum_{u \in \mathcal{N}_v} \mathbf{h}_u^{\ell}]\right)\right)$
CGMM [3]	$\sum_{i=0}^{\ell} w^i * \left(\sum_{c_k \in \mathcal{C}} w_{c_k}^i * \left(\frac{1}{ \mathcal{N}_v^{c_k} } \sum_{u \in \mathcal{N}_v^{c_k}} \mathbf{h}_u^i\right)\right)$
GIN [131]	$MLP^{\ell+1}\left((1 + \epsilon^{\ell+1}) \mathbf{h}_v^{\ell} + \sum_{u \in \mathcal{N}_v} \mathbf{h}_u^{\ell}\right)$

# Обработка рёбер

- ▶ Рёбра в графе могут иметь дополнительную информацию
- ▶ Можно брать различные веса для разных видов рёбер

$$\mathbf{h}_v^{\ell+1} = \phi^{\ell+1}\left(\mathbf{h}_v^{\ell}, \sum_{c_k \in \mathcal{A}} (\Psi(\{\psi^{\ell+1}(\mathbf{h}_u^{\ell}) \mid u \in \mathcal{N}_v^{c_k}\}) * w_{c_k})\right),$$

- ▶ Более общий вид, если у рёбер есть свои признаки

$$\mathbf{h}_v^{\ell+1} = \phi^{\ell+1}\left(\mathbf{h}_v^{\ell}, \Psi(\{e^{\ell+1}(\mathbf{a}_{uv})^T \psi^{\ell+1}(\mathbf{h}_u^{\ell}) \mid u \in \mathcal{N}_v\})\right),$$

# Attention

- ▶ Добавление дополнительных обучаемых весов на рёбра  $\alpha_{uv}^{\ell+1}$

$$\mathbf{h}_v^{\ell+1} = \phi^{\ell+1} \left( \mathbf{h}_v^\ell, \Psi(\{\alpha_{uv}^{\ell+1} * \psi^{\ell+1}(\mathbf{h}_u^\ell) \mid u \in \mathcal{N}_v\}) \right),$$

- ▶ Вычисляем сначала коэффициенты  $w_{uv}$

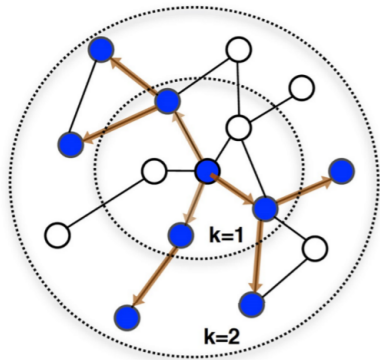
$$a(\mathbf{W}^\ell \mathbf{h}_u^\ell, \mathbf{W}^\ell \mathbf{h}_v^\ell) = \text{LeakyReLU}((\mathbf{b}^\ell)^T [\mathbf{W}^\ell \mathbf{h}_u^\ell, \mathbf{W}^\ell \mathbf{h}_v^\ell]),$$

- ▶ Считаем softmax

$$\alpha_{uv}^\ell = \frac{\exp(w_{uv}^\ell)}{\sum_{u' \in \mathcal{N}_v} \exp(w_{u'v}^\ell)}.$$

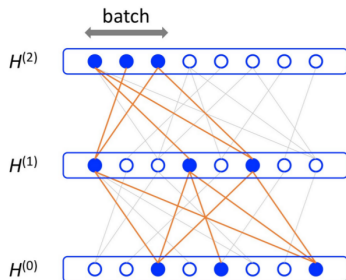
# Sampling (GraphSAGE)

- ▶ Проблема — много вычислений на плотных графах
- ▶ Для каждой вершины на каждом слое выбираем случайное подмножество соседних вершин.
- ▶ Пересчитываем  $h_v$  только от  $h_u$ , которые выбрали.
- ▶ Приходится пересчитывать градиенты для всех вершин.

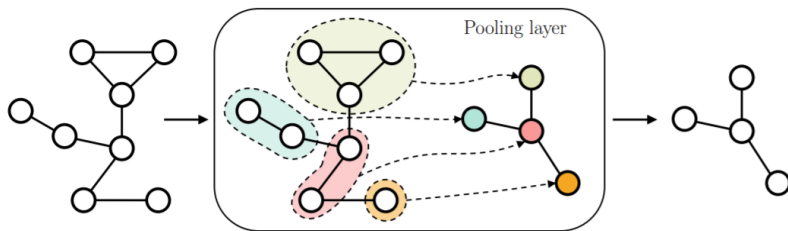


# Sampling (FastGCN)

- ▶ На каждом слое выбираем  $t$  вершин
- ▶ При вычислении  $h_v$  используем только соседей, выбранных на этом слое.



# Pooling



- ▶ Сжатие кластеров похожих вершин в одну
- ▶ Уменьшает размеры графа
- ▶ Уменьшает вычислительную стоимость



# Differentiable Pooling

- ▶ Обучаем для каждой вершины попадание в кластера

$$\mathbf{S}^{\ell+1} = \text{softmax}(\text{GNN}(\mathbf{A}^{\ell}, \mathbf{H}^{\ell})),$$

- ▶ Пересчитываются embedding для новых вершин, и новая матрица смежности

$$\mathbf{H}^{\ell+1} = \mathbf{S}^{\ell+1T} \mathbf{H}^{\ell} \quad \text{and} \quad \mathbf{A}^{\ell+1} = \mathbf{S}^{\ell+1T} \mathbf{A}^{\ell} \mathbf{S}^{\ell+1}.$$

- ▶ Новая матрица смежности оказывается полной

# Top-k Pooling

- ▶ Для каждой вершины посчитаем его вес как проекция его эмбединга на обучаемый вектор  $p$

$$s^{\ell+1} = \frac{\mathbf{H}^{\ell} p^{\ell+1}}{\|p^{\ell+1}\|}.$$

- ▶ Выбираем  $k$  вершин с наибольшим полученным весом
- ▶ Оставляем в графе только найденные вершины
- ▶ Расширение метода Self-attention pooling

$$s^{\ell+1} = \sigma(\text{GCN}(\mathbf{A}^{\ell}, \mathbf{H}^{\ell})).$$

# Edge Pooling

- ▶ Считаем вес для рёбер

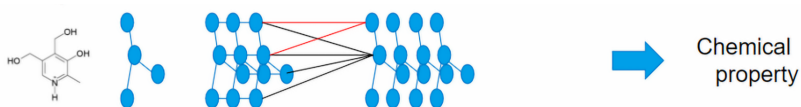
$$s^{\ell+1}((v, u) \in \mathcal{E}_g) = \sigma(\mathbf{w}^T [\mathbf{h}_v^\ell, \mathbf{h}_u^\ell] + \mathbf{b}).$$

- ▶ Сжимаем две вершины, соединённые этим ребром в одну
- ▶ Повторяем итерационно

# Топологические pooling-и

- ▶ GRACCLUS — алгоритм основанный на спектральном анализе матрицы смежности
- ▶ Non-negative Factorization Matrix Pooling — pooling основанный на NFM факторизации матрицы смежности.

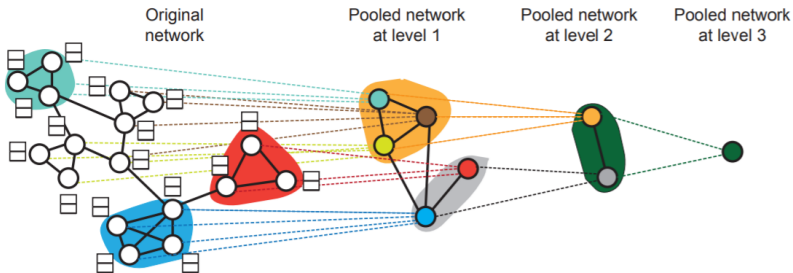
# Graph embedding



- ▶ Рассматриваем в задаче граф целиком
- ▶ Хотим построить embedding для всего графа

# Graph embedding

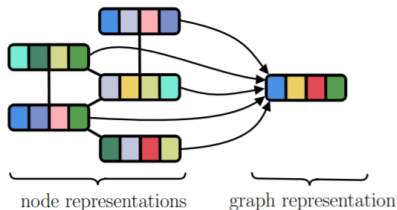
- ▶ Делаем pooling пока не останется одна вершина
- ▶ Оставшаяся вершина хранит информацию обо всём графе

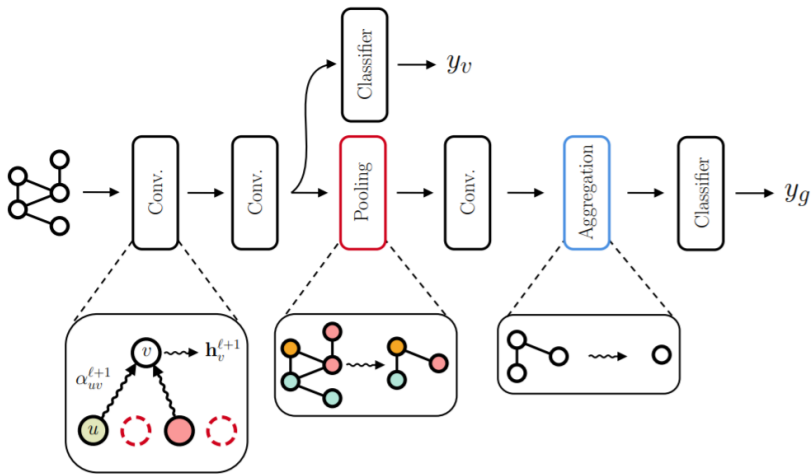


# Graph embedding

$$\mathbf{h}_g^\ell = \Psi\left(\{f(\mathbf{h}_v^\ell) \mid v \in \mathcal{V}_g\}\right),$$

- ▶ Выбрать  $\Psi$  как сумму, максимум или минимум, а  $f$  — тождественную функцию.
- ▶  $f$  как нейронная сеть







# Список литературы

- ▶ <https://arxiv.org/pdf/1912.12693.pdf>
- ▶ <https://arxiv.org/pdf/1609.02907.pdf>
- ▶ <https://arxiv.org/pdf/1812.04202.pdf>
- ▶ <https://arxiv.org/pdf/1801.10247.pdf>
- ▶ <https://arxiv.org/pdf/1706.02216.pdf>