Нейронные сети с рекуррентной архитектурой на графах

Иваник Даниил, БПМИ192

Национальный исследовательский университет "Высшая школа экономики"

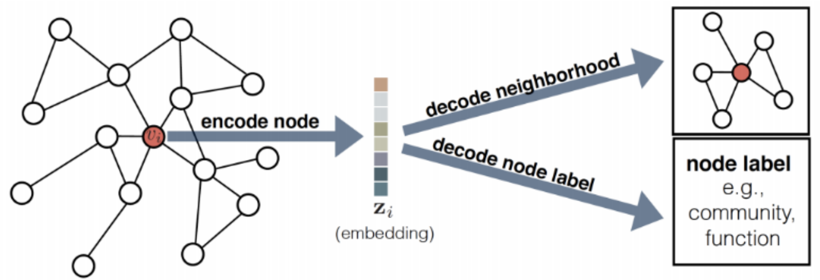
.

23 ноября, 2021

Проблемы глубокого обучения на графах

- Графы имеют переменное число вершин
- Изначальные признаки вершины в графе могут не учитывать структуру графа (признаков может не быть вовсе)
- Не очевидно, как учесть структуру графа в привычном числовом виде
- Часто не существует естественного порядка вершин, поэтому бывает непонятно, в каком порядке проводить вычисления

Решение: последовательный подсчёт эмбедингов в каждой вершине, использующий структуру графа

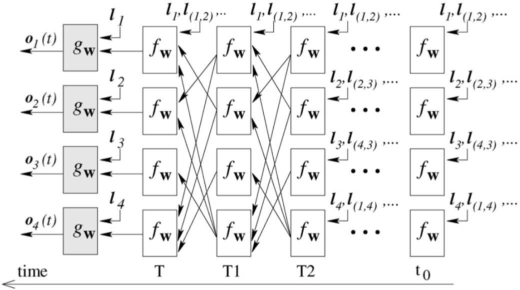


Изначально дан граф, признаки в каждой вершине x_u и признаки на рёбрах z_{uv}

Дальше мы будем считать эмбеддинги, исходные признаки в течении задачи не меняются!

Общий вид процесса в GNN

Обучаем сеть, которая похожа на рекуррентную.



Есть большие шаги обучения

Внутри каждого шага считаем эмбеддинги используя f_w

Считаем выходную функцию с помощью g_w

f_w, g_w - обучаемые модели машинного обучения (обучаем w)

Подсчёт эмбеддингов в Graph Neural Network (внутри одного большого шага)

Общий вид рекуррентного соотношения итерационном процессе:

$$h_u^{(n)} = \Psi_{v \in N(u)}(f_w(x_u, x_v, z_{uv}, h_v^{(n-1)}))$$

Ψ - permutation-invariant функция с переменным числом параметров. В примере с GNN будем использовать только Σ

f_w - это функция, общая для всех вершин и всех шагов

Проводим такую операцию до тех пор, пока $h_u^{(n)}$ не перестанут меняться

О сходимости итерационного процесса

Для сходимости эмбеддингов потребуем, чтобы шаг итерационного процесса являлся сжимающим отображением Пусть $X \subset \mathbb{R}^n$ и $\phi : X \to X$. ϕ называется сжимающим, если для какого-то $0 < \mu < 1$ и любых $y, z \in X$:

$$\|\phi(y) - \phi(z)\| < \mu\|y - z\|)$$

Теорема Банаха:
Сжимающее отображение имеет ровно одну неподвижную точку p, при этом для любого x: $\phi^n(x)$ сходится к p с экспоненциальной скоростью.

Идея Linear GNN

Хотим построить относительно гибкую и простую архитектуру, которая будет гарантировать свойство сжимающего отображения

Потребуем следующее свойство:

$$f_w(h_u^{(n-1)}, x_u, x_v, z_{uv}) = A_w(x_u, x_v, z_{uv})h_u^{(n-1)} + b_w(x_u)$$

На b_w не накладывается никаких специфических ограничений

Далее s - размер эмбеддингов h, $0 < \mu < 1$

$$A_w(x_u, x_v, z_{uv}) = A_{u,v} = \frac{\mu}{s|N(v)|} reshape(tanh(P_w(x_u, x_v, z_{uv})), (s, s))$$

P - любая модель.

$g_w(h_u^{(T)}, x_u)$ - тоже может быть любой.


OMB
MISS ME?

Вспомогательные факты

Факт 1.

Пусть B - квадратная блочная матрица с квадратными блоками. Тогда:

$$\|B\|_1 \leq \max_v \sum_u \|B_{u,v}\|_1$$

Факт 2.

Пусть A - квадратная блочная матрица, $\|A\|_1 < 1$. Тогда, отображение $x \to Ax + b$ - сжимающее относительно первой нормы.

Доказательство

H - конкатенация всех эмбеддингов ($H = [h_1^t, \dots, h_n^t]^t$)
Тогда, можно заметить что $H^{(n)} = AH^{(n-1)} + b$
Пусть A - блочная матрица размера $N \times N$ и

$$B_{u,v} = reshape(tanh(P_w(x_u, x_v, z_{uv})), (s, s)), (u, v) \in E$$

$$A_{u,v} = \frac{\mu}{s|N(v)|} B_{u,v}$$

$$A_{u,v} = 0, (u, v) \notin E$$

$$\|A\|_1 \leq max_v \sum_u \|A_{u,v}\|_1 \leq max_v \sum_{u \in N(v)} \|A_{u,v}\|_1 \leq$$

$$\mu max_{u,v} \frac{\|B_{u,v}\|}{s} \leq \mu$$

Пара слов о нелинейной GNN

Чтобы приблизить F_w к сжимающему отображению, введём регуляризацию:

$$L_{new} = L_{old} + \beta L(\|\frac{\partial F_w}{\partial H}\|)$$

Где $L(y) = (y - \mu)^2$ при $y > \mu$ и 0 иначе.
Смысл: штрафуем модель, пока она не находится в области, где наше отображение является сжимающим

Преимущества и недостатки GNN

Преимущества:

- Гибкая архитектура (переменное число вершин и слоёв)
- Используя Linear GNN с правильными параметрами, получаем теоретическую гарантию сходимости
- Не нужно думать про инициализацию

Недостатки

- Затухание градиентов
- Время обучения(однако для подсчёта градиентов используется алгоритм Almeida-Pineda)

Graph-Gated Neural Network

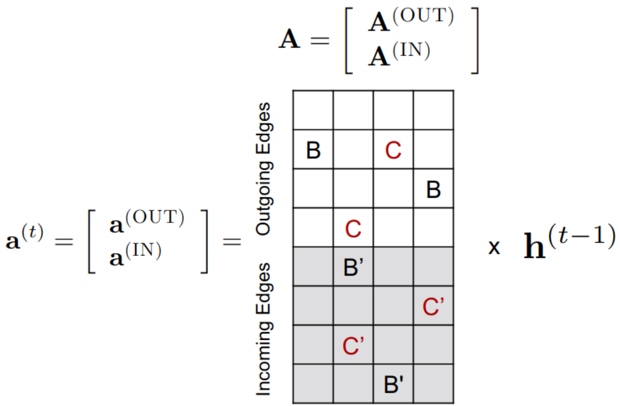
За основу взята Linear-GNN.

$$h_v^{(0)} = [x_v^T, 0]^T$$

Эмбеддинги пересчитываются следующим образом:

$$\begin{aligned}
\mathbf{a}^{(t)} &= \mathbf{A}\mathbf{h}^{(t-1)} + \mathbf{b} \\
\mathbf{r}_v^t &= \sigma\left(\mathbf{W}^r\mathbf{a}_v^{(t)} + \mathbf{U}^r\mathbf{h}_v^{(t-1)}\right) \\
\mathbf{z}_v^t &= \sigma\left(\mathbf{W}^z\mathbf{a}_v^{(t)} + \mathbf{U}^z\mathbf{h}_v^{(t-1)}\right) \\
\widetilde{\mathbf{h}_v^{(t)}} &= \tanh\left(\mathbf{W}\mathbf{a}_v^{(t)} + \mathbf{U}\left(\mathbf{r}_v^t \odot \mathbf{h}_v^{(t-1)}\right)\right) \\
\mathbf{h}_v^{(t)} &= (1 - \mathbf{z}_v^t) \odot \mathbf{h}_v^{(t-1)} + \mathbf{z}_v^t \odot \widetilde{\mathbf{h}_v^{(t)}}
\end{aligned}$$

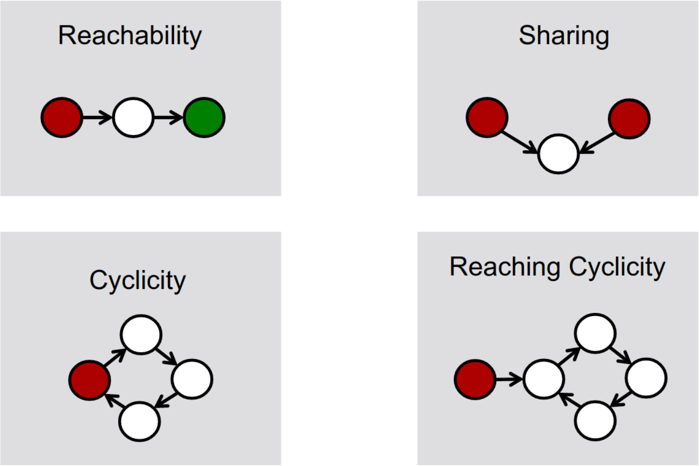
Матрица, которая используется для подсчёта эмбеддингов на следующем шаге



Отличия от GNN

- Не надо подбирать сжимающее отображение
- Важна инициализация. Можно передать какую-то особенную информацию на вход
- Количество шагов фиксировано, можно предугадать ресурсы для одного шага обучения
- Борется с затуханием градиентов

Задачи, для которых используется GG-NN



Источники

- https://arxiv.org/pdf/1912.12693.pdf
- https://persagen.com/files/misc/scarselli2009graph.pdf
- https://www.cs.toronto.edu/~yujiali/files/talks/iclr16_ggnn_talk.pdf
- https://arxiv.org/pdf/1511.05493.pdf
- https://en.wikipedia.org/wiki/Gated_recurrent_unit