ВЛОЖЕНИЕ СЛОЖНЫХ СЕТЕЙ В ГИПЕРБОЛИЧЕСКОЕ ПРОСТРАНСТВО С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

Хорошеньких С.Н. [1]
(Московский физико-технический институт, Москва)

В данной работе предлагается алгоритм поиска вложений графов в гиперболическое пространство с помощью суррогатов – искусственных графов, порождённых моделью случайного гиперболического графа. Из суррогатов формируется обучающая выборка, на которой тренируется модель, предсказывающая координаты вершин входного графа. Предлагаемый алгоритм находит вложения, которые позволяют находить близкие к оптимальным пути с помощью жадной маршрутизации. Также получаемые вложения подходят для задачи предсказания рёбер.

Ключевые слова: Граф, вложение, гиперболическая плоскость, машинное обучение.

1. Введение

Графы являются естественной математической моделью для так называемых «сложных систем», примерами которых являются Интернет, социальные и биологические сети. Законы взаимодействия частей «сложных систем» зачастую неизвестны, а потому их естественно моделировать в вероятностных терминах. Отсюда следует, что полезной моделью изучения сложных систем являются случайные графы [13]. Важным подклассом случайных графов являются случайные геометрические графы, имеющие следующую структурную особенность: вершинам соответствуют точки в некотором метрическом пространстве, а рёбра между вершинами проводятся в зависимости от расстояния между точками, которые соответствуют вершинам. В частности, значительный интерес представляет модель случайного гиперболического графа на диске Пуанкаре [10]. Данная модель отражает многие свойства

[1] Сергей Николаевич Хорошеньких (khoroshenkikh@phystech.edu).

сложных сетей: степенной закон распределения степеней вершин и асимптотически константный локальный кластерный коэффициент [7], транзитивность [4] и малый диаметр [9].

Геометрические модели графов интересны, в том числе, тем, что для них можно сформулировать своего рода «обратную задачу»: по заданному графу отобразить его вершины в соответствующее метрическое пространство так, чтобы обратное отображение имело максимальное правдоподобие в заданной модели. В более общем и менее формальном виде эта задача формулируется следующим образом: отобразить вершины графа в точки метрического пространства так, чтобы «близким» в графе парам вершин (например, тем парам, между которыми есть рёбра) соответствовали близкие пары точек метрического пространства, и наоборот. Для такого отображения обычно используется термин «вложение» (embedding). Задача поиска вложения представляет самостоятельный интерес и может быть решена без использования вероятностной модели графа. В частности, известны способы поиска вложения вершин в евклидово пространство, основанные на матричной факторизации и коротких случайных блужданиях по графу [8].

Поскольку гиперболические пространства позволяют вкладывать иерархические структуры более точно по сравнению с евклидовыми пространствами [6], естественным образом возникает задача построения алгоритма, который находит такое вложение. В [2, 16, 15, 1] предлагаются алгоритмы, основанные на максимизации правдоподобия входного графа для различных моделей случайных графов (включая модель случайного гиперболического графа [10]).

В данной работе предлагается новый подход к построению алгоритмов поиска вложений графов. В его основе лежит идея построения суррогатов – искусственных графов, порождённых моделью случайного гиперболического графа [10]. Эта модель описана в разделе 2.

Для входного графа подбираются такие параметры ги-

перболической модели, для которых макроскопические свойства суррогатов (распределение степеней вершин, средняя степень вершины и коэффициент кластеризации) оказываются наиболее близкими к свойствам исходного графа. В разделе 3 представлен способ оценки этих параметров.

Затем с помощью алгоритма, описанного в [12], строится несколько суррогатов, после чего на основе известных вложений для суррогатов с помощью машинного обучения строятся вложения для входного графа.

Из суррогатов и из входного графа извлекаются признаки для предсказания радиальных координат (см. раздел 4) и попарных расстояний между вершинами (см. раздел 6). Второе множество признаков извлекается не для всех пар вершин, поскольку это приведёт к квадратичной сложности алгоритма вложения. Алгоритм генерации пар вершин, необходимых для предсказания, описан в разделе 5.

После обучения модели на признаках, извлеченных из суррогатов, и её применения на признаках, извлеченных из входного графа, предсказания модели используются для нахождения вложений с помощью численной оптимизации. Соответствующий алгоритм описан в разделе 7.

2. Модель случайного гиперболического графа

В работе [10] была предложена модель случайного гиперболического графа.

В модели случайного гиперболического графа вершинам соответствуют точки на круге Пуанкаре ([19], гл. 7). Каждая точка на круге Пуанкаре описывается парой (r, ϕ), где r-радиальная координата, ϕ-угловая координата. Угловые координаты точек в данной модели распределены равномерно на отрезке $[0, 2\pi]$, а радиальные координаты имеют плотность распределения

$$f(r) = \alpha \frac{\sinh \alpha r}{\cosh \alpha R - 1} \tag{1}$$

на отрезке $[0, R]$, где $R = 2 \log n + C$, C – параметр моде-

ли, контролирующий среднюю степень вершины. Связь между параметром C и средней степенью вершины установлена в [10], уравнение 22.

Ещё один параметр модели α контролирует распределение степеней вершин. В [7] доказано, что при $\alpha > \frac{1}{2}$ степени вершин случайного гиперболического графа распределены согласно степенному закону с экспонентой $\gamma = 2\alpha + 1$.

Расстояние между парой точек $x = (r_x, \phi_x)$ и $y = (r_y, \phi_y)$ на круге Пуанкаре задаётся формулой

(2)
$$d(x, y) = \cosh^{-1}\left(\cosh r_x \cosh r_y - \sinh r_x \sinh r_y \cos\left(\phi_x - \phi_y\right)\right)$$

В модели случайного гиперболического графа ребро между вершинами (точками) u и v проводится с вероятностью

(3)
$$\Pr\left((u, v) \in E\right) = \frac{1}{1 + \exp\left(\frac{1}{2T}(d(u, v) - R)\right)}$$

где T – параметр модели (температура).

3. Оценка параметров суррогатов

Для генерации суррогатов необходимо задать 4 параметра: количество вершин n, среднюю степень вершин k, экспоненту степенного закона распределения вершин γ и температуру T.

Здесь и далее будем обозначать входной граф через $G = (V, E)$, где V – множество вершин, E – множество рёбер.

Количество вершин n оценивается методом, предложенным в [1]:

(4)
$$n = |V| \cdot (1 + \max\{0, 2F(1) - F(2)\}),$$

где $F(i)$ – доля вершин степени i во входном графе.

Параметр k (средняя степень вершин) оценивается как $2\frac{|E|}{|V|}$.

Экспонента степенного закона вычисляется из эмпирического распределения степеней вершин с помощью метода, предложенного в [5].

Наконец, температура T подбирается так, чтобы локальный кластерный коэффициент суррогата совпадал с локальным кластерным коэффициентом входного графа с точностью 0.01. Поскольку зависимость локального кластерного коэффициента от температуры монотонная, то необходимое значение температуры находится с помощью бинарного поиска на интервале $(0, 1)$.

Предлагаемый способ оценки параметров суррогатов представлен в алгоритме 1.

4. Признаки для предсказания радиальных координат

Обозначим множество соседей вершины u через $\Gamma(u)$. Для каждой вершины u вычисляются следующие признаки:

1) Степень $\deg(u) = |\Gamma(u)|$.
2) Коэффициент кластеризации $c(u) = \dfrac{2T(u)}{\deg(u)(\deg(u) - 1)}$, где $T(u)$ – количество треугольников, проходящих через вершину u.
3) Минимальная степень среди соседних вершин: $\min_{w \in \Gamma(u)} \deg(w)$.
4) Максимальная степень среди соседних вершин: $\max_{w \in \Gamma(u)} \deg(w)$.
5) Средняя степень соседних вершин: $\dfrac{\sum\limits_{w \in \Gamma(u)} \deg(w)}{\deg(u)}$.
6) Дисперсия степеней соседних вершин: $\dfrac{\sum\limits_{w \in \Gamma(u)} (\deg(w) - \mathbb{E}\deg(w))^2}{\deg(u)}$.
7) PageRank [14] вершины u.

Предлагаемый способ извлечения признаков вершины представлен в алгоритме 2.

Алгоритм 1 Оценка параметров суррогатов

Процедура EstimateParameters(V, E) ▷ Входом является граф с множеством вершин V и множеством ребер E

    $n = |V| \cdot (1 + \max\{0, 2F(1) - F(2)\})$ ▷ $F(i)$ – доля вершин степени i во входном графе

    $k \leftarrow 2\frac{|E|}{|V|}$

    $\gamma \leftarrow \text{FitPowerLaw}(F(1), F(2), \ldots, F(n-1))$ ▷ Процедура FitPowerLaw находит экспоненту степенного закона из эмпирического распределения степеней вершин [5]

    $T_{min} \leftarrow 0$

    $T_{max} \leftarrow 1$

    Повторять ▷ T подбирается так, чтобы локальный кластерный коэффициент суррогата совпадал с локальным кластерным коэффициентом входного графа с точностью 0.01.

        $T \leftarrow \frac{T_{min} + T_{max}}{2}$

        $V_{surrogate}, E_{surrogate} = \text{GenerateHyperbolicGraph}(n, k, \gamma, T)$ ▷ Процедура GenerateHyperbolicGraph реализует алгоритм генерации случайного гиперболического графа, описанный в [12]

        $LCC \leftarrow \text{LocalClusteringCoefficient}(V, E)$

        $LCC_{surrogate} \leftarrow \text{LocalClusteringCoefficient}(V_{surrogate}, E_{surrogate})$

        Если $LCC_{surrogate} > LCC$ тогда

            $T_{min} \leftarrow T$

        иначе

            $T_{max} \leftarrow T$

        Конец условия

    Пока выполняется $|LCC - LCC_{surrogate}| > 0.01$

    Вернуть n, k, γ, T

Конец процедуры

Алгоритм 2 Извлечение признаков вершины

Процедура ExtractVertexFeatures(u, V, E) ▷ Входом является вершина u графа (V, E)
    Вернуть
    $\deg(u)$, ▷ $\deg(u)$ – степень вершины u
    $\dfrac{2T(u)}{\deg(u)(\deg(u)-1)}$, ▷ $T(u)$ – количество треугольников, проходящих через вершину u
    $\min_{w\in\Gamma(u)} \deg(w)$,▷ $\Gamma(u)$ – множество соседей вершины u
    $\max_{w\in\Gamma(u)} \deg(w)$.
    $\dfrac{\sum\limits_{w\in\Gamma(u)} \deg(w)}{\deg(u)}$,
    $\dfrac{\sum\limits_{w\in\Gamma(u)} (\deg(w) - \mathbb{E}\deg(w))^2}{\deg(u)}$,
PageRank(u, V, E) ▷ PageRank [14] вершины u
Конец процедуры

5. Алгоритм генерации пар вершин

Для больших графов предсказание попарных расстояний для всех пар вершин неосуществимо на практике. Поэтому необходимо выбрать некоторое подмножество пар вершин, размер которого пропорционален $|V|$. Далее будем предполагать, что входной граф разрежен, то есть $|E| = C \cdot |V|$, где C – некоторая константа. Ниже описан алгоритм построения множества пар вершин $\hat{P}$, которые будут использоваться для предсказания попарных расстояний.

Назовём ядром графа множество вершин, имеющих степень не меньше $\sqrt{|V|}$:

$$V_c = \{v \in V : \deg(v) \geqslant \sqrt{|V|}\} \tag{5}$$

Оставшиеся вершины назовём периферией:

$$V_p = V \setminus V_c \tag{6}$$

Множество $\hat{P}$ является объединением трёх множеств:

1) Множества рёбер графа E.

2) Множества всех пар вершин из ядра $V_c \times V_c$. Это множество имеет размер $O(V)$ (см. [1]).

3) Множества пар вида (u_p, u_c), где $u_p \in V_p$, $u_c \in V_c$, $(u_p, u_c) \notin E$, где для каждой вершины u_p берётся $\deg(u_p)$ случайных вершин из ядра. Это множество имеет размер $O(|E|) = O(|V|)$, поскольку количество пар в этом множестве не превосходит суммарной степени всех вершин графа.

Ясно, что $|\hat{P}| = O(|V|)$.

Предлагаемый способ генерации пар вершин представлен в алгоритме 3.

6. Признаки для предсказания попарных расстояний

Для каждой пары вершин (u, v), порожденной алгоритмом 5, вычисляются следующие признаки (предполагается, что $\deg(u) > \deg(v)$):

Алгоритм 3 Генерация пар вершин

Процедура GeneratePairs(V, E) ▷ Входом является граф с множеством вершин V и множеством ребер E
    $V_c \leftarrow \{v \in V : \deg(v) \geqslant \sqrt{|V|}\}$
    $V_p \leftarrow V \setminus V_c$
    $P \leftarrow \{\}$
    Цикл $u_p \in V_p$ выполнять ▷ $Choice(A, n)$ возвращает n случайных элементов из A без возвращения
        Цикл $u_c \in Choice(V_c, deg(u_p))$ выполнять
            $P \leftarrow P \bigcup \{(u_p, u_c)\}$
        Конец цикла
    Конец цикла
    Вернуть $E \bigcup (V_c \times V_c) \bigcup P$
Конец процедуры

1) Является ли пара (u, v) ребром.
2) Индекс выделения ресурсов $\sum\limits_{w \in \Gamma(u) \bigcap \Gamma(v)} \frac{1}{|\Gamma(w)|}$ [18]
3) Коэффициент Жаккара $\dfrac{|\Gamma(u) \bigcap \Gamma(v)|}{|\Gamma(u) \bigcup \Gamma(v)|}$ [11]
4) Коэффициент предпочтительного присоединения $|\Gamma(u)| \cdot |\Gamma(v)|$ [11]
5) Количество общих соседей $|\Gamma(u) \bigcap \Gamma(v)|$
6) $\dfrac{|\Gamma(u) \bigcap \Gamma(v)|}{\sqrt{|\Gamma(u)| \cdot |\Gamma(v)|}}$
7) $\dfrac{|\Gamma(u) \bigcap \Gamma(v)|}{\min\{|\Gamma(u)|, |\Gamma(v)|\}}$
8) Минимальная степень общих соседей: $\min_{w \in \Gamma(u) \bigcap \Gamma(v)} \deg(w)$

9) Максимальная степень общих соседей: $\max_{w \in \Gamma(u) \bigcap \Gamma(v)} \deg(w)$.

10) Средняя степень общих соседей: $\dfrac{\sum\limits_{w \in \Gamma(u) \bigcap \Gamma(v)} \deg(w)}{\deg(u)}$.

11) Дисперсия степеней общих соседей: $\dfrac{\sum\limits_{w \in \Gamma(u) \bigcap \Gamma(v)} (\deg(w) - \mathbb{E}\deg(w))^2}{\deg(u)}$.

12) Все признаки из 4 для вершины u.

13) Все признаки из 4 для вершины v.

Предлагаемый способ извлечения признаков пар вершин представлен в алгоритме 4.

7. Алгоритм вложения вершин на основе предсказаний радиальных координат и попарных расстояний

Пусть $G = (V, E)$ – входной граф. Обозначим n_s число генерируемых суррогатов (это число является настраиваемым параметром алгоритма).

Процедура извлечения признаков выглядит так:

1) В соответствии с 4 для входного графа G извлекается матрица признаков X_r.

2) В соответствии с 5 и 4 для входного графа G извлекается матрица признаков X_d.

3) В соответствии с алгоритмом 3 для входного графа G оцениваются параметры n, k, γ, T.

4) Генерируется n_s суррогатов $G_1, G_2, \ldots, G_{n_s}$.

Алгоритм 4 Извлечение признаков пар вершин

Процедура ExtractVertexPairFeatures(u, v, V, E) ▷ Входом является пара вершин u, v графа (V, E)
    Вернуть
    $(u, v) \in E$,
    $\sum\limits_{w \in \Gamma(u) \bigcap \Gamma(v)} \frac{1}{|\Gamma(w)|}$, ▷ Индекс выделения ресурсов [18]
    $\dfrac{|\Gamma(u) \bigcap \Gamma(v)|}{|\Gamma(u) \bigcup \Gamma(v)|}$, ▷ Коэффициент Жаккара [11]
    $|\Gamma(u)| \cdot |\Gamma(v)|$, ▷ Коэффициент предпочтительного присоединения [11]
    $|\Gamma(u) \bigcap \Gamma(v)|$,
    $\dfrac{|\Gamma(u) \bigcap \Gamma(v)|}{\sqrt{|\Gamma(u)| \cdot |\Gamma(v)|}}$,
    $\dfrac{|\Gamma(u) \bigcap \Gamma(v)|}{\min\{|\Gamma(u)|, |\Gamma(v)|\}}$,
    $\min_{w \in \Gamma(u) \bigcap \Gamma(v)} \deg(w)$,
    $\max_{w \in \Gamma(u) \bigcap \Gamma(v)} \deg(w)$,
    $\dfrac{\sum\limits_{w \in \Gamma(u) \bigcap \Gamma(v)} \deg(w)}{\deg(u)}$,
    $\dfrac{\sum\limits_{w \in \Gamma(u) \bigcap \Gamma(v)} (\deg(w) - \mathbb{E}\deg(w))^2}{\deg(u)}$,
    ExtractVertexFeatures(u, V, E),
    ExtractVertexFeatures(v, V, E)
Конец процедуры

5) Для всех суррогатов в соответствии с 4 вычисляются матрицы признаков, их вертикальная конкатенация образует матрицу признаков $\overline{X}_r$. Соответственно ей составляется вектор ответов $\overline{y}_r$, содержащий радиальные координаты соответствующих вершин.

6) Для всех суррогатов в соответствии с 5 и 4 вычисляются матрицы признаков, их вертикальная конкатенация образует матрицу признаков $\overline{X}_d$. Соответственно ей составляется вектор ответов $\overline{y}_d$, содержащий гиперболические расстояния между соответствующими парами вершин.

Далее на паре $(\overline{X}_r, \overline{y}_r)$ обучается модель M_r, а на паре $(\overline{X}_d, \overline{y}_d)$ обучается модель M_d. Алгоритм обучения может быть произвольным. В данной работе используется алгоритм CatBoost [17], который является промышленным стандартом в машинном обучении. CatBoost реализует метод градиентного бустинга – итеративного приближения неизвестной функции по конечному набору точек, где на каждой итерации жадным образом дообучается аддитивная добавка к уже обученной функции.

Предсказание модели M_r на матрице признаков X_r обозначим y_r, а предсказание модели M_d на матрице признаков X_d обозначим y_d. Вектор y_r содержит предсказания радиальных координат для всех вершин $u \in V$, а вектор y_d содержит предсказания гиперболических расстояний для всех пар вершин из $\hat{P}$.

Обозначим через V_{sorted} список вершин графа G, отсортированных по убыванию степени. Зафиксируем шаг $\Delta\phi$ и обозначим $\Phi_{\Delta\phi} = [0, \Delta\phi, 2\Delta\phi, 3\Delta\phi, \dots,]$ равномерную сетку от 0 до 2π с шагом $\Delta\phi$. Также обозначим $V_{embedded}$ множество вершин (изначально пустое), для которых вложения уже найдены.

Для каждой вершины $u \in V_{sorted}$ найдём все пары из $\hat{P}$, в которые эта вершина входит. Множество всех вершин

в этих парах обозначим $\hat{V}^{(u)}$. Далее, обозначим $\hat{V}^{(u)}_{embedded} = \hat{V}^{(u)} \bigcap V_{embedded}$. Для каждой пары (u, v), где $v \in \hat{V}^{(u)}_{embedded}$, выбирается то значение $\phi \in \Phi_{\Delta\phi}$, которое даёт минимум невязки

$$\sum_{v \in \hat{V}^{(u)}_{embedded}} (d((r(u), \phi), (r(v), \phi_v)) - d_{uv})^2 \tag{7}$$

где

- $r(u), r(v)$ – предсказания радиальных координат вершин u, v соответственно;
- ϕ_v – найденная ранее угловая координата вершины v (напомним, что $v \in V_{embedded}$);
- $d((r_1, \phi_1), (r_2, \phi_2)) = \operatorname{arccosh}(\cosh r_1 \cosh r_2 - \sinh r_1 \sinh r_2 \cos(\phi_1 -$ – гиперболическое расстояние между точками (r_1, ϕ_1) и (r_2, ϕ_2);
- d_{uv} – предсказанное гиперболическое расстояние между вершинами u и v (предсказание определено, поскольку $(u, v) \in \hat{P}$).

Предлагаемый способ вложения вершин в гиперболическое пространство представлен в алгоритме 5.

Численный пример вложения представлен на рисунке 1.

8. Экспериментальные результаты

Для экспериментальной оценки предлагаемого алгоритма было сгенерировано 30 реализаций случайного гиперболического графа с различными наборами параметров n, k, γ, T. Конкретные значения параметров указаны на графиках 2, 3, 4.

В каждой реализации графа была взята его самая большая связная компонента, на которой был запущен предлагаемый алгоритм вложения.

Качество вложения оценивается по следующим метрикам:

Алгоритм 5 Вложение вершин в гиперболическое пространство

Процедура MakeEmbedding(V, E, n_s) ▷ Входом является граф с множеством вершин V и множеством ребер E, а также параметр n_s
    $X_r \leftarrow$ Matrix()
    $X_d \leftarrow$ Matrix()
    Цикл $u \in V$ выполнять
        Append(X_r, ExtractVertexFeatures(u, V, E))
    Конец цикла
    $\hat{P} \leftarrow$ GeneratePairs(V, E)
    Цикл $(u, v) \in \hat{P}$ выполнять
        Append(X_d, ExtractVertexPairFeatures(u, v, V, E))
    Конец цикла
    $n, k, \gamma, T \leftarrow$ EstimateParameters(V, E)
    $\overline{X}_r \leftarrow$ Matrix()
    $\overline{y}_r \leftarrow$ Vector()
    $\overline{X}_d \leftarrow$ Matrix()
    $\overline{y}_d \leftarrow$ Vector()
    Цикл iот 1 до n_s выполнять
        $(\overline{V}, \overline{E}) \leftarrow$ GenerateHyperbolicGraph(n, k, γ, T)
        Цикл $\overline{u} \in \overline{V}$ выполнять
            Append($\overline{X}_r$, ExtractVertexFeatures($\overline{u}, \overline{V}, \overline{E}$))
            Append($\overline{y}_r, r(\overline{u})$) ▷ $r(\overline{u})$ – радиальная координата вершины $\overline{u}$
        Конец цикла
        Цикл $(\overline{u}, \overline{v}) \in$ GeneratePairs($\overline{V}, \overline{E}$) выполнять
            Append($\overline{X}_d$, ExtractVertexPairFeatures($\overline{u}, \overline{v}, \overline{V}, \overline{E}$))
            Append($\overline{y}_d, d((r(\overline{u}), \phi(\overline{u})), (r(\overline{v}), \phi(\overline{v}))$) ▷ $d((r_1, \phi_1), (r_2, \phi_2))$ – расстояние, вычисленное по формуле 2
        Конец цикла
    Конец цикла
    $M_r \leftarrow$ Fit($\overline{X}_r, \overline{y}_r$) ▷ Процедура Fit обучает модель
    $y_r \leftarrow$ Predict(M_r, X_r) ▷ Процедура Predict применяет обученную модель
    $M_d \leftarrow$ Fit($\overline{X}_d, \overline{y}_d$)
    $y_d \leftarrow$ Predict(M_d, X_d)
    $V_{sorted} \leftarrow$ SortByDegreeDesc(V, E) ▷ SortByDegreeDesc(V, E) возвращает список вершин графа,

Рис. 1. Слева: реализация случайного гиперболического графа с параметрами $n = 50$, $\gamma = 2.5$, $k = 4$, $T = 0.1$. Справа: вложение этого графа, полученное с помощью алгоритма 5.

- Относительная ошибка среднего гармонического расстояния. Между всеми парами вершин графа с помощью алгоритма Дейкстры вычисляются кратчайшие пути, среднее гармоническое длин всех кратчайших путей обозначим d. Также для всех пар вершин найдём пути с помощью алгоритма жадной маршрутизации (если алгоритм не нашёл пути между парой вершин, то считаем длину пути между ними равной бесконечности); среднее гармоническое длин всех найденных путей обозначим $\hat{d}$. Тогда относительная ошибка среднего гармонического расстояния вычисляется по формуле $\dfrac{\hat{d} - d}{d}$. Обозначение на графиках: HDS.

- Средний ранг. Для каждой вершины u остальные вершины графа сортируются по возрастанию гиперболического расстояния, после чего суммируются порядковые номера тех вершин v, для которых справедливо $(u, v) \in E$. Полученное значение называется рангом вер-

шины $\mathrm{rank}(u)$. Средний ранг равен $\frac{\sum\limits_{u \in V} \mathrm{rank}(u)}{|V|}$. Обозначение на графиках: MR.

- Средний относительный ранг. То же самое, что средний ранг, только ранг каждой вершины v перед усреднением нормируется на минимально возможное значение ранга для этой вершины, равное $\frac{\deg(v)+1}{2}$. Обозначение на графиках: MRR.

- Полнота@1d. Для каждой вершины v отбирается $\deg(v)$ ближайших по гиперболическому расстоянию вершин, после чего для полученного множества вершин вычисляется доля рёбер $\frac{\sum\limits_{u}[(u,v) \in E]}{\deg(v)}$ Обозначение на графиках: R1.

- Полнота@2d. То же самое, что Полнота@1d, только для каждой вершины v отбирается $2 \cdot \deg(v)$ ближайших по гиперболическому расстоянию вершин. Обозначение на графиках: R2.

Результаты моделирования представлены на графиках 2, 3, 4. В качестве величины ошибки на графиках показано одно стандартное отклонение. Вложения, полученные предлагаемым алгоритмом, сравниваются с истинными вложениями вершин.

9. Заключение

Полученные результаты показывают, что предлагаемый алгоритм находит вложения, которые позволяют находить близкие к оптимальным пути с помощью жадной маршрутизации. Также высокие значения метрик R1 и R2 говорят о том, что получаемые вложения подходят для задачи предсказания рёбер.

10. Перспективы

Одной из основных частей предлагаемого алгоритма является процедура извлечения матрицы признаков для вершин. Альтернативой этому может быть применение глубоких нейросетей, способных автоматически извлекать признаковое описание суррогатов. Обзор архитектур таких нейросетей для графов представлен в [3].

Также алгоритм численной оптимизации, используемый 7, может быть существенно улучшен (например, за счёт применения градиентных методов).

Список литературы

[1] Thomas Bläsius и др. “Efficient embedding of scale-free graphs in the hyperbolic plane”. В: LIPIcs-Leibniz International Proceedings in Informatics. Т. 57. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2016.

[2] Marián Boguná, Fragkiskos Papadopoulos и Dmitri Krioukov. “Sustaining the internet with hyperbolic mapping”. В: Nature communications 1 (2010), с. 62.

[3] Michael M Bronstein и др. “Geometric deep learning: going beyond euclidean data”. В: IEEE Signal Processing Magazine 34.4 (2017), с. 18—42.

[4] Elisabetta Candellero и Nikolaos Fountoulakis. “Clustering and the hyperbolic geometry of complex networks”. В: Internet Mathematics 12.1-2 (2016), с. 2—53.

[5] Aaron Clauset, Cosma Rohilla Shalizi и Mark EJ Newman. “Power-law distributions in empirical data”. В: SIAM review 51.4 (2009), с. 661—703.

[6] Christopher De Sa и др. “Representation tradeoffs for hyperbolic embeddings”. В: arXiv preprint arXiv:1804.03329 (2018).

[7] Luca Gugelmann, Konstantinos Panagiotou и Ueli Peter. “Random hyperbolic graphs: degree sequence and clustering”. В: International Colloquium on Automata, Languages, and Programming. Springer. 2012, с. 573—585.

[8] William L Hamilton, Rex Ying и Jure Leskovec. “Representation learning on graphs: Methods and applications”. В: arXiv preprint arXiv:1709.05584 (2017).

[9] Marcos Kiwi и Dieter Mitsche. “A bound for the diameter of random hyperbolic graphs”. В: 2015 Proceedings of the Twelfth Workshop on Analytic Algorithmics and Combinatorics (ANALCO). SIAM. 2014, с. 26—39.

[10] Dmitri Krioukov и др. “Hyperbolic geometry of complex networks”. В: Physical Review E 82.3 (2010), с. 036106.

[11] Kleinberg J. Liben-Nowell D. “The Link Prediction Problem for Social Networks”. В: CIKM 2003. 2003.

[12] Moritz von Looz, Henning Meyerhenke и Roman Prutkin. “Generating random hyperbolic graphs in subquadratic time”. В: International Symposium on Algorithms and Computation. Springer. 2015, с. 467—478.

[13] Mark E.J. Newman. Networks: an introduction. Oxford university press, 2010.

[14] Lawrence Page и др. The PageRank citation ranking: Bringing order to the web. Тех. отч. Stanford InfoLab, 1999.

[15] Fragkiskos Papadopoulos, Rodrigo Aldecoa и Dmitri Krioukov. “Network geometry inference using common neighbors”. В: Physical Review E 92.2 (2015), с. 022807.

[16] Fragkiskos Papadopoulos, Constantinos Psomas и Dmitri Krioukov. “Network mapping by replaying hyperbolic growth”. В: IEEE/ACM Transactions on Networking (TON) 23.1 (2015), с. 198—211.

[17] Liudmila Prokhorenkova и др. “CatBoost: unbiased boosting with categorical features”. В: Advances in Neural Information Processing Systems. 2018, с. 6638—6648.

[18] Linyuan Lü Tao Zhou и Yi-Cheng Zhang. “Predicting missing links via local information”. В: The European Physical Journal 71.4 (2009), с. 623—630.

[19] А.Б. Сосинский. Геометрии. МЦНМО, 2017.

Рис. 2. Зависимость качества вложения от параметра T при $n = 100,\ \gamma = 2.5,\ k = 6$

Рис. 3. Зависимость качества вложения от параметра γ при $n = 100,\ k = 6,\ T = 0.5$

Рис. 4. Зависимость качества вложения от параметра k при $n = 100,\ \gamma = 2.5,\ T = 0.5$