4. ЛЕКЦИЯ – Мультиреляционные данные и графы знаний

Большинство методов обработки мультиреляционных данных, изначально были разработаны для задачи графа знаний. В графах знаний, представляет мультиреляционный граф $G = (V, E)$, где ребра определяются как кортежи $e = (u, \tau, v)$, указывающие на наличие определенного отношения $\tau \in T$ между двумя вершинами u и v. Такие многореляционные графы часто называют графами знаний, поскольку можно интерпретировать кортеж (u, τ, v) как указание на то, что конкретный «факт» имеет место между двумя вершинами u и v. Например, в биомедицинском графе знаний может быть тип ребра $\tau =$ ЛЕЧЕНИЕ, а ребро $(u,$ ЛЕЧЕНИЕ$, v)$ может указывать на то, что лекарство, связанное с вершиной u, лечит заболевание, связанное с вершиной v. Обычно цель графа знаний состоит в том, чтобы предсказать недостающие ребра в графе, т. е. предсказание отношений, но есть и примеры задач классификации вершин с использованием мультиреляционных графов.

Рассмотрим краткий обзор методов встраивания мультиреляционных графов, но важно отметить, что полное рассмотрение графа знаний выходит за рамки данного учебного пособия. Однако заинтересованных читателей для более углубленного изучения предлагаем познакомиться с работами Сорокина А,Б. посвященные ситуационному анализу Болотовой Л.С.

4.1. Реконструкция мультиреляционных данных

Как и в случае с простыми графами, можно рассматривать вложение мультиреляционных графов как задачу реконструкции. Если есть вложения z_u и z_v двух вершин, то цель – восстановить отношения между этими вершинами. Однако теперь сложность (по сравнению с темой предыдущей главы) заключается в том, что нам теперь приходится иметь дело с наличием нескольких типов ребер. Чтобы устранить эту сложность, необходимо расширить декодер, чтобы сделать его многореляционным. Вместо того, чтобы принимать пару вложений вершин в качестве входных данных, теперь необходимо определить декодер как принимающий не только пару вложений вершин, но и тип ребра, т. е. $DEC: \mathbb{R}^d \times \mathcal{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^+$.

Можно интерпретировать вывод этого декодера, т. е. $DEC(z_u, \tau, z_v)$ как вероятность существования ребра (u, τ, v) в графе. Чтобы привести конкретный пример, один из самых простых и ранних подходов к изучению многореляционных вложений (называемых алгоритмом RESCAL) декодер был определен как (4.1):

$$DEC(u, \tau, v) = {z_u}^T, R_\tau, z_v \quad (4.1)$$

где $R_\tau \in \mathbb{R}^{\mathrm{d}\times\mathrm{d}}$ – обучаемая матрица, характерная для отношения $\tau \in \mathcal{R}$. Чтобы упростить работу с этим декодером, можно обучить матрицу вложения Z и матрицы отношений R_τ, $\forall\tau \in \mathcal{R}$, используя базовую потерю восстановления (4.2, 4.3):

$$L = \sum_{u\in V}\sum_{v\in V}\sum_{\tau\in\mathcal{R}}\|DEC(u, \tau, v) - A[u, \tau, v]\|^2 \quad (4.2)$$

$$= \sum_{u\in V}\sum_{v\in V}\sum_{\tau\in\mathcal{R}}\|{z_u}^T, R_\tau, z_v - A[u, \tau, v]\|^2 \quad (4.3)$$

где $A \in \mathbb{R}^{|V|\times|\mathcal{R}|\times|V|}$ – тензор смежности для мультиреляционного графа. Если оптимизировать уравнение (4.2), то фактически выполняется своего рода тензорная факторизация. Таким образом, идея факторизации тензора обобщает подходы к матричной факторизации.

Как уже обсуждали, разнообразие методов вложений вершин в значительной степени связано с использованием различных декодеров (DEC), мер подобия ($S[u, v]$) и функций потерь (L). Декодер дает оценку между парой вложений вершин; функция подобия определяет, какое сходство между вершинами необходимо декодировать; функция потерь показывает, как оценить несоответствие между выходными данными декодера и мерой сходства истинности.

В многореляционном представлении также видим разнообразие декодеров и функций потерь. Однако почти все мультиреляционные методы вложения просто определяют меру подобия непосредственно на основе тензора смежности.

Другими словами, все методы предполагают, реконструкцию непосредственных (мультиреляционных) соседей из низкой размерности вложений. Это связано со сложностью определения отношений соседства более высокого порядка в мультиреляционных графах, а также с тем фактом, что большинство методов мультиреляционного вложения были специально разработаны для предсказания отношений.

4.2. Функции потерь

Как уже обсуждалось, двумя ключевыми компонентами метода многореляционного вложения вершин являются декодер и функция потерь. Начнем с краткого обсуждение стандартных функций потерь, используемых для выбранной задачи, прежде чем обратить внимание на множество декодеров, которые были предложены в литературе.

Как мотивацию рассматриваемой функций потерь, стоит учесть недостатки простой потери реконструкции, которая введена в уравнении 4.2. Есть две основные проблемы функции потерь. Первая проблема заключается в том, что это чрезвычайно дорого вычислительно. Вложенные суммы в уравнении 4.2 требуют $O(|V|^2||R|)$ операций, и это время вычислений будет непомерно большим для многих больших графов. Более того, поскольку многие мультиреляционные графы разрежены, т. е. $|\mathrm{E}| << |V|^2||R| -$ в идеале нам нужна функция потерь, равная $O(|E|)$. Вторая проблема более тонкая. Наша цель — декодировать тензор смежности из низкоразмерных вложений вершин. Мы знаем, что (в большинстве случаев) этот тензор будет содержать только бинарные значения, но среднеквадратическая ошибка в уравнении 4.2 не совсем подходит для такого бинарного сравнения. На самом деле среднеквадратическая ошибка – это естественная потеря для регрессии, тогда как наша цель – нечто более близкое к классификации на ребрах.

Кросс-энтропия с отрицательной выборкой. Одной из популярных функций потерь, которая одновременно эффективна и подходит для нашей задачи, является кросс-энтропийная потеря с отрицательной выборкой. Она определяется как (4.4):

$$L = \sum_{(u,\tau,v)\in E} -\log\left(\sigma\left(DEC(z_u,\tau,z_v)\right)\right) - \gamma\mathbb{E}_{v_n\sim P_{n,u}(V)}\left[\log\left(\sigma\left(-DEC\left(z_u,\tau,z_{v_n}\right)\right)\right)\right] \quad (4.4)$$

где σ обозначает логистическую функцию, $P_{n,u}(V)$ обозначает распределение «отрицательной выборки» по множеству вершин V (которое может зависеть от u), а $\gamma > 0$ — гиперпараметр. По сути, это те же потери, что и для node2vec (уравнение 3.12). Однако будут рассматриваться общие мультиреляционные декодеры.

Это называется кросс-энтропийной потерей, потому что она выводится из стандартной бинарной кросс-энтропийной потери. Поскольку передаются выходные данные декодера логистической функции и получаются нормализованные оценки [0,1], которые можно интерпретировать как вероятности. Выражение (4.5):

$$\log\left(\sigma\left(DEC(z_u,\tau,z_v)\right)\right) \quad (4.5)$$

равняется логарифмической вероятности того, что предсказывается «истина» для ребра, которое действительно существует в графе. С другой стороны, выражение (4.6):

$$\mathbb{E}_{v_n \sim P_{n,u}(V)} \left[\log \left(\sigma \left(-DEC\left(z_u, \tau, z_{v_n}\right) \right) \right) \right] \tag{4.6}$$

равняется ожидаемой логарифмической вероятности того, что правильно предсказана «ложь» для ребра, которого нет в графе.

На практике, математическое ожидание вычисляется с использованием метода Монте-Карло, наиболее популярная форма этой функции выглядит так (4.7):

$$L = \sum_{(u,\tau,v) \in E} - \log \left(\sigma\left(DEC(z_u, \tau, z_v) \right) \right) - \sum_{v_n \in P_{n,u}} \left[\log \left(\sigma \left(-DEC\left(z_u, \tau, z_{v_n}\right) \right) \right) \right] \tag{4.7}$$

где $P_{n,u}$ – это (обычно небольшой) набор вершин, отобранных из $P_{n,u}(V)$.

Примечание по отрицательной выборке. Способ генерации отрицательных выборок может оказать большое влияние на качество уже обработанных вложенных слоев (эмбеддинга). Наиболее распространенный подход к определению распределения заключается в выборе простого равномерного распределения между всеми вершинами графа. Если использовать эту стратегию, то есть шанс получить «ложноотрицательные результаты» при расчете кросс-энтропии возрастает в разы. Другими словами, есть вероятность, что случайно может быть выбран «отрицательный» кортеж (u, τ, v_n), который действительно существует в графе. Эту проблему решают путем фильтрации подобных ложноотрицательных результатов.

Другие варианты работы с отрицательной выборкой направлены на получение еще более «сложных» отрицательных выборок. Например, некоторые связи могут существовать только между определенными типами вершин (как вариант, вершина, представляющая человека в графе знаний, вряд ли будет связанна с ребром, у которого тип отношений «произведено»). Таким образом, одна из стратегий состоит в том, чтобы отбирать только отрицательные примеры, которые удовлетворяют подобным ограничениям по типу отношений. Предлагают подойти к отбору сложных отрицательных выборок с использованием состязательных моделей.

Так же стоит обратить внимание (без потери общности) на то, что отрицательная выборка находится во второй вершине в пограничном кортеже. То есть предполагается, что для изображения отрицательной выборки, необходимо заменить последнюю вершину в кортеже (u, τ, v) отрицательной выборкой v_n. Выборка последней вершины всегда упрощает нотацию, но может и привести к смещениям в мультиреляционных графах, где важно направление ребер. На практике иногда лучше использовать отрицательные выборки как для первой (т.е. u), так и для последней вершины (т.е. v) отношения.

Максимальная маржинальная потеря. Другой популярной функцией потерь, используемой для встраивания мультиреляционных узлов, является (4.8):

$$L = \sum_{(u,\tau,v)\in E} \sum_{v_n \in P_{n,u}} max\left(0, -DEC(z_u, \tau, z_v) + DEC\left(z_u, \tau, z_{v_n}\right) + \Delta\right) \quad (4.8)$$

Таким образом, снова сравнивается декодированную оценку для истинной пары с отрицательной выборкой — стратегия, которую часто называют контрастной оценкой. Однако вместо того, чтобы рассматривать это как задачу двоичной классификации, в уравнении 4.8 сравниваются выходные данные декодеров. Если оценка для «истинной» пары больше, чем для «отрицательной» пары, то образуется потеря. Этот термин называется Δ маржой, и проигрыш будет равен 0, если разница в оценках будет по крайней мере такой же.

4.3. Мультиреляционные декодеры

Таким образом были представлены две, наиболее популярные, функции потерь, которые используются для изучения вложений (эмбеддингов) мультиреляционных узлов. Эти потери могут быть объединены с различными функциями для декодироания, и теперь необходимо обратить внимание на определение декодеров. До сих пор мы обсуждали только один возможный мультиреляционный декодер, так называемый RESCAL (4.9):

$$DEC(z_u, \tau, z_v) = {z_u}^T R_\tau z_v \quad (4.9)$$

В декодере RESCAL связывается обучаемая матрица $R_\tau \in \mathbb{R}^{d\times d}$ с каждым отношением. Однако одним из ограничений этого подхода и причиной, по которой он используется нечасто, является его высокая вычислительная и статистическая стоимость для представления отношений. Существует $O(d^2)$ параметров для каждого типа отношений в RESCAL. Это означает, что для представления отношений требуется на порядок больше параметров, чем для сущностей.

Более популярные современные декодеры стремятся использовать только параметры $O(d)$ для представления каждого отношения. Рассмотрим несколько популярных вариантов мультиреляционных декодеров, которые обобщены в таблице 4.1.

Таблица 4.1. Краткое описание некоторых популярных декодеров, используемых для обработки мультиреляционных данных.

Название	Декодер	Параметры отношения
RESCAL	$z_u{}^T R_\tau z_v$	$R_\tau \in \mathbb{R}^{d\times d}$
TransE	$-\|z_u + r_\tau - z_v\|$	$r_\tau \in \mathbb{R}^d$
TransX	$-\|g_{1,\tau} z_u + r_\tau - g_{2,\tau} z_v\|$	$r_\tau \in \mathbb{R}^d, g_1, \tau, g_2, \tau \in \mathbb{R}^d \to \mathbb{R}^d$
DistMult	$\langle z_u, r_\tau, z_v\rangle$	$r_\tau \in \mathbb{R}^d$
ComplEx	$Re(\langle z_u, r_\tau, \bar{z}_v\rangle)$	$r_\tau \in \mathbb{C}^d$
RotatE	$-\|z_u \circ r_\tau - z_v\|$	$r_\tau \in \mathbb{C}^d$

Трансляционные декодеры. Это один из популярных классов декодеров представляет отношения как преобразования в пространстве вложений (эмбеддингов). Этот подход был инициирован в виде модели TransE, которая определила декодер как (4.10)

$$DEC(z_u, \tau, z_v) = -\|z_u + r_\tau - z_v\| \tag{4.10}$$

В подобных подходах представляем каждое отношение, используя d-мерное вложение. Вероятность появления ребра пропорциональна расстоянию между вложением головной и хвостовой вершин, после операции по переводу головной вершины в соответствии с вложением отношения. TransE является одним из самых ранних предложенных мультиреляционных декодеров и продолжает оставаться надежной основой во многих приложениях.

Однако одним из ограничений модели TransE является её простота, и многие в своих работах предлагали идеи её расширения. Подобные модифицированные версии называются TransX, и они имеют вид (4.11):

$$DEC(z_u, \tau, z_v) = -\|g_{1,\tau} z_u + r_\tau - g_{2,\tau} z_v\| \tag{4.11}$$

где $g_{i,\tau}$ – обучаемые преобразования, зависящие от отношения τ. Например, модель TransH определяет декодер как (4.12)

$$DEC(z_u, \tau, z_v) = -\|(z_u - w_r^T z_u w_r) + r_\tau - (z_v - w_r^T z_v w_r)\| \quad (4.12)$$

Подход TransH проецирует вложения сущностей на обучаемую гиперплоскость, определяемую отношением, заданным вектором нормалей w_r, перед выполнением преобразования.

Многолинейные скалярные произведения. Вместо того, чтобы определять декодер на основе преобразования вложений, можно построить его, обобщение скалярного произведения декодера из простых графов. В этом подходе, часто называемом DistMult определяется декодер как (4.13, 4.14)

$$DEC(z_u, \tau, z_v) = \langle z_u, r_\tau, z_v \rangle \quad (4.13)$$

$$= \sum_{i=1}^{d} z_u[i] \times r_\tau[i] \times z_v[i] \quad (4.14)$$

Этот подход использует прямое обобщение скалярного произведения, которое должно быть определено по трем векторам.

Сложные декодеры. Одним из ограничений декодера DistMult в уравнении (4.13, 4.14) является то, что он может работать только с симметричными отношениями.

Это серьезное ограничение, поскольку многие типы отношений в мультиреляционных графах являются направленными и асимметричными. Чтобы решить эту проблему, был предложено расширение для DistMult за счет использования комплекснозначных вложений (эмбеддингов). Они определяют ComplEx как (4.15, 4.16)

$$DEC(z_u, \tau, z_v) = Re(\langle z_u, r_\tau, \bar{z}_v \rangle) \quad (4.15)$$

$$= Re\left(\sum_{i=1}^{d} z_u[i] \times r_\tau[i] \times z_v[i]\right) \quad (4.16)$$

где $z_u, r_\tau, z_v \in \mathbb{C}^{\mathrm{d}}$ теперь являются комплекснозначными вложениями, а R обозначает вещественную компоненту комплексного вектора. Так как берется комплексно сопряженный вектор хвостового вложения (эмбеддинга) z_v ,то подход к декодированию может учитывать асимметричные отношения.

Связанный подход, называемый RotatE, определяет декодер как вращения на комплексной плоскости следующим образом (4.17):

$$DEC(z_u, \tau, z_v) = -\|z_u \circ r_\tau - z_v\| \quad (4.17)$$

где $\circ$ – это произведение Адамара. В уравнении 4.17 предполагается, что все вложения являются комплексными значениями, а также дополнительно вводится ограничение на элементы r_τ так, чтобы $|r_\tau[i]| = 1, \forall i \in \{1, \dots, d\}$. Это ограничение подразумевает, что каждое измерение вложения отношения может быть представлено как $r[i] = e^{tQr,i}$ и, таким образом, соответствует повороту в комплексной плоскости.

Репрезентативные способности. Одним из способов охарактеризовать различные мультиреляционные декодеры является их способность представлять различные логические шаблоны в отношениях.

Рассмотрим симметрию и антисимметрию. Например, многие отношения симметричны, что означает, что (4.18)

$$(u, \tau, v) \in E \leftrightarrow (v, \tau, u) \in E \tag{4.18}$$

В других случаях имеем явно антисимметричные отношения, которые удовлетворяют (4.19):

$$(u, \tau, v) \in E \rightarrow (v, \tau, u) \notin E \tag{4.19}$$

Один важный вопрос заключается в том, способны ли разные декодеры моделировать как симметричные, так и антисимметричные отношения. DistMult, например, может представлять только симметричные отношения. С другой стороны, TransE может представлять только антисимметричные отношения. По определению для данных подходов.

Инверсия. С симметрией связано понятие инверсии, когда одно отношение подразумевает существование другого, с противоположной направленностью (4.20):

$$(u, \tau_1, v) \in E \leftrightarrow (v, \tau_2, u) \in E \tag{4.20}$$

Большинство декодеров способны представлять обратные отношения, хотя, опять же, DistMult не в состоянии смоделировать такой шаблон.

Композиционность. Также необходимо рассмотреть вопрос о том, могут ли декодеры кодировать композиционность между представлениями отношений формы (4.21):

$$(u, \tau_1, y) \in E \wedge (y, \tau_2, u) \in E \rightarrow (v, \tau_3, u) \in E \tag{4.21}$$

Например, в TransE можно учесть это, определив $r_{\tau_3} = r_{\tau_1} + r_{\tau_2}$. Аналогичным образом можyj смоделировать композиционность в RESCAL, определив $R_{\tau_3} = R_{\tau_1} + R_{\tau_2}$.

В целом, рассмотрение данных реляционных шаблонов полезно для сравнения возможностей представления различных мультиреляционных декодеров. Но на практике не стоит ожидать, что эти шаблоны будут выполняться один в один, может существовать много различных отношений, которые в некоторой степени будут их отражать.