Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика, искусственный интеллект и системы управления (ИУ5)

КАФЕДРА Системы обработки информации и управления (ИУ5)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

Оценка селективности операций плана выполнения запроса в СУБД методом машинного обучения

Студент ИУ5-31М _____ **Вивчарук Р. В.**
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель _____ **Гапанюк Ю. Е.**
(Подпись, дата) (И.О.Фамилия)

2022 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой __ИУ5__
(Индекс)
_____ ___В. И. Терехов_
(И.О.Фамилия)
« _____ » _____ 20 ____ г.

З А Д А Н И Е
на выполнение курсового проекта

по теме _«Оценка селективности операций плана выполнения запроса в СУБД методом машинного обучения»

Студент группы _ИУ5-31М_

Вивчарук Ростислав Владимирович
(Фамилия, имя, отчество)

Направленность курсового проекта (учебная, исследовательская, практическая, производственная, др.)

исследовательская

Источник тематики (кафедра, предприятие, НИР) _____

График выполнения НИР: 25% к ___ нед., 50% к ___ нед., 75% к __ нед., 100% к ___ нед.

Техническое задание __Изучить существующие методы для оценки селективности операций плана выполнения SQL-запроса в СУБД, которые основываются на применении машинного обучения, провести их сравнительный анализ._____

Оформление курсового проекта:

Расчетно-пояснительная записка на 30-40 листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

Дата выдачи задания « ___ » _____ 20__ г.

Руководитель курсового проекта _____ ____Гапанюк Ю. Е.___
(Подпись, дата) (И.О.Фамилия)

Студент _____ ___Вивчарук Р. В.___
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Оглавление

ВВЕДЕНИЕ

Одной из основных задач информационных систем является обработка больших объемов данных. Банковские системы, различные веб-сервисы, социальные сети, корпоративные программные системы – все они выполняют функцию хранения и обработки данных, к которой предъявляются схожие требования: высокая надежность, скорость чтения и записи, возможность создавать резервные копии, реплики. Для соблюдения перечисленных требований часто используются универсальные системы управления базами данных (СУБД). В данной работе рассматриваются именно реляционные СУБД (РСУБД).

Поступающий в РСУБД SQL-запрос имеет декларативную природу. В запросе формулируется, какие операции над данными требуется выполнить, но не описывается каким образом. СУБД, а именно оптимизатор запросов, производит рад манипуляций над SQL-запросом, чтобы определить способ выполнения, который называется физическим планом выполнения запроса. По одному запросу можно построить множество планов, стоимость выполнения которых может отличаться на несколько порядков, поэтому построение наиболее оптимального плана является ключевым для производительности СУБД.

Одна из ошибок, которая допускается оптимизатором – неправильная оценка количества отбираемых условиями кортежей, так как по умолчанию планировщик считает, что условия в запросе независимы [1]. Приведем пример.

Серверу СУБД поступает следующий запрос: `SELECT * FROM users WHERE age < 15 AND married = TRUE` (получить всех состоящих в браке пользователей, младше 15 лет). Доля пользователей, удовлетворяющих запросу, будет считаться равной произведению доли пользователей младше 15 лет и доли пользователей в браке. Естественно, такая оценка ошибочна, так как среди пользователей младше 15 лет никто не состоит в браке.

Существуют различные подходы к решению этой проблемы. Одним из наиболее популярных в сообществе разработчиков СУБД подходом являются

многомерные гистограммы, которые позволяют напрямую считать долю пользователей, удовлетворяющих набору условий [2]. Очевидной проблемой такого подхода является невозможность построения гистограммы по достаточно большому количеству столбцов и сложность определения тех столбцов, по которым нужно строить гистограммы. Также существуют несколько работ в области машинного обучения, посвященных данной проблеме, которые могут вылиться в перспективные решения, но также имеют свои недостатки.

В соответствие с вышеописанным, целью данной курсовой работы является обзор и сравнение существующих решений с применением машинного обучения для оценки селективности (в некоторых случаях – мощности) операций плана выполнения запроса к СУБД. В работе используется следующий ряд терминов:

• Стоимость – количество ресурсов для выполнения вершины и ее поддерева.

• Мощность (кардинальность) – чисто кортежей, которые будут возвращены вершиной плана на вышестоящий уровень.

• Селективность (выборочность) – доля кортежей, удовлетворяющих условию запроса.

1 Методы предсказания селективности с машинным обучением

1.1 Адаптивная оптимизация

В методе, описанном в источниках [1, 2, 3] предложен подход к реализации адаптивной оптимизации запросов. Адаптивность заключается в том, что при выполнении запроса мы можем получить некоторые данные, например, стоимость выполнения выбранного плана, фактические значения мощностей вершин плана и т. п. То есть, собирается статистика, которая затем может быть использована для машинного обучения. Играет роль тот факт, что запросы к базе данных зачастую имеют схожую структуру, в них изменяются только константы. Спустя некоторое время использования базы данных, оптимизатор адаптируется к тем данным, которыми наполнено хранилище путем обучения. Мощность вершин вычисляется лучше, типовые ошибки, описанные ранее, минимизируются. Схематически принцип действия такого подхода изображен на рисунке 1.

Рис. 1 - Идея адаптивной оптимизации запросов

Пусть имеется некоторый запрос с несколькими условиями, строится план выполнения запроса. Рассматривается совокупность условий в вершине плана и ее поддереве. По этим условиям и их маргинальным селективностям требуется предсказать их совместную селективность, а в последствии определить мощность. Маргинальные селективности вычисляются стандартными статистическими средствами СУБД, такими как одномерные гистограммы.

В методе используется понятие классов эквивалентности. Два выражения, содержащих условия, признаются эквивалентными, если они различаются только в константах. Например, выражения $t.age < 25$ и $t.age < 26$ эквивалентны, т. к. оба имеют структуру вида $t.age < const$, где t – некоторое отношение. Исключением являются классы эквивалентности, содержащие более двух переменных: в таком случае попарные предложения могут быть разными при одних и тех же ограничениях, поэтому такие выражения рассматриваются как один оператор равенства, без вычисления селективности [3]. Таким образом, формируя из условий запроса набор классов эквивалентностей, создается описание структуры запроса, используемое далее.

Вместо значения констант метод оперирует маргинальными селективностями, вычисленными по гистограмме. По селективности можно восстановить значение константы.

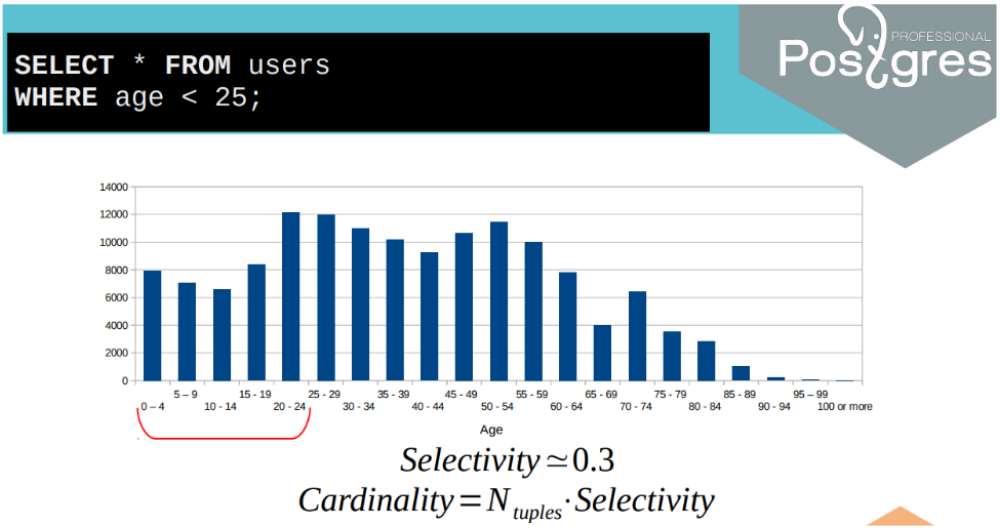

Рис. 2 пример гистограммы отношения user по атрибуту age

1.1.2 Построение пространства признаков

В контексте машинного обучения, построение модели, способной предсказывать численную величину на основе набора признаков – это задача регрессии.

Машинное обучение используется для поиска закономерностей в данных, данные – это набор объектов. В нашем случае, объектом является совокупность условий в одной вершине плана. Признаками являются маргинальные селективности условий вершины плана. Скрытое значение, которое требуется предсказать – мощность вершины плана.

План запроса – сложное бинарное дерево, которое непросто использовать в машинном обучении, поэтому вершину плана и ее поддерево требуется преобразовать в набор чисел из R^n. Манипуляции с выделением классов эквивалентностей, вычислением маргинальных селективностей, описанные ранее, как раз проводились для формирования пространства признаков.

Вектор признаков $\vec{f}$ определяется как набор маргинальных селективностей, а пространство признаков – как пара $< \text{хеш пространства}, \vec{f} >$. Хеш пространства вычисляется на основе информации о участвующих в запросе отношениях, выделенных классах эквивалентности и выражениях, то есть два одинаковых запроса, отличающихся только в константах, получат одинаковый хеш, но различные значения компонент вектора признаков.

Формирование пространства признаков для каждой структуры запроса таким путем обусловлено тем, что в методе не предлагается никаких техник для выделения семантики из запроса, соответственно элементы, формирующие пространство признаков, воспринимаются как атомы. Все, что мы можем с ними сделать – сказать, равны два атома или нет. Естественно, недостаток такого действия в разрастании числа различных пространств признаков с поступлением более структурно разнообразных запросов.

Рассмотрим пример. Дана следующая схема БД (рисунок 3)

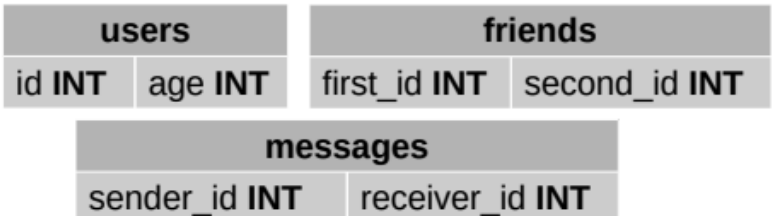

Рис. 3 схема базы данных

Поступает запрос, по которому оптимизатор строит вариант физического плана выполнения (рисунок 4):

```
SELECT * FROM users, messages, friends
WHERE users.age > 25 AND
      users.id > 1000 AND
      users.id = messages.sender_id AND
      users.id = friends.first_id AND
      messages.receiver_id = friends.second_id
```

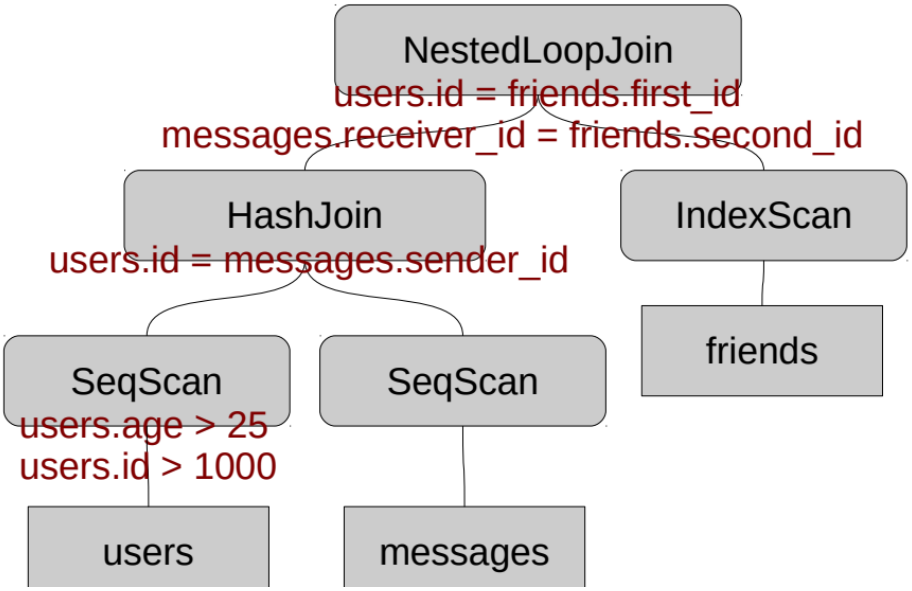

Рис. 4 План запроса

Производится выделение условий и формирование классов эквивалентностей:

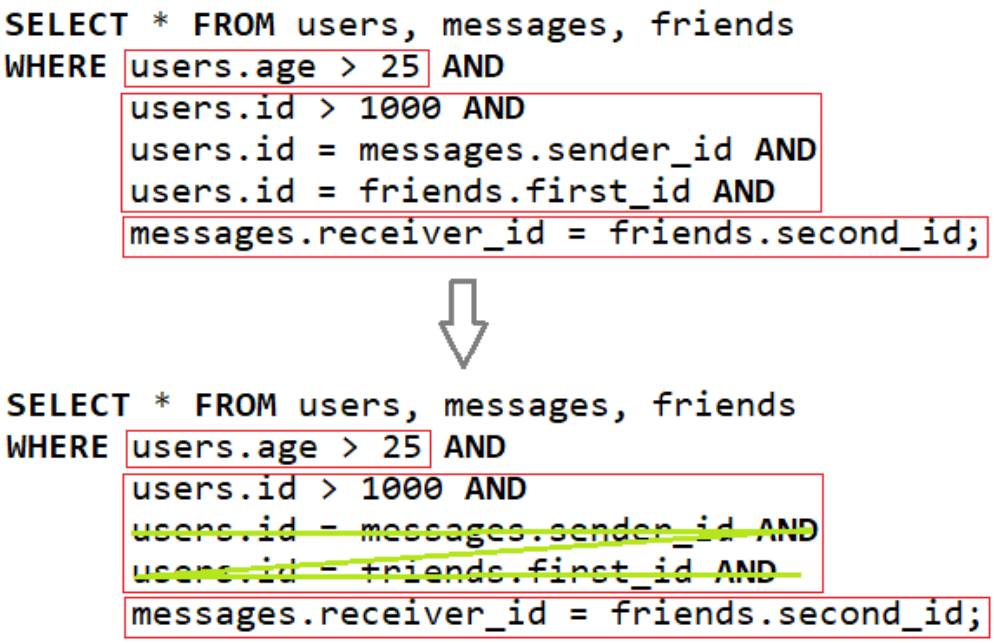

Рис. 5 Выделение условий

Таблица 1. Выделенные классы эквивалентности

Имя класса	**Атрибуты**
eclass_0	`users.age`
eclass_1	`users.id, messages.sender_id, friends.first_id`
eclass_2	`messages.receiver_id, friends.second_id`

Для выделенных классов вычисляются маргинальные селективности по гистограммам, значения констант больше не рассматриваются:

Условие в классе эквивалентности	**маргинальная селективность**
$users.age > const$	0.78
$users.id > const$	0.97
$messages.receiver\_id = friends.second\_id$	0.0001

Формируется вектор признаков: $\vec{f} = (0.78, 0.97, 0.0001)$,

определяется хеш и само пространство признаков: $< 993059063, \vec{f} >$.

Хеш может быть использован для проверки, был ли подобный запрос уже обработан или нет. Вектор признаков используется в качестве входных данных для машинного обучения, в ответ мы получаем предсказанную оценку мощности.

1.2 Авторегрессионные глубокие модели без учителя

Основная проблема модуля оценки селективности, влекущая ошибки при её вычислении - использование следующего ряда эвристических предположений [1]:

- независимость столбцов в таблицах;
- предположение о том, что данные в каждом столбце гистограмм, построенных на основе имеющихся данных, распределены равномерно.

Так или иначе, большинство предлагаемых методов для улучшения оценки селективности на основе машинного обучения, пытаются нивелировать эти эвристики, либо и вовсе отказаться от их использования при вычислении селективности.

Рассматриваемый в данном разделе метод [4], предлагает использовать подход обучения без учителя для глубоких авторегрессионных моделей. Авторегрессионная модель – такая модель, в которой значения временного ряда в данный момент линейно зависят от предыдущих значений этого же ряда.

Каким образом идея подобных моделей сопоставляется с процессом вычисления селективности, будет описано в этой главе. Также приводятся используемые техники векторизации предикатов для поступления в модель. Будут рассмотрены как точечные запросы (предикаты равенства, и т.п.), так и запросы, где запрашиваемые значения могут располагаться в промежутке (range queries).

1.2.1 Математический аппарат

Для начала приведен математический аппарат для лучшего понимания метода.

Пусть T – отношение с доменами атрибутов $\{A_1, \dots, A_n\}$. Оценка селективности – вычисление доли кортежей из T, удовлетворяющих предикату $\theta: A_1 \times \dots \times A_n \to \{0, 1\}$. Селективность в таком случае определена так:

$$sel(\theta) \coloneqq \frac{|\{x \in T: \theta(x) = 1\}|}{|T|}.$$

Совместное распределение данных отношения T определено как

$$\mathrm{P}(a_1, \dots, a_n) \coloneqq \frac{f(a_1, \dots, a_n)}{|T|},$$

где $f(a_1, \dots, a_n)$ – число появлений кортежа $(a_1, \dots, a_n)$ в T. Это формирует валидное распределение, т.к. интегрирование этого выражения по доменам атрибутов в результате даст 1. Отсюда следует, что вычисление селективности эквивалентно интегрированию:

$$sel(\theta) = \sum_{a_1 \in A_1} \dots \sum_{a_n \in A_n} \theta(a_1, \dots, a_n) \cdot \mathrm{P}(a_1, \dots, a_n).$$

При вычислении совместной селективности на таблицах с большим числом столбцов, СУБД старается упростить вычисления совместного распределения через его факторизацию (разложение) в более простой низкоразмерный вид $\hat{P} \approx P$.

Классические одномерные гистограммы, используемые в СУБД, предполагают простейшую факторизацию, основанную на эвристике о независимости столбцов: $\hat{P}(A_1, \dots, A_n) \approx \prod_{i=1}^{n} \hat{P}(A_i)$. Значение $\hat{P}$ может быть

материализовано как раз в виде гистограммы, которая дешево вычисляется и хранится в памяти.

При таком классическом подходе, расчет приближенной совместной селективности сводится к вычислению поколоночных селективностей и их перемножению:

$$sel(\theta) \approx \left(\sum_{a_1 \in A_1} \theta_1(a_1) \cdot \widehat{\mathrm{P}}(a_1) \right) \times \ldots \times \left(\sum_{a_n \in A_n} \theta_n(a_n) \cdot \widehat{\mathrm{P}}(a_n) \right),$$

где θ_i – предикат θ спроецированный на атрибут a_i.

В рассматриваемом методе, полагается, что наилучшая факторизация совместного распределения может быть достигнута при использовании теоремы об умножении вероятностей из теории вероятностей:

$$\widehat{\mathrm{P}}(A_1, \ldots, A_n) = \widehat{\mathrm{P}}(A_1)\widehat{\mathrm{P}}(A_2|A_1) \ldots \widehat{\mathrm{P}}(A_n|A_1, \ldots, A_{n-1}) \qquad (1)$$

Данное представление совместного распределения и будет использоваться в методе. Ключевой момент – отсутствие эвристики о независимости столбцов, чего мы и пытаемся достичь. Более того, вычисление множителей этого выражения и будет производиться авторегрессионной моделью.

1.2.2 Глубоки авторегрессионные модели

В качестве вариантов авторегрессионных моделей предлагаются следующие три модели:

- MADE;
- ResMADE;
- Transformer.

Первые две представляют собой модификации маскированного многослойного перцептрона, Трансформер – маскированная нейронная сеть с

механизмом внимания. Существуют работы, где эти модели использовались как раз в задачах оценки распределений, в том числе и условных.

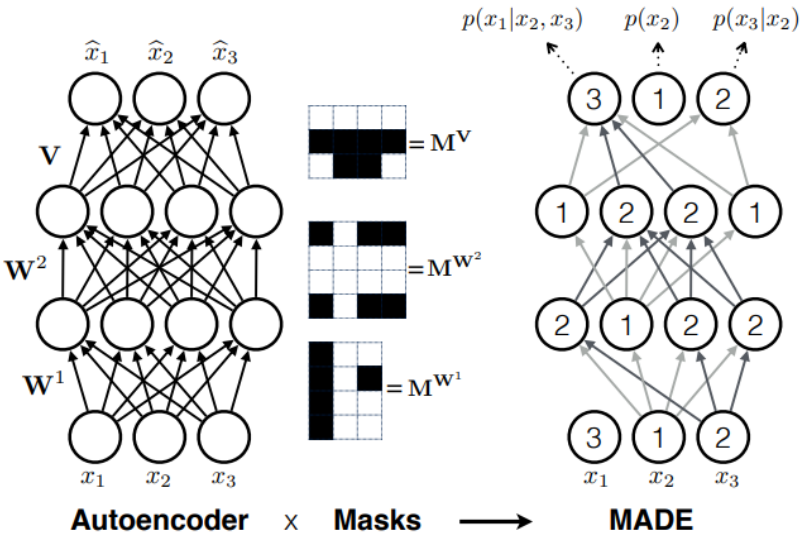

Рис.5 – Концептуальная идея MADE [5]*: на выходе получены условные распределения* $p(x_3|x_2), p(x_1|\, x_2, x_3)$*. Маски используются для достижения свойства авторегресионности. Это позволяет получить на выходе модели значение, к примеру* $p(x_3)$*, зависящее только от конкретных предшествующих ему входов –* x_1*,* x_2*.*

Оценка точечного распределения $\widehat{P}(x)$

Модель может продуцировать оценку точечного распределения $\hat{P}(x)$ после обучения на множестве *n*-мерных кортежей $T = \{x_1, \dots x_n\}$ с применением метода максимального правдоподобия (*maximum likelihood objective*).

Оценка условного распределения $\widehat{P}(x_i|x < i)$

Дополнительно, авторегрессионнаяная модель предоставляет доступ ко всем условным распределениям, представленным в правиле умножения вероятностей (1). Пусть дан входной кортеж $x = \{x_1, \dots x_n\}$. Возможно получить от модели n оценок условных распределений $\hat{P}(x_i|x < i)$. Модель может быть сконструирована для использования любого порядка атрибутов (x_1, x_2, x_3), или (x_2, x_1, x_3) и т.д. В рассматриваемой работе используется порядок слева-направо.

Одна из преимуществ авторегрессивных моделей заключается в том, что доступ к условным распределениям критически эффективен для оценки запросов с промежутком значений (range queries).

1.2.2 Применение к реляционным данным

В общем смысле авторегрессионную модель M можно описать следующим образом:

$$M(x) \rightarrow [\ \hat{P}(X_1), \hat{P}(X_2|x_1), \ldots, \hat{P}(X_n|x_1, \ldots, x_n - 1)\ ]$$

На вход поступает один кортеж, на выходе получается список условных плотностей распределений, каждый элемент списка является распределением $i-$го атрибута с учетом условий зависимости от предыдущих атрибутов.

Для лучшего объяснения идеи, допустим, что для каждого столбца i в таблице имеется своя компактная нейронная сеть, вход которой агрегирует информацию о значениях в предыдущих столбцах x < i. Рассмотрим пример:

Имеется таблица (год, город, оценка). На вход модели поступает кортеж $<$ Портлэнд, 2017, 10 $>$. Сначала, специфический для столбца кодировщик $\mathrm{E}_{\mathrm{col}}()$ трансформирует каждое значение атрибута в векторное представление, годное для поступления в нейронную сеть:

$$[\mathrm{E}_{\text{город}}(\text{Портлэнд}), \mathrm{E}_{\text{год}}(2017), \mathrm{E}_{\text{оценка}}(10)].$$

Затем, входные данные поступают в «поколонучную» нейронную сеть $M_{\text{столбец}}$:

$$0 \rightarrow M_{\text{город}}$$

$$\mathrm{E}_{\text{город}}(\text{Портлэнд}) \rightarrow M_{\text{год}}$$

$$\oplus \left(\mathrm{E}_{\text{город}}(\text{Портлэнд}), \mathrm{E}_{\text{год}}(2017)\right) \rightarrow M_{\text{оценка}}$$

Оператор $\oplus$ описывает агрегацию информации из нескольких закодированных атрибутов. На практике, это может быть простая конкатенация векторов или как-раз механизм внимания авторегрессионной модели.

Первый выход $M_{\text{город}}$ не зависит ни от какого атрибута.

Второй выход $M_{\text{год}}$ зависит только от значения атрибута *город*.

Третий выход $M_{\text{оценка}}$ зависит от обоих атрибутов: *город, год.*

Таким образом, три выхода можно интерпретировать так:

$$[\hat{P}(\text{город}), \hat{P}(\text{год} \mid \text{город}), \hat{P}(\text{оценка} \mid \text{город}, \text{год})].$$

Описанное выше демонстрирует принцип вычисления условных плотностей распределений столбцов, понятно, уже без классической эвристики о том, что столбцы независимы. Внедрение такого подхода в оптимизатор должно значительно повысить точность оценки селективности или кардинальности, которая вычисляется на основе данных о плотности распределения столбцов.

В процессе обучения модели, в качестве функции потерь, используется кросс-энтропия между реальным распределением P и оценкой от модели $\hat{P}$, рассчитанная по всем кортежам отношения T. Она может быть использована в стандартном оптимизаторе машинного обучения – градиентном спуске.

$$\mathcal{H}(P, \hat{P}) = -\sum_{\boldsymbol{x} \in T} P(\boldsymbol{x}) \log \hat{P}(\boldsymbol{x}) = -\frac{1}{|T|} \sum_{\boldsymbol{x} \in T} \log \hat{P}(\boldsymbol{x})$$

$H(P, \hat{P}) - H(P)$ – это энтропийный разрыв. Низкое значение этого показателя свидетельствует о высоком качестве оценки плотности распределения и используется как мониторинговая метрика в процессе и после обучения.

1.2.3 Конструирование оценочного модуля

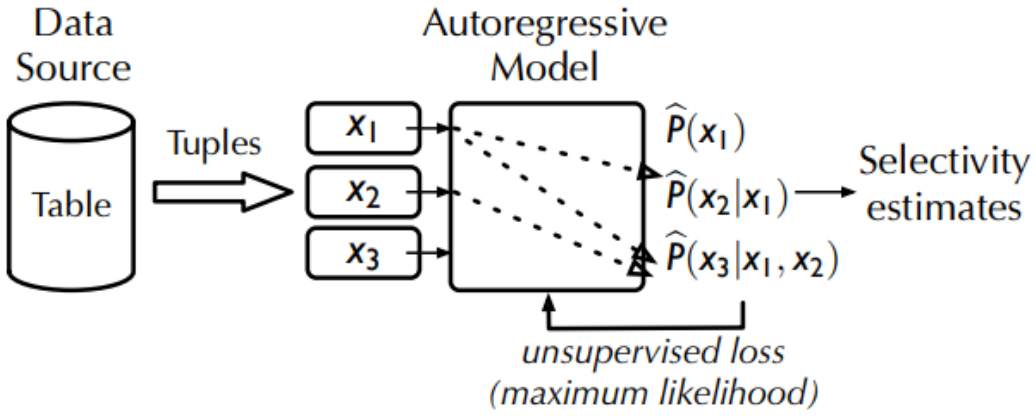

Рис.6 - Концептуальная схема работы оценочного модуля

Предполагается, что обучение модели производится периодически, во время низкой активности СУБД.

Стратегия кодирования (векторизации)

Для кодирования столбцов с малым по размеру доменом используется широко известный подход *one-hot encoding.*

В случае, если размерность домена признана высокой, используется встраивание (embedding). Здесь используется матрица размерности $R^{|A_i| \times h}$ инициализированная случайным образом. Кодировщик $E_{\text{столбец}}()$ просто ищет строку в матрице. Например: $E_{\text{год}}(4) \rightarrow$ 4 строка матрицы – h-размерный вектор. Матрица постепенно обновляется в процессе градиентного спуска при обучении модели. Такой тип кодировки хорошо подходит для доменов со значимой семантической разницей.

Стратегия декодирования

В случае с малоразмерными доменами, нейронная сеть выделяет полноразмерный выходной слой для вычисления распределения $\hat{P}(X_i|x < i)$, - $|A_i|$ - размерный вектор, который используется для вычисления селективности. Например, для столбца с городами с тремя различными значениями в домене,

выходное значение может быть таким: [Москва=0.2, Портлэнд=0.5, Вайкики=0.3]. В процессе оптимизации, функция потерь минимизирует расхождение между выходом сети и реальным распределением данных.

Для декодирования столбцов с большим доменом, используют предложенный в работе e*mbedding reuse*. На выходе модели получается h-размерный вектор $H \subseteq R^{1\times|A_i|}$. Затем вычисляется HE_i^T, где $E_i \subseteq R^{|A_i|\times h}$ - уже существующая в памяти embedding-матрица, содержащая столбец i. Полученный результат нормализуется и интерпретируется как оценка распределения.

1.2.4 Запросы к оценочному модулю

Предикаты равенства

Если значения в запросе указаны для всех столбцов, оценка для предиката достаточно проста. Такой запрос имеет следующую форму: $P(X_1 = x_1, \dots, X_n = x_n)$ и требуется только один проход кортежа $(x_1, \dots, x_n)$ через сеть, чтобы получить последовательность условных вероятностей, которые затем перемножаются:

$$[\, P(X_1 = x_1),$$
$$P(X_2 = x_2 \mid X_1 = x_1),$$
$$\dots,$$
$$P(X_n = x_n \mid X_1 = x_1, X_2 = x_2, \dots, X_{n-1} = x_{n-1})].$$

Ранжированные предикаты

В случае, если запрашиваемый в предикате промежуток R достаточно мал, можно просто просуммировать плотности:

$$sel(X_1 \in R_1, \dots, X_n \in R_n) \approx \sum_{X_1 \in R_1} \dots \sum_{X_n \in R_n} \hat{P}(x_1, \dots, x_n).$$

Если же запрашиваемый в предикате промежуток R слишком большой, предлагается использовать новую технику аппроксимации *progressive sampling.*

Подход *progressive sampling* базируется на предположении, о том, что не все точки из заданного промежутка значений R важны для вычисления селективности. Метод близок к идее метода Монте-Карло и старается найти центр масс плотности многомерного распределения в рамках заданных промежутков с помощью случайно-подобранных из данных промежутков чисел.

Рассмотрим пример:

Дан запрос на вычисление селективности ранжированного предиката:

$$sel(\text{возраст} \in R_1, \text{зарплата} \in R_2).$$

Вычисление $s_1 \coloneqq \hat{P}(\text{возраст} \in R_1)$ можно свести к вычислению за несколько шагов среднего арифметического значения $\hat{P}(\text{возраст} = y)$ с использованием авторегрессионной модели, где y – случайное число из R_1, вычисляемое заново на каждом шаге. Аналогично оценивается и следующая часть предиката, но уже с учетом аппроксимированной оценки s_1. Затем вычисляется и оценка совместной плотности распределения:

$$\hat{P}(\text{возраст} \in R_1) \cdot \hat{P}(\text{зарплата} \in R_2 \mid \text{возраст} = s_1)$$

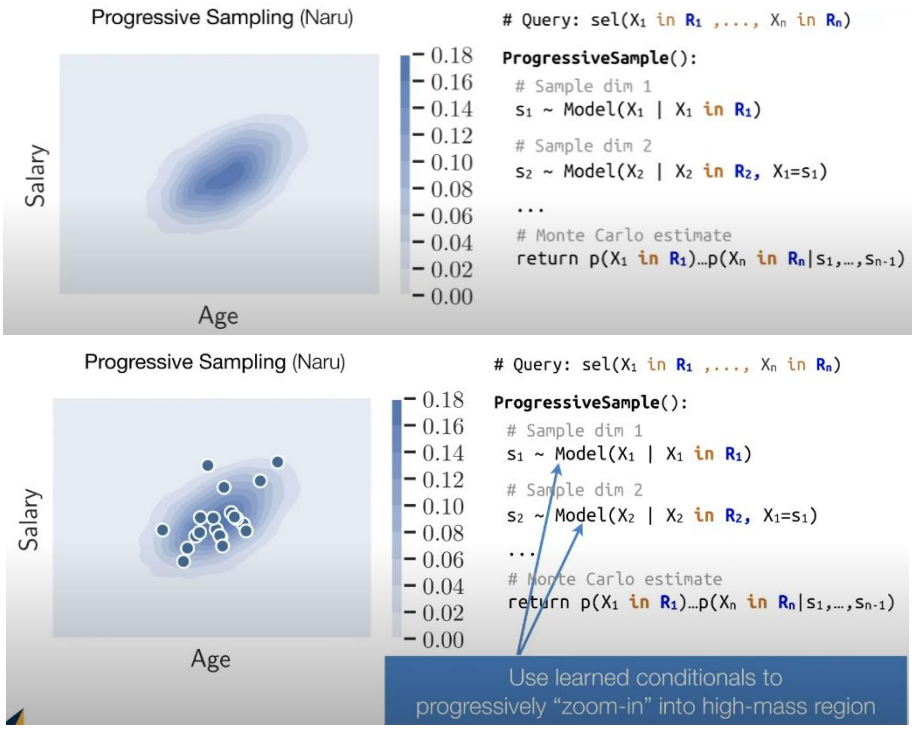

Рис 7. Progressive sampling [6]

2. Экспериментальный раздел

2.1 Исходные данные и условия эксперимента

Для экспериментов в этой работе используются две базы данных. StrongCor – это синтетическая база данных, используемая специально для определения качества оценки выборочности условий. Запросы в наборе данных StrongCor построены таким образом, что план содержит только одну вершину, поэтому в этом экспериментах с этими данными одна итерация обучения, один запрос, один план выполнения запроса и одна вершина плана являются синонимами.

TPC–H. Для экспериментов в данной работе используется известный тест производительности для СУБД TPC–H [28]. Этот тест создан для оценок производительности СУБД и аппаратного обеспечения, поэтому максимально приближен к реальной нагрузке на СУБД.

Тесты производительности для СУБД TPC–H содержит генератор базы данных различных размеров, а также генератор запросов. Запрос генерируется методом подстановки случайно сгенерированных значений в один из выбираемых пользователем шаблонов запросов. Всего тесте производительности TPC–H есть 22 шаблона запросов. Таким образом моделируется реальные условия, когда все запросы к базе данных различны, но при этом есть ограниченное и небольшое количество структур этих запросов.

Эксперименты проводились на ноутбуке со следующими характеристиками:

- Core i7
- 8Gb RAM
- Linux Ubuntu 20.04 LTS

В качестве реляционной СУБД использовалась СУБД с открытым исходным кодом PostgreSQL версии 9.6dev.

2.1 Оценка авторегрессионных моделей

В качестве метрики ошибки используется мультипликативная ошибка (“Q-error”):

$$Error := \frac{max(\text{вычисленное значение}, \text{актуальное значение})}{min(\text{вычисленное значение}, \text{актуальное значение})}.$$

В таблице 1 в качестве результатов экспериментов в статье приводятся точности оценки селективности с использованием авторегрессионных моделей и других, как классических подходов, так и оценочных модулей на основе иных методов глубокого обучения:

- Оценочные модули классических СУБД: Postgres, DBMS-1: обе СУБД опираются на стандартные методы одномерных гистограмм. Postgres также и на эвристику о независимости столбцов, DBMS-1 в добавок подсчитывает число уникальных значений в столбцах.
- MSCN – Mult-set Convolutional Network;
- Indep – простой оценочный модуль на основе эвристики о независимости столбцов. Вычисляется максимально точная селективность для каждого столбца независимо, затем эти значения перемножаются.
- MHIST – многомерные гистограммы. Наиболее точный подход среди методов, использующих гистограммы.
- BayesNet - Байесовская сеть.
- KDE – Kernel Density Estimator.
- KDE-superv – Kernel Density Estimator с учителем
- Naru – авторегрессионная модель, описанная в данной работе. Число, сопутствующее названию модели, описывает количество шагов, использованное при выполнении *progressive sampling*

Таблица 1. Ошибка оценки селективности. Ошибки показаны в перцентилях, рассчитанных на основе 2000 запросов.

Estimator	High ((2%, 100%])				Medium ((0.5%, 2%])				Low (≤ 0.5%)			
	Median	95th	99th	Max	Median	95th	99th	Max	Median	95th	99th	Max
DBMS-1	1.75	5.25	7.77	9.12	3.93	13.6	19.9	31.3	8.63	176	636	4737
Sample	**1.02**	**1.06**	**1.09**	**1.11**	**1.04**	**1.14**	**1.18**	**1.23**	1.18	49.3	218	696
KDE	105	$2 \cdot 10^5$	$5 \cdot 10^5$	$8 \cdot 10^5$	347	$6 \cdot 10^4$	$8 \cdot 10^4$	$8 \cdot 10^4$	224	$1 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$
KDE-superv	1.99	7.97	14.5	33.6	2.04	8.44	17.0	49.7	2.76	74.5	251	462
MSCN-base	1.14	1.27	1.36	1.48	1.15	1.55	2.26	57.7	2.05	20.3	84.1	370
MHIST	1.54	5.24	11.2	$1 \cdot 10^4$	2.22	10.5	30.3	$3 \cdot 10^4$	6.71	792	4170	$8 \cdot 10^4$
BayesNet	1.15	1.75	2.05	2.70	1.31	2.70	4.95	47.6	1.67	14.8	78.0	998
Naru-1000	**1.02**	1.11	1.18	1.37	1.05	1.17	1.27	1.40	1.10	1.71	3.01	185
Naru-2000	**1.02**	1.10	1.16	1.28	1.05	1.17	1.27	1.38	**1.09**	1.66	3.00	58.0
Naru-4000	**1.02**	1.10	1.17	1.21	**1.04**	1.15	1.27	1.36	**1.09**	**1.57**	**2.50**	**4.00**

Таблица 2. Сравнение предлагаемых авторегрессионных моделей. FLOPs – число операций с плавающей точкой, требуемых для прямого прохода одного входного кортежа. Ent. Gap – энтропийный разрыв (раздел 1.3.1).

	Params	FLOPs	Ent. Gap	Max Error
MADE	3.3M	6.7M	0.59	8.0×
ResMADE	3.1M	6.2M	0.56	8.0×
Transformer	2.8M	35.5M	0.54	8.2×

Таблица 3. Сравнение размеров нейронной сети MADE и их влияние на энтропийный разрыв

Architecture	Size (MB)	Entropy gap, 5 epochs
32 × 32 × 32 × 32	0.6	4.23 bits per tuple
64 × 64 × 64 × 64	1.1	2.25 bits per tuple
128 × 128 × 128 × 128	2.7	1.01 bits per tuple
256 × 256 × 256 × 256	3.8	0.84 bits per tuple

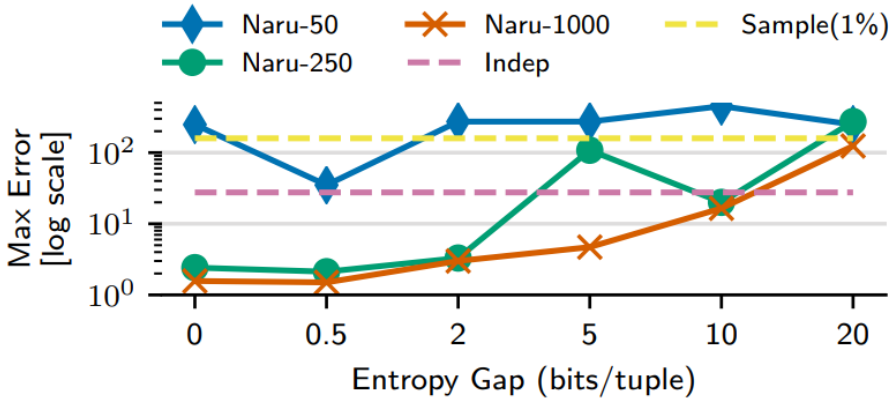

Рис. 8 – Зависимость значения энтропийного разрыва от числа шагов в progressive sampling (50, 250, 1000)

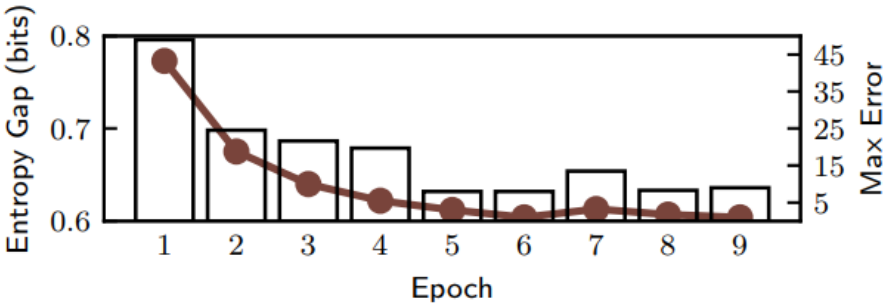

Рис. 9 – Время обучения и качество. Столбцы показывают максимальную ошибку в оценке селективности

2.2 Исследование характеристик метода адаптивной оценки

На графике на рисунке 10 по оси Y отложена ошибка предсказания. Как мы видим из графика результата эксперимента, меньше всего ошибку предсказания допускает метод градиентного бустинга, однако метод k-ближайших соседей за малое число запросов начинает показывать хорошую точность, это говорит о быстрой обучаемости метода, что на практике будет иметь большее значение.

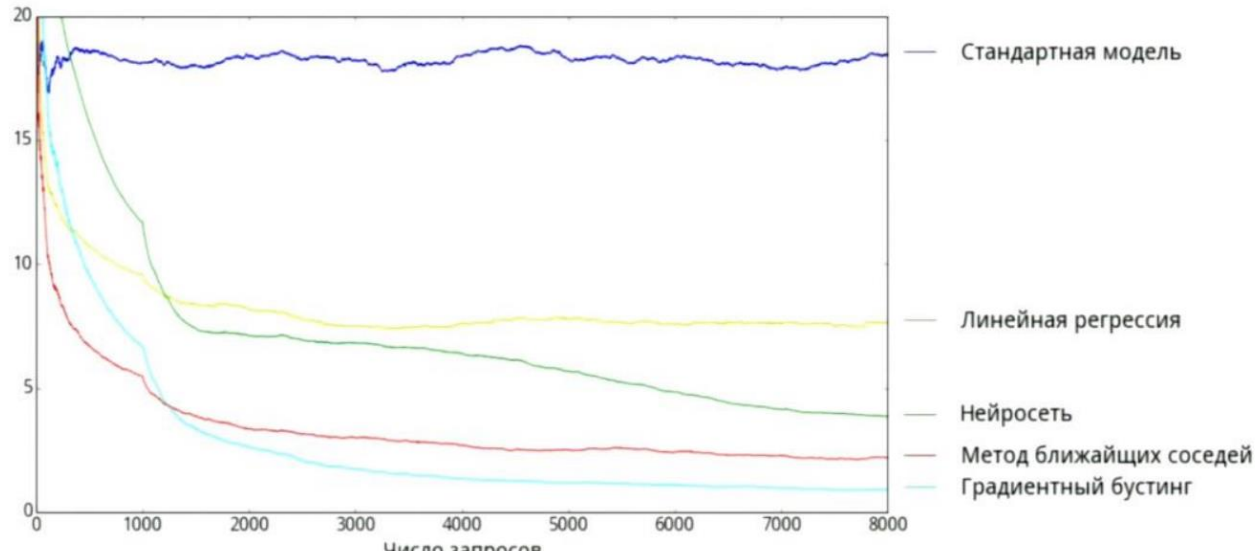

Рис. 10 - зависимость ошибки от метода МО

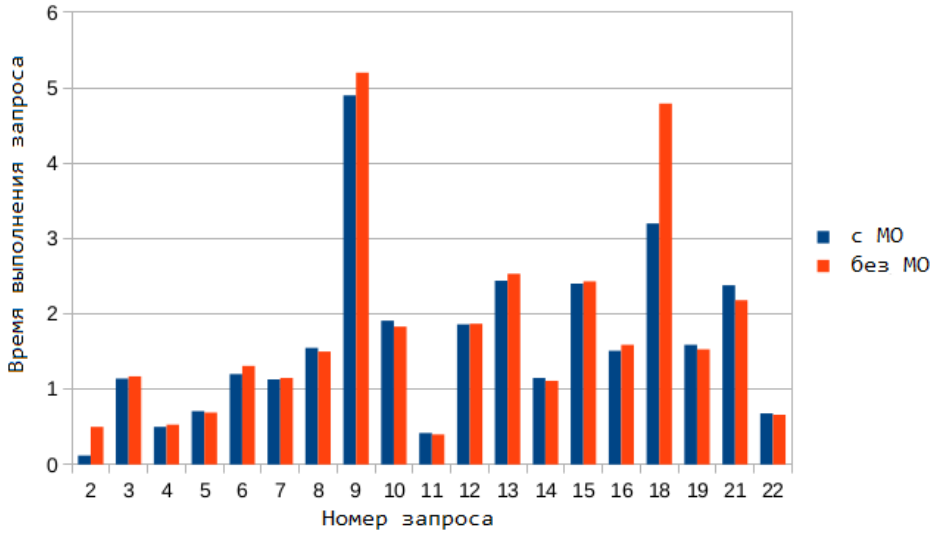

Рис. 11 - Сравнение времени выполнения запросов с использованием MO

ЗАКЛЮЧЕНИЕ

Модели глубокой авторегрессии показывают достаточно точные оценки селективности и превосходят некоторые семейства оценочных модулей по точности. Один из плюсов модели без учителя для оптимизатора запросов СУБД – более эффективное обучение, т.к. нет необходимости выполнять запросы для сбора результатов, требуется только прочитать имеющиеся данные.

Как видно из таблицы 2, по скорости вычислений более предпочтительна архитектура ResMADE, самый низкий энтропийный разрыв наблюдается у архитектуры «Трансформер», но ее вычисления – самые трудоемкие.

Из таблицы 3 видно, что увеличение размерности модели MADE влечет улучшение точности оценки селективности.

При повышении числа шагов в технике *progressive sampling*, ошибка падает достаточно сильно, как видно на рисунке 4.

Модели глубокой авторегрессии могут достаточно хорошо справляться с повышением точности оценки селективности. Адаптивная оценка селективности, предложенная в первом разделе первой главы, также показывает положительные результаты, но способ построения пространства признаков в ней обладает недостатками – не учитывается семантика запросов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Postgres Professional. Адаптивная оптимизация запросов в Postgres Pro, 2020 [электронный ресурс]. Режим доступа: [https://habr.com/ru/company/postgrespro/blog/508766/];
2. Postgres Professional. Применение машинного обучения для увеличения производительности PostgreSQL, 2015: [электронный ресурс]. Режим доступа: [https://habr.com/ru/company/postgrespro/blog/273199/]
3. Oleg Ivanov. Postgres Professional. Adaptive Cardinality Estimation, 2017: [электронный ресурс]. Режим доступа: [https://arxiv.org/pdf/1711.08330.pdf]
4. Zongheng Yang, Eric Liang, Amog Kamsetty, Chenggang Wu, Yan Duan, Xi Chen, Pieter Abbeel, Joseph M. Hellerstein, Sanjay Krishnan, Ion Stoica. UC Berkeley, University of Chicago. 2019. Deep Unsupervised Cardinality Estimation, Электронный ресурс: https://arxiv.org/pdf/1905.04278v2.pdf
5. Mathieu Germain, Karol Gregor, Iain Murray, Hugo Larochelle. 2015. MADE: Masked Autoencoder for Distribution Estimation. Электронный ресурс: https://arxiv.org/pdf/1502.03509.pdf
6. Deep Unsupervised Cardinality Estimation, Youtube-видео презентации, 2020. Электронный ресурс: https://www.youtube.com/watch?v=u2glA-S1AEs