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

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

«Исследование метода замещения страниц в разделяемом кэш буфере postgres»

Студент ИУ7-43М (Группа) _____ (Подпись, дата) Мицевич М. Д. (И. О. Фамилия)

Руководитель НИР _____ (Подпись, дата) Тассов К. Л. (И. О. Фамилия)

2025 г.

РЕФЕРАТ

Расчетно-пояснительная записка к научно-исследовательской работе содержит 18 страниц, 7 иллюстраций, 0 таблиц, 6 источников.

Научно-исследовательская работа представляет собой исследование метода замещения страниц в разделяемом кэш буфере PostgreSQL. Проведено исследование разработанного метода и выявлена зависимость коэффициентов попадания и совпадения от количества обращений к страницам на тестовой выборке. Проведено сравнение полученных результатов и значений этих метрик для существующих аналогов.

Разработанный метод замещения страниц может быть использован в СУБД Postgres. Использование метода позволит повысить коэффициент попадания в разделяемом кэш буфере, что должно привести к уменьшению времени отклика системы.

Ключевые слова: страница, замещение, кэш буфер, PostgreSQL.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные базы данных, такие как PostgreSQL, сталкиваются с постоянно растущими требованиями к производительности и эффективности управления ресурсами. Одним из ключевых аспектов работы базы данных является управление памятью, включая работу с разделяемым кэш буфером [1].

Загрузка данных с диска занимает гораздо больше времени, чем из оперативной памяти, поэтому современные системы управления базами данных используют область в оперативной памяти в качестве буфера для кэширования недавно просмотренных страниц, чтобы в будущем запросы к страницам в буфере выполнялись быстрее. Обычно буфер делится на части одинакового размера, где каждая часть может содержать страницу. Когда транзакция базы данных запрашивает страницу, которая в данный момент не хранится в буфере, она должна быть загружена в буфер. Если для кэширования этой страницы больше нет места, то одна из страниц в буфере должна быть вытеснена, чтобы освободить место для новой запрашиваемой страницы. Выбор такой страницы важен для уменьшения задержки доступа. Если все время для вытеснения будет выбираться страница, к которой в скором времени опять произойдет обращение, то производительность может ухудшиться до случая, когда данные в основном берутся с диска.

Для выбора страницы, которую надо исключить из буфера, применяются различные эвристические алгоритмы. Все эти алгоритмы являются приближением оптимального алгоритма и не учитывают структуру конкретной рабочей нагрузки. Если алгоритм замещения страниц будет учитывать особенности рабочей нагрузки, то число операций чтения и записи на диск может быть снижено, что приведет к повышению производительности системы.

Помимо систем управления базами данных методы замещения страниц используются в операционных системах, аппаратном и программным кэше, а также в других местах, где присутствует два типа памяти, один из которых меньше по объему и быстрее по скорости доступа.

Целью данной работы является исследование метода замещения страниц в разделяемом кэш буфере postgres с использованием нейронных сетей.

Для достижения поставленной цели требуется выполнить следующие задачи:

— описать и спроектировать метод замещения страниц с использованием нейронных сетей;

— провести сравнение разработанного метода с существующими аналогами по коэффициентам совпадения и попадания.

1 Проектирование и исследование метода

Кодировщик запроса обращения к странице отвечает за скрытое представление атрибутов страницы, к которой происходит очередное обращение. Схема кодировщика запроса обращения к странице изображена на рисунке 1.1.

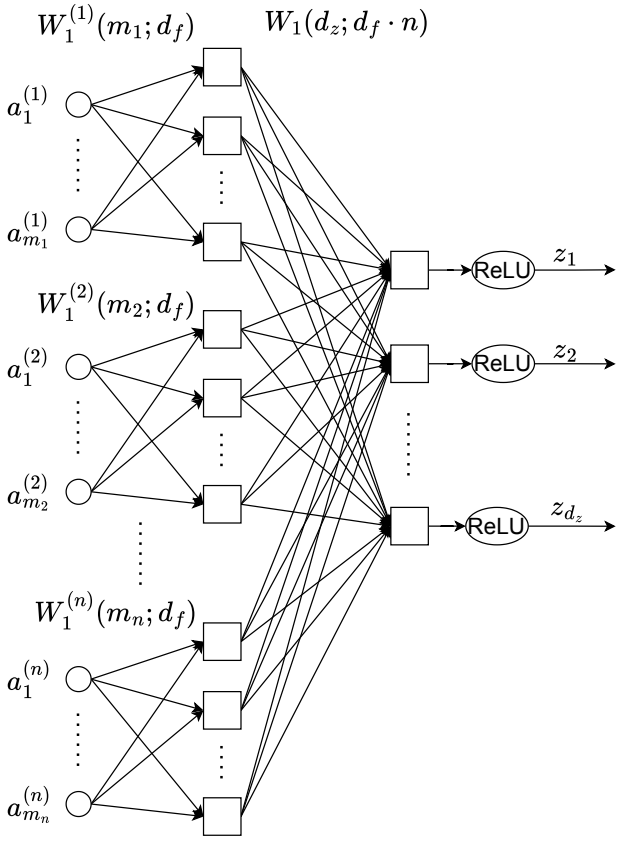

Рисунок 1.1 – Схема кодировщика запроса обращения к странице

На вход кодировщика поступают n атрибутов страницы. Каждый атрибут может иметь m_i возможных значений, где i – индекс атрибута. Каждый атрибут представляется в виде вектора $a^{(i)}$ размерности m_i. Для категориальных данных используется техника однозначного кодирования, а для числовых – применяется хэш функция и к полученному результату применяется техника однозначного кодирования. $W_1^{(i)}$ – матрица обучаемых весов для скрытого представления i-го атрибута. Вектор z – выходной вектор из сети. W_1 – матрица обучаемых весов, при помощи который получается результирующий вектор из скрытых представлений атрибутов сети. В качестве функции активации на последнем слое используется функция Relu. d_f и d_z являются настраива-

емыми параметрами, которые отвечают за число нейронов, отвечающий за скрытое представление каждого атрибута, и число нейронов на выходном слое соответственно.

Работу сети можно описать с помощью выражений 1.1 - 1.3:

$$f^{(i)} = a^{(i)} W_1^{(i)} i \in \{1; n\}, \tag{1.1}$$

$$f = [f^{(1)}, f^{(2)}, ..., f^{(n)}], \tag{1.2}$$

$$z = ReLU(W_1 f^T + l_1), \tag{1.3}$$

где f является конкатенацией векторов скрытых состояний атрибутов страницы, а l_1 – обучаемым вектором.

Кодировщик страниц в буфере нужен для скрытого представления каждой страницы в буфере. Схема кодировщика представлена на рисунке 1.2

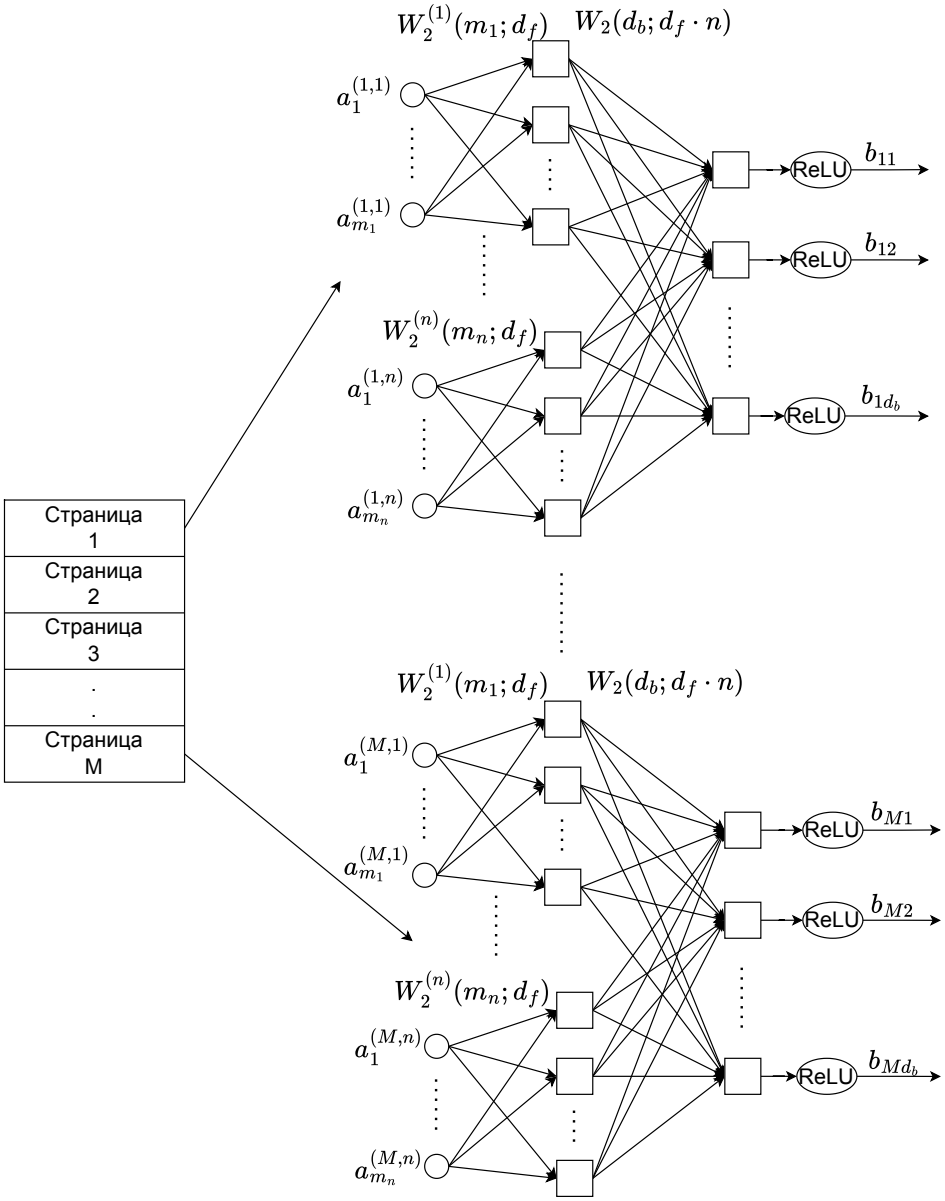

Рисунок 1.2 – Схема кодировщика страниц в буфере

На вход кодировщика поступают M страниц из буфера. Каждая страница представляется в виде n атрибутов. Процесс обработки атрибутов для каждой страницы такой же, как и в кодировщике запроса обращения к странице. Для каждой i-ой страницы в буфере вычисляется вектор b_i. d_b является настраиваемым параметром, который отвечает за размерность векторов b_i.

Для получения скрытого представления каждого атрибута и для вычисления закодированного представления страницы используется одни и те же матрицы весов $W_2^{(i)}$ и W_2 для всех страниц в буфере. За счет этого матрицы весов не привязаны к конкретной позиции страницы в буфере и истории

страниц на этой позиции. При обратном распространении ошибки влияние веса из матрицы W_2 будет учитываться для всех векторов b_i.

Обозначим результат работы сумматора нейрона на выходном слое как s_{ij}. Индексация в матрице s совпадает с матрицей b. Тогда для вычисления ошибки по весу w_{ij} из матрицы W_2 на ребре, которое соединяет j-ый нейрон из второго слоя и i-ый нейрон из выходного слоя, используется выражение 1.4:

$$\frac{\delta E}{\delta w_{ij}} = \sum_{k=1}^{M} \frac{\delta E}{\delta b_{ki}} \frac{\delta b_{ki}}{\delta s_{ki}} \frac{\delta s_{ki}}{\delta w_i j}, \tag{1.4}$$

где E – функция ошибки, $\frac{\delta E}{\delta b_{ki}}$ – ошибка полученная со следующего слоя.

Функционирование кодировщика определяется выражениями 1.5 - 1.7:

$$f^{(j,i)} = a^{(j,i)} W_2^{(i)} j \in \{1; M\} i \in \{1; n\}, \tag{1.5}$$

$$f^{(j)} = [f^{(j,1)}, f^{(j,2)}, ..., f^{(j,n)}], \tag{1.6}$$

$$b_j = ReLU(W_2 f^{(j)T} + l_2), \tag{1.7}$$

где $f^{(j)}$ – конкатенация скрытых представлений атрибутов для j-ой страницы в буфере, b_j – скрытое представление этой страницы, l_2 – вектор обучаемых весов.

Кодировщик истории обращений. Для обновления истории обращений используется сеть LSTM. На вход сети поступают результат работы кодировщика обращения к странице, предыдущий результат кодировщика истории обращений и предыдущее состояние ячейки.

Функционирование кодировщика определяется выражениями 1.8 - 1.13:

$$f_t = \sigma(W_f[h_{t-1}, z_t] + b_f), \tag{1.8}$$

$$i_t = \sigma(W_i[h_{t-1}, z_t] + b_i), \tag{1.9}$$

$$\hat{C}_t = \tanh(W_C[h_{t-1}, z_t] + b_C), \tag{1.10}$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t, \tag{1.11}$$

$$o_t = \sigma(W_o[h_{t-1}, z_t] + b_o), \tag{1.12}$$

$$h_t = o_t * \tanh(C_t), \tag{1.13}$$

где $[h_{t-1}, z_t]$ – конкатенация результата работы предыдущего слоя кодировщика истории и скрытого состояния, полученного из кодировщика обращения к странице, W_f и b_f – матрица и вектор обучаемых весов, f_t – результат работы фильтра забывания, i_t определяет, какие значения будут сохранены в ячейке, $\hat{C}_t$ – новые значения кандидатов на попадание в ячейку, W_i, W_C, b_i, b_c – матрицы и вектора обучаемых весов, C_t – новое состояние ячейки, C_{t-1} – состояние ячейки на прошлом шаге, h_t – результат работы текущего слоя, C_t – состояние ячейки, W_o и b_o – матрица и вектор обучаемых весов. Вектора h_t и C_t имеют размерность d_h, где d_h – настраиваемый параметр.

Модуль выбора страниц для замещения. На вход модуля поступают результаты работы кодировщика страниц в буфере и кодировщика истории обращений. Для выбора страницы, которая будет удалена из буфера используется указательная нейронная сеть с механизмом внимания [**vinyals2015pointer**].

Нейронные сети с механизмом внимания – это архитектуры, которые позволяют моделям динамически фокусироваться на наиболее релевантных частях входных данных при обработке информации. Этот подход нашел применения в областях обработки естественного языка, компьютерного зрения и других задач, где важно учитывать контекст и зависимости между элементами последовательности. В модуле выбора страниц для замещения контекстом является результат работы кодировщика истории, а элементами последовательности – результаты работы кодировщика страниц в буфере.

Указательные сети – архитектура сетей с механизмом внимания, предназначенная для решения задач, где выходные элементы представляют собой позиции в входной последовательности. В указательных сетях механизм внимания используется как указатель на один из элементов входной последовательности, а не для создания контекстного вектора, как в классических

моделях с механизмом внимания.

Схема модуля выбора страниц для замещения представлена на рисунке 1.3

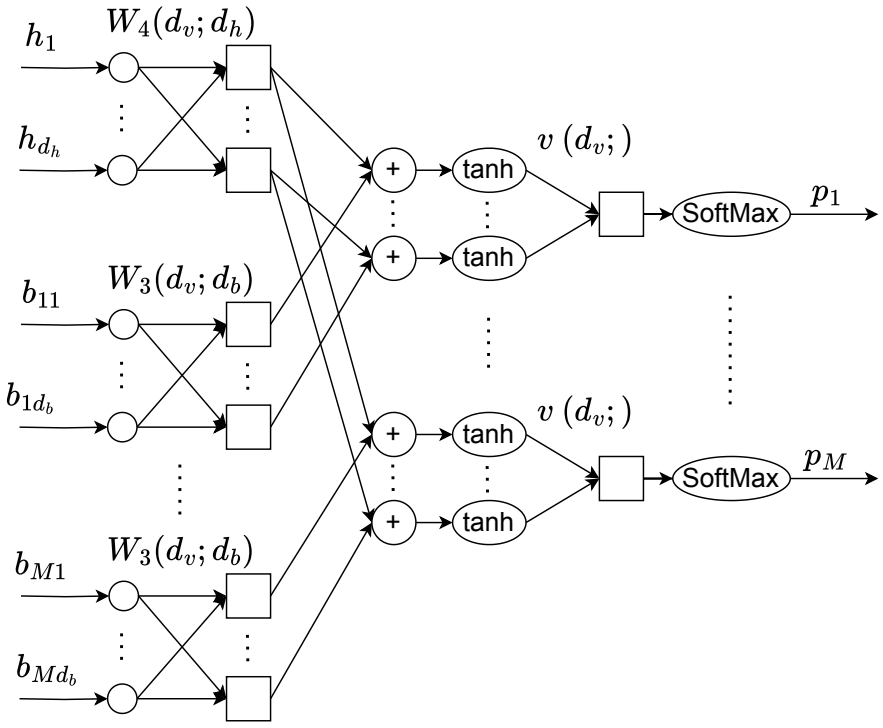

Рисунок 1.3 – Схема модуля выбора страниц для замещения

W_4 – матрица обучаемых весов, которая используется для преобразования вектора h, полученного из кодировщика истории в вектор контекста размерности d_v. d_v является настраиваемым параметром модели.

W_3 – матрица обучаемых весов, которая используется для преобразования закодированного состояния очередной страницы в буфере в вектор размерности d_v, который будет использован в функции внимания.

v – вектор обучаемых весов, который который используется при вычислении функции внимания.

Функционирование модуля выбора страниц для замещения определяется выражениями 1.14 - 1.16:

$$u_i = v * \tanh(W_3 b_i^T + W_4 h^T), i \in \{1; M\}, \tag{1.14}$$

$$p_i = SoftMax(u_i), \tag{1.15}$$

$$r = \arg\max_i p_i, \tag{1.16}$$

где M – число страниц в буфере, u_i – результат функции внимания для i-ой страницы в буфере, $\arg\max_i p_i$ – функция, которая возвращает индекс максимального элемента в последовательности, r – результат работы спроектированного метода замещения страниц.

1.1 Обучение и тестирование модели

Обучение модели проводилось на машине с процессором Intel Core i9-10900, 64 гигабайтами оперативной памяти и графической картой NVIDIA GeForce RTX 3080 с 16 гигабайтами памяти типа GDDR6.

В качестве оптимизатора функции потерь был выбран Adam, так как он автоматически адаптирует скорость обучения для каждого параметра в зависимости от его градиента, что позволяет более эффективно использовать скорость обучения и ускоряет сходимость.

Обучение модели проводилось на протяжении 100 эпох. После прохождения каждой эпохи веса модели сохранялись в файл и вычислялась точность модели на тестовой выборке. Была выбрана модель с наивысшей точностью на тестовой выборке.

Графики зависимостей точности модели на тестовой и обучающей выборках от номера эпохи обучения приведены на рисунке 1.4.

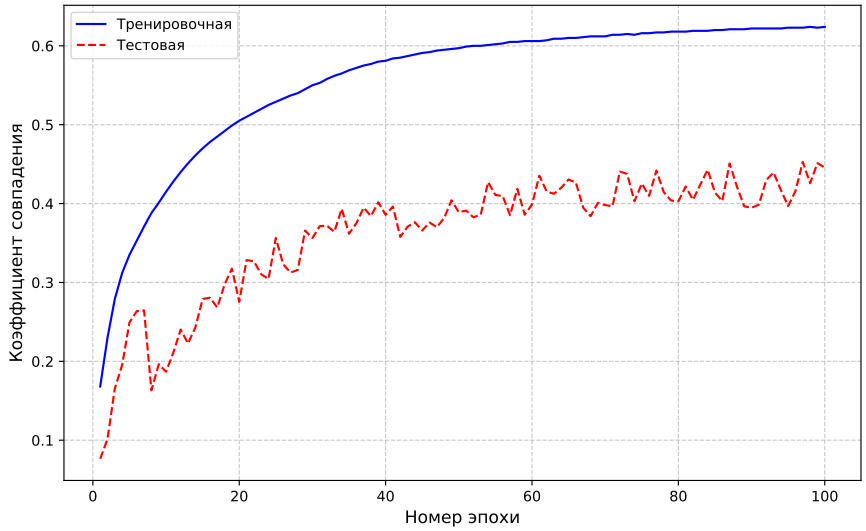

Рисунок 1.4 – Точность при обучении модели на тренировочной и тестовой выборках

Наивысшая точность модели была получена на 97 эпохе – 45.2 процента. Точность на обучающей выборке составила 63 процента.

1.2 Подбор параметров сети

Для оценки разработанного метода вводятся следующие метрики качества:

— коэффициент попадания – отношение числа обращений к страницам, которые уже загружены в буфер, к общему числу обращений;

— коэффициент совпадения – отношение количества совпавших с оптимальным алгоритмов кандидатов на замещение с общим числом запросов поиска страниц для вытеснения.

Размер скрытых слоев модели подбирался экспериментально. Графики зависимости коэффициента совпадения в зависимости от эпохи обучения для различных размеров скрытых слоев на обучающей и тестовой выборках представлены на рисунках 1.5 и 1.6 соответственно.

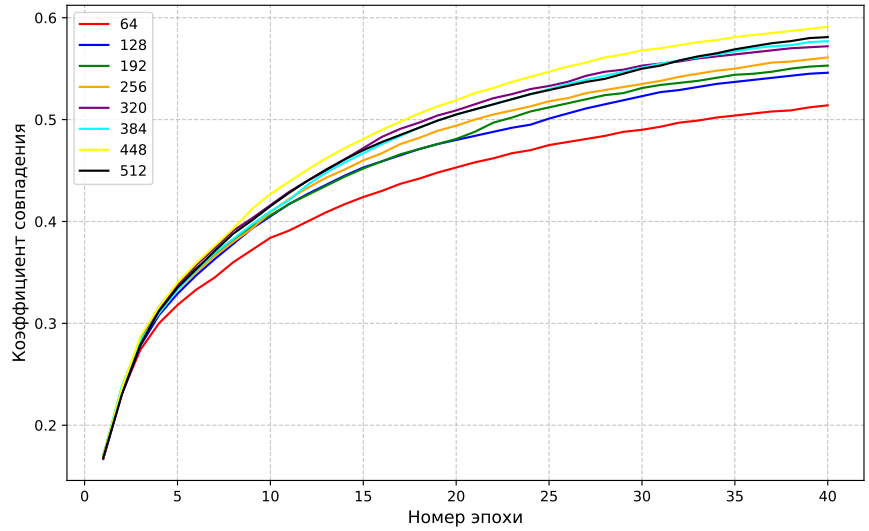

Рисунок 1.5 – Точность модели для различных размеров скрытых слое на тренировочной выборке

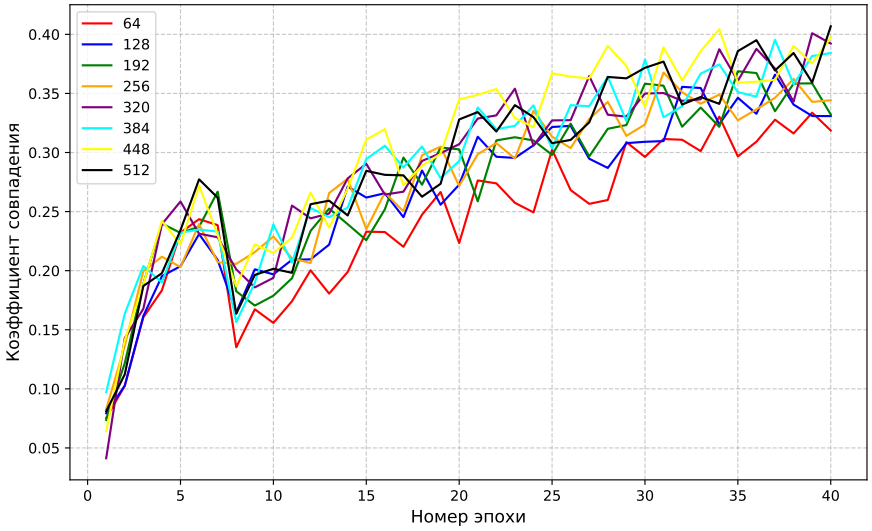

Рисунок 1.6 – Точность модели для различных размеров скрытых слое на тестовой выборке

Исходя из полученных результатов, настраиваемые параметры модели: d_z, d_b, d_h и d_v были выбраны равными 448, а d_f – 32.

1.3 Сравнение с аналогами

Сравнение коэффициентов попадания для разработанного метода и существующих аналогов приведено на рисунке 1.7.

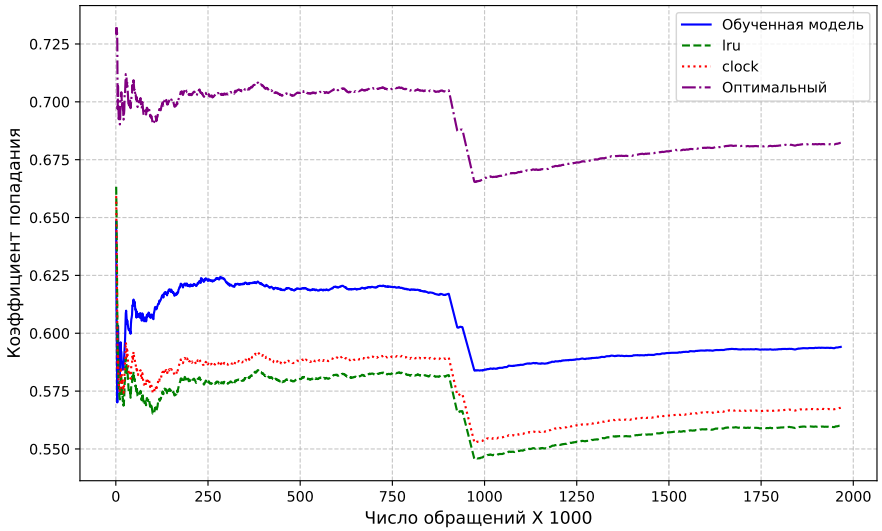

Рисунок 1.7 – Коэффициент попадания в зависимости от числа обращений для различных методов

Из графиков видно, что коэффициент попадания для разработанного метода в среднем на 0.03 выше чем для алгоритма clock, который в настоящее время используется в PostgreSQL. Также коэффициент попадания для разработанного метода на 0.08 ниже, чем у оптимального алгоритма. Таким образом, разработанный метод лучше существующий аналогов, но все еще имеет возможность для улучшения.

Коэффициент попадания для существующих аналогов ниже одного процента.

ЗАКЛЮЧЕНИЕ

В данной научно-исследовательской работе были:

— описан и спроектировать метод замещения страниц с использованием нейронных сетей;

— проведено сравнение разработанного метода с существующими аналогами по коэффициентам совпадения и попадания.

В рамках работы были выполнены все поставленные задачи. Цель работы была достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Peiquan Y.* Learned buffer replacement for database systems // Proceedings of the 2022 5th International Conference on Data Storage and Data Engineering. — 2022. — С. 18—25.
2. *Shaik B.* PostgreSQL Configuration: Best Practices for Performance and Security. — Apress, 2020.
3. *Zhang J.* A review of recurrent neural networks: LSTM cells and network architectures // Neural computation. — 2019. — Т. 31, № 7. — С. 1235—1270.