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 и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ:

«Метод построения поисковых индексов в реляционной базе данных на основе глубоких нейронных сетей»

Студент: ИУ7-83Б _____ М. Д. Маслова
(группа) (подпись, дата) (И. О. Фамилия)

Руководитель: _____ А. А. Оленев
(подпись, дата) (И. О. Фамилия)

Нормоконтролер: _____ _____
(подпись, дата) (И. О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

1 Конструкторская часть

1.1 Требования и ограничения метода

Метод построения поисковых индексов в реляционной базе данных на основе глубоких нейронных сетей (далее – метод построения индексов) должен:

1. получать из таблицы реляционной базы данных набор ключей и набор соответствующих указателей на записи в индексируемой таблице реляционной базы данных или иных значений, выполняющих роль указателей;
2. выполнять предварительную обработку полученных наборов, такую, как их совместную сортировку по значениям ключей, получение позиций ключей в отсортированном виде и нормализацию ключей и позиций;
3. обучать модель нейронной сети на подготовленных набора ключей и позиций;
4. сохранять параметры обученной модели для каждой таблицы с целью возможности выполнять запросы поиска без переобучения;
5. обеспечивать поиск записи (диапазона записей) таблицы по ключу (диапазону ключей) с использованием обученной модели;
6. обеспечивать корректность операции поиска после вставки/удаления новых записей путем переобучения модели;

На разрабатываемый метод накладываются следующие ограничения:

— в качестве ключей на вход принимаются целые числа для исключения решения дополнительной задачи преобразования входных данных;
— ключи во входном наборе уникальны.

1.2 Особенности метода построения индекса

1.2.1 Общее описание метода построения индекса

Основные этапы метода построения индекса приведены на функциональной декомпозиции метода на рисунке 1.1.

На вход методу подается набор уникальных целочисленных ключей, которые перед обучением модели глубокой нейронной сети проходят предварительную обработку по определенным правилам, описанным далее. Отдельным

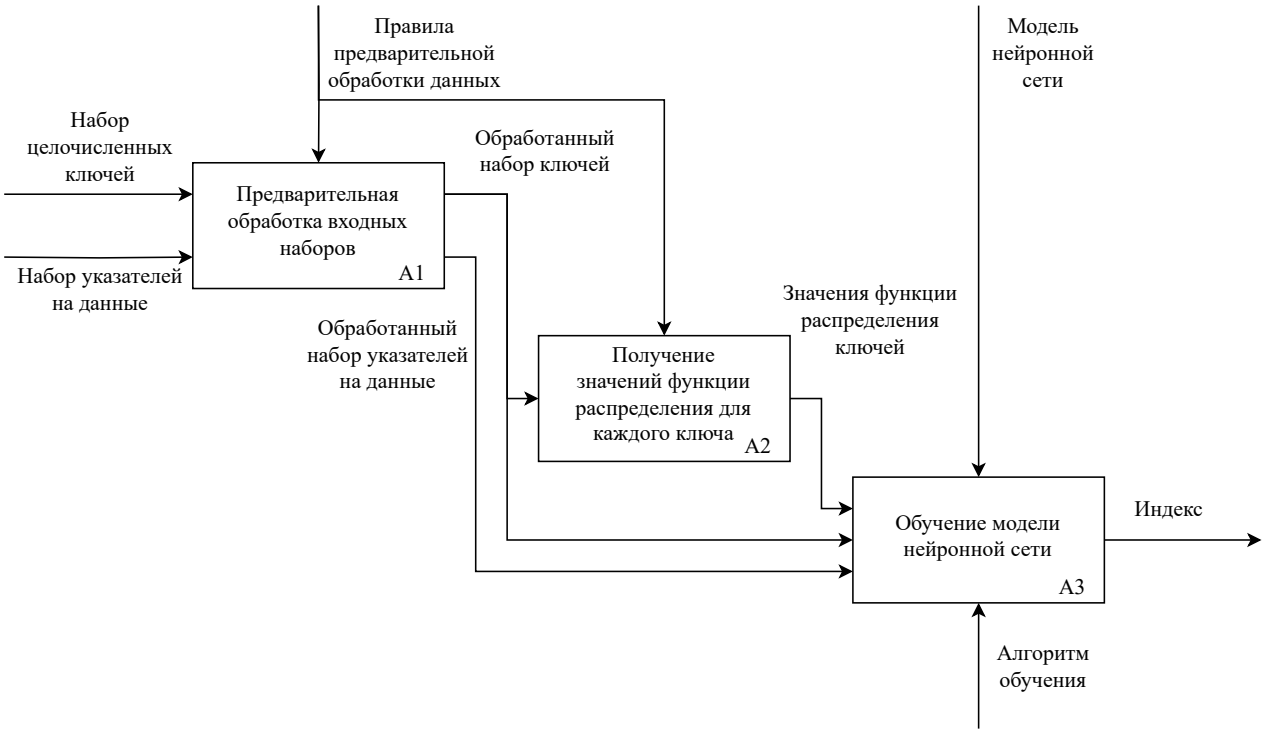

Рисунок 1.1 – Функциональная схема метода построения индекса

этапом выделено получение значений функций распределения для каждого ключа, относящееся к предварительной обработке, но представляющее собой ее ключевой момент. Полученные после первых двух этапов обработанные ключи и соответствующие значения функций используются для обучение модели глубокой нейронной сети в качестве признаков и меток соответственно.

Ключевым моментом метода является представление в отсортированном (по ключам) виде наборов ключей и набора соответвующих указателей на данные. Именно отсортированный вид позволяет использовать закономерность распределения ключей по позициям для обучения модели, предсказывать позиции ключей и уточнять их.

Результатом работы метода является структура данных, предствляющая собой индекс на основе глубокой нейронной сети и имеющая следующие поля:

— отсортированный массив ключей, поданных на вход;
— отсортированный по значениям ключей массив указателей на данные, соответствующие ключам;
— модель обученной глубокой нейронной сети, с помощью которой будет предсказываться положение ключа в отсортированном массиве;
— средняя и максимальная абсолютные ошибки предсказания позиции

ключа, для ее уточнения и возврата верного указателя на данные.

Краткое описания индекса, являющегося результатом работы метода, как структуры данных представлено на рисунке 1.2.

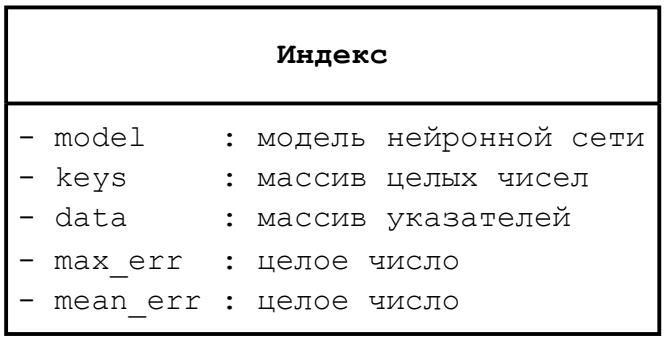

Рисунок 1.2 – Индекс как структура данных

Подробное описание каждого этапа приведено в следующих пунктах данного подраздела.

1.2.2 Предварительная обработка данных

Разрабатываемый метод построения индекса предполагает предварительную обработку набора целочисленных ключей, схема алгоритма которой представлена на рисунке 1.3.

На вход подаются согласованные массивы ключей и указателей на данные, то есть считается, что ключ, стоящем на первой позиции в массиве ключей, идентифицирует данные по указателю, стоящем на первой позиции в массиве указателей; ключ, стоящей на второй, — указатель, стоящий на второй, и так далее. С учетом этого происходит согласованная сортировка двух массивов по значениям ключей.

Далее для последующего обучения модели глубокой нейронной сети производится нормализация ключей, выступающих в качестве входных данных сети, в дапазон $[0, 1]$, для чего используется метод минимакс-нормализации, при котором нормализованное значение вычисляется по формуле:

$$x_{\text{норм}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \tag{1.1}$$

где $x_{\text{норм}}$ — нормализованное значение ключа;

x — натуральное значение ключа;

$x_{\min}$, $x_{\max}$ — минимальное и максимальное возможное значение ключа в

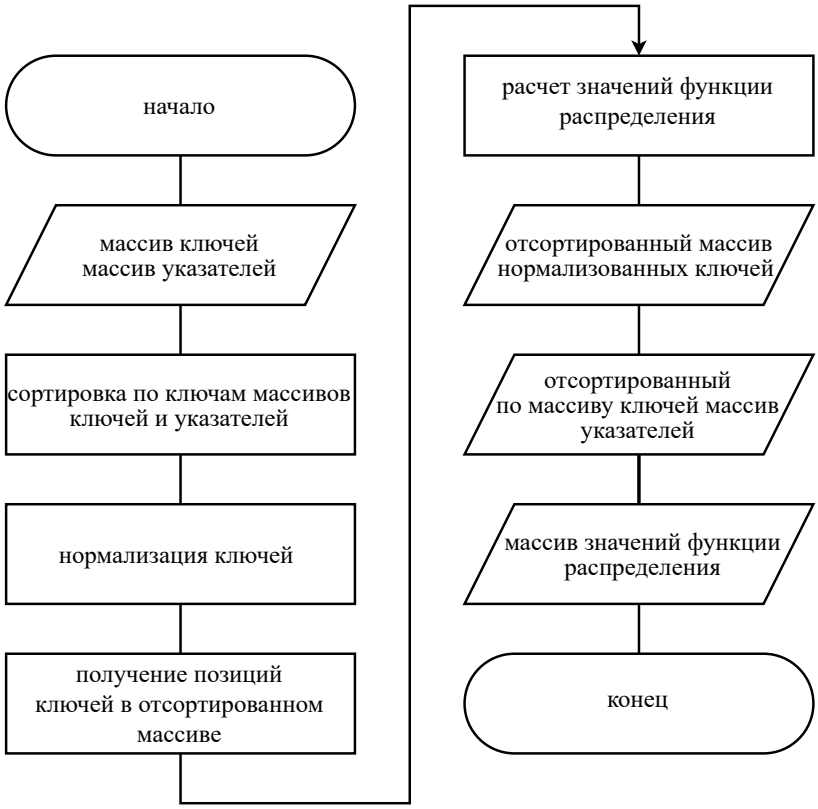

Рисунок 1.3 – Схема алгоритма предварительной обработки данных

наборе соответственно.

Далее полученный набор ключей размечается путем вычисления для каждого ключа K значения функции распределения F по его позиции P в отсортированном массиве и количества индексируемых ключей N с помощью формулы:

$$F(K) = \frac{P}{N} \tag{1.2}$$

На этом предварительная обработка завершается и полученные отсортированные массивы ключей и указателей, а также соответствующие значения функции распределения передаются в качестве входных данных на этап обучения модели глубокой нейронной сети.

Полное описание алгоритма предварительной обработки представлено на листинге 1.1.

Листинг 1.1 – Предварительная обработка данных

Вход:
$keys$: массив целочисленных ключей;
$data$: массив указателей на данные, соответствующие ключам;
N : длина массивов.
Выход:
$keys$: отсортированный массив нормализованных ключей;
$data$: отсортированный по массиву ключей массив указателей;
cdf : массив значений функции распределения.

```
1 begin
2     сортировать keys и data по keys;
      ▷ здесь и далее под операцией к вектору (массиву) и числу понимается
      ▷ применение данной операции с данным числом к каждому элементу вектора
3     keys ← (keys − keys[0]) / (keys[N−1] − keys[0]);
4     positions ← [0, 1, ..., N − 1];
5     cdf ← positions / (N − 1);
6     return keys, data, cdf;
7 end
```

1.2.3 Разработка архитектуры глубокой нейронной сети

Полученные на этапе предварительной обработки массивы ключей и соответствующих им значений функций распределения используются для обучения глубокой нейронной сети. Массив данных, хранимый в индексе, используется для возврата нужных данных при поиске, но не для обучения.

Задача построения индекса на основе нейронной сети сводится к задаче аппроксимации функции распределения, для решения которой подходят полносвязные нейронные сети.

За основу архитектуры глубокой нейронной сети принята архитектура из исследования [1], которая представлена на рисунке 1.4. Это полносвязная нейронная сеть с двумя скрытыми слоями по 32 нейрона. В качестве фунцкции активации в каждом нейроне скрытых слоев используется линейный выпрямитель или ReLU (*Rectified Linear Unit*), значение которой вычисляется по формуле 1.3.

$$f(x) = \max(0, x). \tag{1.3}$$

Активационная функция выходного слоя является линейной.

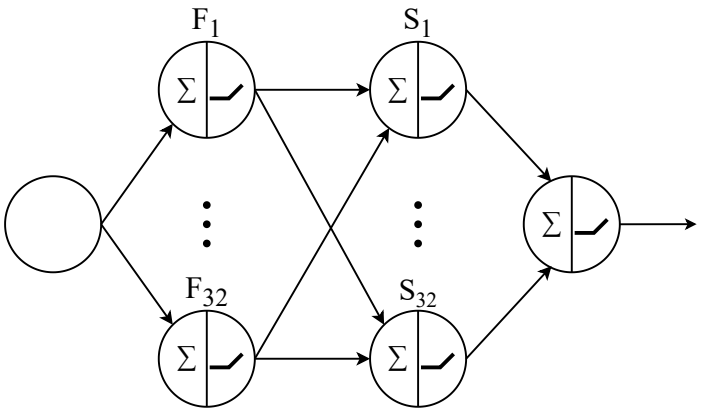

Рисунок 1.4 – Полносвязная нейронная сеть с двумя скрытыми слоями

Для исследования возможности увеличения точности предсказания, и как следствие уменьшение времени поиска, в качестве модели глубокой нейронной сети, представляющей основу индекса, используется полносвязная нейронная сеть с тремя слоями, представленная на рисунке 1.5. Число нейронов в слоях и активационные функции приняты такими же, как в случае глубокой нейронной сети с двумя скрытыми слоями.

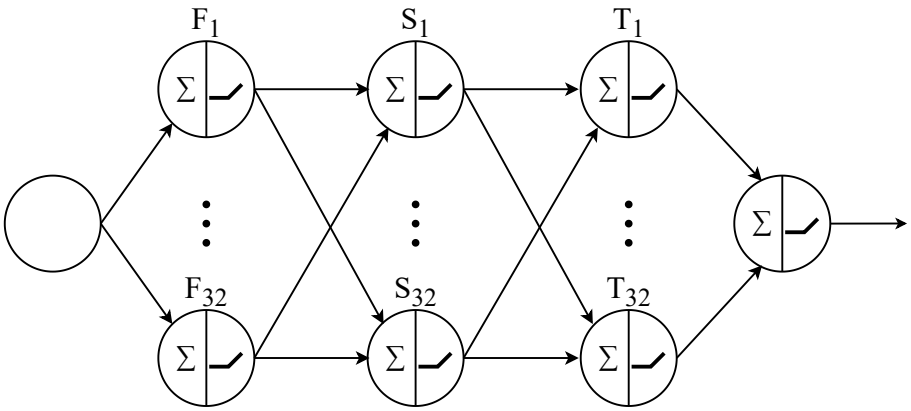

Рисунок 1.5 – Полносвязная нейронная сеть с тремя скрытыми слоями

Обучение обоих моделей глубокой нейронной сети начинается с инициализации весов случайно сгенерированными значениями по распределению $U(-\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}})$. Собственно обучение проводится методом стохастического градиентного спуска с оптимизацией в качестве функции потерь среднеквадратической ошибки (*MSE — mean squared error*). Описание алгоритма обучения приведено на листинге 1.2.

Листинг 1.2 – Алгоритм обучения глубокой нейронной сети на основе градиентного спуска

Вход:

$keys$: массив нормализованных ключей;

cdf : массив значений функции распределения для каждого ключа;

N : длина массивов;

$epochs$: количество эпох;

$alpha$: скорость обучения.

Выход:

$model$: обученная модель.

begin

  $model \leftarrow$ структура модели;

  ▷ вектор смещений включей в матрицу весов

  $model.weights \leftarrow U(-\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}})$; ▷ случайные значения из распределения

  for $epoch \leftarrow 1$ **to** $epochs$ **do**

    согласованно перемешать $keys$ и cdf;

    for $i \leftarrow 0$ **to** $N - 1$ **do**

      $\hat{y} = model.predict(key[0])$; ▷ прямой проход

      $mse = (\hat{y} - cdf[i])^2$;

      $gradients \leftarrow backward\_pass(model, mse)$; ▷ обратный проход

      $model.weights \leftarrow model.weights - alpha \cdot gradients$

    end

  end

end

Так как в случае поискового индекса глубокая нейронная сеть будет предсказывать положения только тех ключей, на которых она обучалась, явления переобучения нейронной сети является положительным, поэтому в качестве одного батча выступает одна пара (ключ; значение функции распределения), что также отражено на листинге выше.

1.3 Разработка алгоритмов поиска и вставки

Основной операцией, выполняемой с помощью индекса, является поиск, функциональная схема выполнения которого представлена на рисунках 1.6-1.7.

Разрабатываемый алгоритм поиска должен поддерживать операцию поиска по единичиному ключу и по диапазону ключей, то есть принимать диапазоны, представленные в виде операций сравнения `=, >, <, >=, <=`, а `a <= key <= b, ...`, где key — ключ, a, b — границы диапазона.

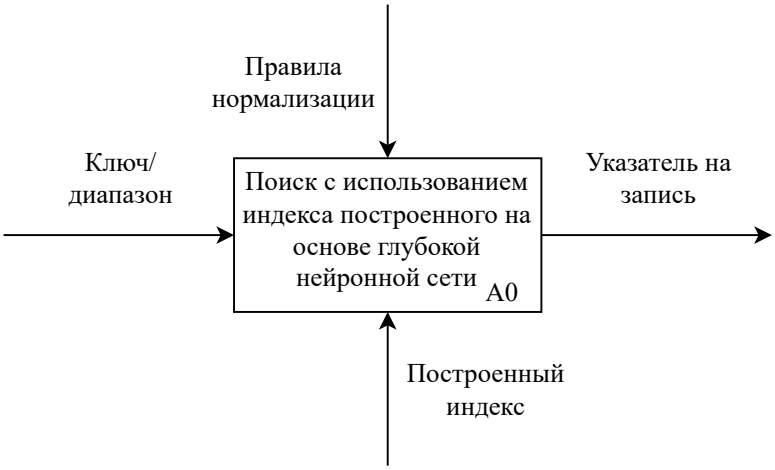

Рисунок 1.6 – Функциональная схема нулевого уровня поиска

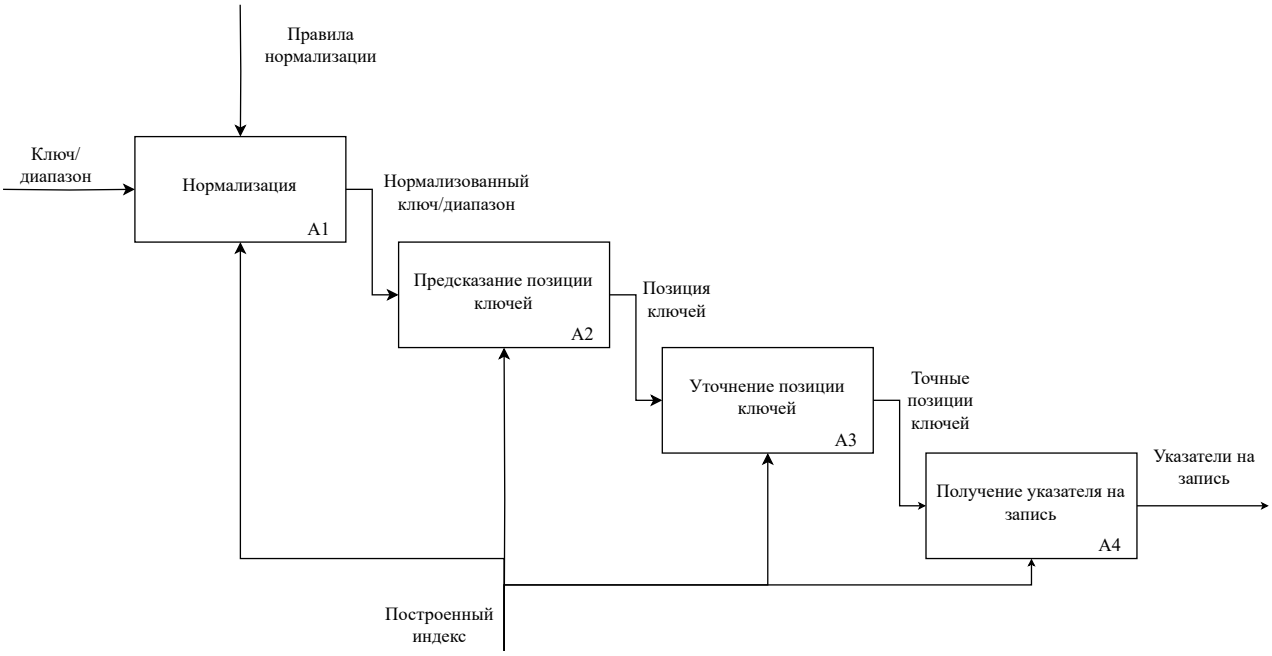

Рисунок 1.7 – Функциональная схема первого уровня поиска

В случае ограничения с одной стороны производится поиск позиции одного ключа, со значением соответствующим значению границы, с двух сторон — двух ключей. По найденным позициям и позициям первого и последнего ключа формируется диапазон позиций в соответствии с заданным ограничением. По найденному диапазону позиций выполняется получение указателей на нужные записи.

Получаемые по результатам поиска с помощью модели глубокой нейронной сети позиции требуют уточнения, происходящего за счет получаемого в результате обучения модели среднего и максимального отклонения от истинно-

го расположения. Так ключи в индексе отсортированы применяется алгоритм бинарного поиска.

Подробный алгоритм поиска представлен на листинге 1.3.

Листинг 1.3 – Алгоритм поиска с использованием индекса на основе глубокой нейронной сети

Вход:

$keys$: кортеж двух значений ключей, задающие ограничения поиска;
$constraints$: ограничение границ (включая (0)/не включая (1));
$index$: построенные индекс.

Выход:

$data$: массив указателей на записи.

```
function clarify(key, limit, is_lower, constraint, index)
    if is_lower и limit == None then
        return 0;
    else if limit == None then
        return index.keys.size − 1;
    end
    bin_lower ← max(limit − index.error, 0);
    bin_upper ← min(limit + index.error, index.keys.size − 1);
    ▷ бинарный поиск, но возвращаются итоговые нижняя и верхняя границы
    lower, upper ← binary_search(key, bin_lower, bin_upper);
    if lower == upper then
        return lower − (−1)^(is_lower) · constraint;
    else if is_lower then
        return upper;
    else
        return lower;
    end
end
begin
    ▷ при передаче None возвращается None
    limits ← index.normalize(keys);
    positions_limits ← index.predict(limits);
    ▷ lower, upper — нижняя и верхняя границы позиций ключей в диапазоне
    lower ← clarify(key[0], positions_limits[0], True, constraints[0], index);
    upper ← clarify(key[1], positions_limits[1], False, constraints[1], index);
    return index.data[lower : upper]
end
```

Вставка осуществляется путем комбинирования алгоритма поиска и построения: сначала происходит поиск позиции, куда должна произойти вставка, далее новые ключ и указатель помещаются в данную позицию отсортированных массивов и в конце происходит дообучение модели на новом наборе данных, то есть обучение модели происходит с уже имеющихся весов.

1.4 Разработка архитектуры программного обеспечения

Метод построения индекса разработан

1.5 Данные для обучения и тестирования индекса

Так как в основе индекса на основе глубоких нейронных сетей лежит аппроксимация функции распределения ключей, работу разработанного метода необходимо протестировать на различных законах, перечисленных далее.

— Равномерный закон $R[a, b]$, фукнция распределения которого описывается формулой 1.4.

$$F(x) = \begin{cases} 0, & \text{если } x < a \\ \frac{x-a}{b-a}, & \text{если } a \leq x \leq b \\ 1, & \text{если } x > b. \end{cases} \tag{1.4}$$

Нормализованные ключей лежат в диапазоне $[0, 1]$, значения функции распределения за пределами этого диапазона не представляют интереса для построения индекса, поэтому можно считать, что функция имеет вид, представленный формулой 1.5.

$$F(x) = x, \ x \in [0, 1]. \tag{1.5}$$

— Нормальный закон $N(\mu, \sigma^2)$, функция распределения которого описывается формулой 1.6.

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(t-\mu)^2}{2\sigma^2}\right) dt \tag{1.6}$$

Для тестирования метода построения по заданным законам генерируются значения ключей, по совокупности которых формируется эмпирическая функция распределения, как это было описано выше.

Также для проверки работы метода требуется оценить его работоспособность на реальных данных, в качестве которых выбраны уникальные идентификаторы элементов из открытого набора географических данных *OpenStreetMap*, или *OSM* [**osm**], функция распределения которых имеет более сложный вид, чем основные законы распределения.

Графики функций распределения каждого из набора входных ключей, представлены на рисунке 1.8.

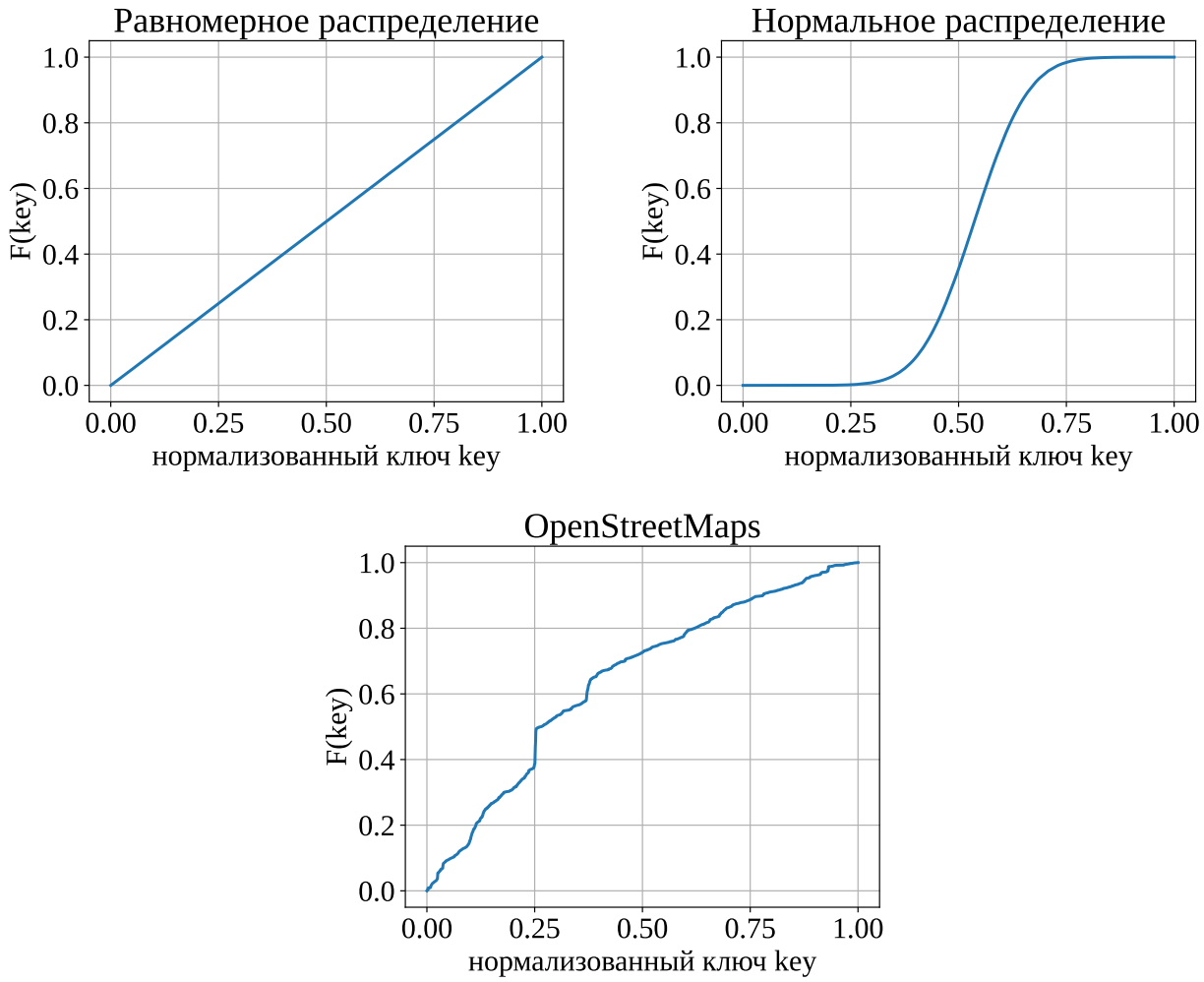

Рисунок 1.8 – Графики функций распределения ключей из наборов данных для обучения и тестирования

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Kraska T.* The Case for Learned Index Structures / T. Kraska, A. Beutel, E. H. Chi [и др.] // Proceedings of the 2018 International Conference on Management of Data. — 2018. — С. 489—504.