Нейроконнекторы (ML)

Цель

Создание инструмента для формирования скрипта загрузки без необходимости использовать SQL в секции подключения.

Концепт ML

Языковая модель, способная обрабатывать запросы от пользователя и давать ответ + преобразовывать их в код на языке ClickHouse. Должна уметь распознавать и выполнять следующие типы запросов:

- Работа с типами данных: модель должна уметь определять и подставлять подходящий тип данных столбца. Пример: "Выбери наиболее подходящий тип данных для столбца Cost" или "Ты неправильно определил тип данных столбца Cost. Смени его на вещественный".
- Обработка данных: модель должна уметь писать код для трансформации таблицы, изменения имеющихся столбцов, создания новых на основании запроса (в т.ч. с использованием агрегирующих функций). Пример: "Добавь столбец Profit как результат выражения (Price - Cost) * Count - 5000" или "Транспонируй текущую таблицу".
- Фильтрация и ограничение: модель должна уметь оставлять в наборе данных только те столбцы, строки и значения, которые удовлетворяют запросу от пользователя. Пример: "Удали все значения больше 20000" или "Оставь только первые 5000 строк".

Задание на разработку ML

При обработке запросов модель должна уметь определять столбцы и таблицы, даже если пользователь совершает ошибки в написании их названий либо пишет их на другом языке.

> При обучении модели необходимо учитывать, что организация работы с данными происходит на ClickHouse, поэтому особенно важно обрабатывать особенности этого языка запросов, такие как использование сортировки для индексации и специфичность наименований типов данных.

> **ВАЖНО!**Нейросеть не работает с готовым SQL-кодом или таблицей (не создаётся секции ALTER и запросов на изменение). Её задача - написать код для формирования модели данных как результата выборки из БД ClickHouse по шаблону, подставляя в него фрагменты, позволяющие решить поставленную пользователем задачу.

Вход модели:

Таблицы со всеми столбцами и значениями в них + промпт от пользователя. Задачи от пользователя отправляются последовательно через чат, в то время как исходные таблицы неизменны.

Модель должна определить решаемую задачу в зависимости от запроса пользователя. Если пользователь неудовлетворен результатом работы модели, то нейросеть должна пересмотреть решаемую задачу.

Работа с типами данных

Типы данных

Модель должна уметь:

- Определять текущий тип данных указанного пользователем столбца. Эта информация передаётся вместе с метаданными, поэтому для решения этой задачи необходимо лишь передать в ответе имеющееся значение.
- Определять наиболее подходящий тип данных. Причём самый простой из подходящих (не выбирать словари или LowCardinality). Для решения этой задачи модель должна обрабатывать имеющиеся в столбце данные и на их основании предлагать нужный тип.
- Менять тип данных. Менять можно только в том случае, если значения столбца можно хранить в конечном типе данных. Модель должна уметь переводить данные как в конкретный тип ("Измени тип данных столбца Доход на вещественный"), так и в подходящий ("Измени тип данных столбца Расход на наиболее подходящий"). Измененный тип данных попадает в итоговый скрипт загрузки.

Обработка данных

Модель должна уметь изменять имеющиеся столбцы, создавать новые, сортировать по указанным столбцам.

- Изменение столбцов – модель записывает выражение на языке ClickHouse для текстового запроса от пользователя на редактирование существующего столбца. В качестве выражения может выступать комбинация арифметических и агрегирующих операций над столбцами, числами и текстом. Результат – вместо передачи названия существующего столбца в скрипт загрузки передаётся это выражение.

Пример: "Сделай так, чтобы в столбце Сумма результат выводился в миллионах, а не в 1000".
Результат (в Select): toFloat32("Sum")/1000.

- Создание столбцов – модель записывает выражение на языке ClickHouse для текстового запроса от пользователя на формирование нового столбца. В качестве выражения может выступать комбинация арифметических и агрегирующих операций над столбцами, числами и текстом. Результат – после списка с названиями существующих столбцов добавляется это выражение.
  Пример: "Создай столбец Дельта как разность между доходами и расходами".
  Результат (в Create): "Delta" Int32.
  Результат (в Select): toInt32("Income")-toInt32("Cost").
- Переименование столбцов – в секции CREATE вместо существующего имени столбца модель добавляет введенное пользователем.
- Сортировка столбцов – модель добавляет секцию ORDER BY согласно запросу от пользователя.

Фильтрация и ограничение

Модель должна уметь конструировать запросы для секций WHERE, LIMIT и OFFSET:

- Ограничение по условию – модель формирует секцию WHERE: интерпретирует текст от пользователя как выражение, содержащее оператор сравнения, принадлежности или шаблон для столбцов, чисел и текста.
  Пример: "Оставь только те значения в столбце Доход, которые больше 500 000"
  Результат (в Where): toInt32("Income")>500000
- Ограничение по числу значений:
  - "Оставь только первые N значений" – модель добавляет секцию LIMIT N.
  - "Оставь последние N значений" – модель добавляет секцию OFFSET N.
  - "Оставь значения, начиная с N и заканчивая M" – модель добавляет секции LIMIT N и OFFSET M.

> Если пользователь пытается ввести запрос, не имеющий отношения к формированию скрипта загрузки или изучению данных, то модель должна выдать ответ вида: "Я могу помочь только с формированием скрипта загрузки или изучением данных.
> Пожалуйста, уточните Ваш запрос или составьте новый таким образом, чтобы он имел отношение к представленным данным!"

Шаблон кода для скрипта загрузки (выход модели)

```
Table "Название таблицы"


Create @@@ -- шаблонная секция, подставляются из модели только названия столбцов и их типы данных
CREATE TABLE IF NOT EXISTS "Название таблицы"(
  "Название столбца 1" type, -- тип данных, если допускается null, то Nullable(type) // ЗАДАЧА НА РАБОТУ С
```

```
ТИПАМИ ДАННЫХ
  "Название столбца 2" type, -- название столбца может быть изменено моделью
  ...
  "Название столбца n" type
) ENGINE = MergeTree ()

ORDER BY
  tuple ()
@@@

Delete @@@ -- ещё одна шаблонная секция
  ALTER TABLE "Название таблицы" DELETE WHERE 1=1
@@@

Source "Название подключения"

Read @@@ -- основная секция для обработки моделью
SELECT
  "Название столбца 1", -- перенос столбцов из из источника
  "Название столбца 2",
  ...
  "Название столбца k",
  Выражение для изменения столбца 1, -- ЗАДАЧА НА ИЗМЕНЕНИЕ СТОЛБЦОВ
  Выражение для изменения столбца 2,
  ...
  Выражение для изменения столбца m,
  Выражение для нового столбца 1 as "Название нового столбца 1", -- ЗАДАЧА НА СОЗДАНИЕ НОВЫХ
СТОЛБЦОВ
  Выражение для нового столбца 2 as "Название нового столбца 2",
  ...
  Выражение для нового столбца n as "Название нового столбца n"
FROM
  "Название таблицы"
WHERE
  toType("Название столбца k") Условие 1 -- для каждого столбца, используемого в WHERE, у которого
изменился тип данных, необходимо указывать тип из Create (например, для типа Int32 будет toInt32 и
т.д.) // ЗАДАЧА НА ОГРАНИЧЕНИЕ
  ...
ORDER BY
  "Название столбца k" тип сортировки -- ЗАДАЧА НА СОРТИРОВКУ
```

```
  ...
LIMIT a
OFFSET b -- ЗАДАЧА НА ОГРАНИЧЕНИЕ

Optimize @@@ -- шаблонная секция на удаление дубликатов
OPTIMIZE TABLE "Название таблицы" DEDUPLICATE
@@@
```

Пример выхода модели для скрипта загрузки

```
Table "Costs"

Create @@@
CREATE TABLE IF NOT EXISTS "Costs" (
  "Date" Date,
  "Sum" Int32,
  "Product_name" Nullable (String),
  "Category" Nullable (String),
  "file_name" Nullable (String),
  "new_object" Int32
) ENGINE = MergeTree ()

ORDER BY
  tuple ()
@@@

Delete @@@
ALTER TABLE "Costs" DELETE WHERE 1=1
@@@

Source "01"

Read @@@
SELECT
  "Date",
  "Sum",
  "Product_name",
  "Category",
  "file_name",
  toInt32("Sum")+666
```

```
FROM
  "Costs"
WHERE
  toInt32("Sum")>5000
ORDER BY
  "Product_name" desc
LIMIT 10
OFFSET 5
@@@

Optimize @@@
OPTIMIZE TABLE "Costs" DEDUPLICATE
@@@
```

Выход модели в секции подключения:

- Ответ языковой модели в чате (либо "Выполнено/Не выполнено", если поступил запрос на обработку данных).
- Код для изменения выборки из 10 строк для предпросмотра (СОСТЫКОВАТЬСЯ С БЭКОМ!)

"Память" модели

Для каждого подключения нейросеть хранит собственную "Память", включающую в себя по меньшей мере историю переписки с пользователем и код для изменения выборки из 10 строк для предпросмотра. Кроме того, существует один из трёх вариантов (обсуждаемых с заказчиком), в зависимости от которого в памяти модели может храниться:

☐ Ничего – модель обращается напрямую к базе ClickHouse. Этот вариант возможен в том случае, если в секции подключения останется только редактор скрипта, а инструмент Data Discovery будет выделен в отдельную задачу.

☐ Кэшированную таблицу объёмом до 1 млн строк

☐ Всю таблицу целиком

Поиск и анализ в секции подключения (дополнительно)

Поиск в данных ("Что и где находится?")

Не возвращает SQL-кода ни для таблицы-примера, ни для добавления в скрипт загрузки.

Модель должна по запросу от пользователя определить зону поиска в данных (локализовать до столбца, набора столбцов, интервала строк и т.д.) после чего выполнить сам запрос на поиск и написать ответ в чате. Решаемые в рамках данной задачи запросы *могут* включать в себя:

- Поиск конкретного значения (числового или текстового).
  Запрос вида: "Есть ли среди Компаний AVA?".
  Возвращает ответ вида: "{Количество} {Значение} есть/отсутствует в {Зона поиска}" – "AVA есть среди компаний".
- Вывод всех уникальных значений в столбце. При первом запросе модель должна выдавать 10 самых часто встречающихся результатов с припиской "и ещё n", где n: (число уникальных значений)-10. При повторном запросе от пользователя выдаёт все значения через запятую (даже если их 100500).
- Поиск/подсчёт значений из диапазона/в диапазоне (набор строк, элемент строки, диапазон числовых значений).
  Запрос вида: "У скольких работников из таблицы зарплата больше 100 000?".
  Возвращает ответ вида: "{Количество} {Значение} есть/отсутствует в {Зона поиска}" – "У 28 работников зарплата больше 100 000".
- Поиск агрегированного значения из числа следующих: минимум, максимум, среднее, сумма, разность, произведение, частное, статистические функции (мода, медиана, дисперсия, ско).
  Возвращает предупреждение о больших затратах времени на выполнение операции вида "Этот запрос может выполняться длительное время. Напишите "да", если вы уверены, что хотите получить результат. Иначе можете продолжить чат".
  Запрос вида: "Найди проект с наибольшими затратами".
  Если пользователь напишет "Да", то модель должна выдать ответ вида: "{Запрашиваемое агрегированное значение} в {Зона поиска} – {Результат работы модели}" – "Проект с наибольшими затратами – AVA". Иначе продолжает отвечать на другие запросы.
- Другие запросы на поиск в данных.

Вид ответа может меняться в зависимости от сути вопроса пользователя. Модель должна определять последовательность логики запроса (когда пользователя интересует показатель, а когда - его аналитический разрез). Например, ответом на вопрос: "В каком городе наибольшее число жителей?" должно быть название города, а результатом поиска по запросу: "Найди наибольшее число жителей среди представленных городов" – число жителей.

Анализ использования ("Зачем и где применяется?")

Не возвращает SQL-кода ни для таблицы-примера, ни для добавления в скрипт загрузки.

Модель должна по запросу от пользователя определить и описать в ответе в чате контекст применения набора данных: решаемую задачу, экономическое применение, назначение

столбцов.

> Для решения этой задачи необходимо анализировать не только метаданные (названия таблиц и столбцов), но и содержимое этих таблиц.

- Решаемая задача – ответ на вопросы вида: "Что я могу сделать с этими данными?". Модель должна определять спектр задач по типу: "Вы можете использовать эти данные для определения финансовых показателей компании за 2024 год".
- Экономический контекст – ответ на вопросы вида: "Где используются эти данные?". Модель должна определять сферу применения по типу: "Предположительно, эти данные используются в сфере продаж".
- Назначение столбцов – детализация решаемой задачи в области применения отдельных столбцов. Ответ на вопросы вида: "Зачем мне нужен этот столбец?". Модель должна определять назначение столбцов по типу: "Это вспомогательный столбец, который нужен для организации корректной сортировки".
- Другие запросы на выявление области применения данных.

Revision #6
Created 27 November 2024 11:47:17 by Артём
Updated 21 January 2025 05:58:00 by Артём