Санкт-Петербургский государственный университет

Щукин Илья Вячеславович

Выпускная квалификационная работа

Определение семантического типа колонки в табличных данных с помощью методов машинного обучения

Уровень образования: бакалавриат

Направление *02.03.03 «Математическое обеспечение и администрирование информационных систем»*

Основная образовательная программа *СВ.5006.2019 «Математическое обеспечение и администрирование информационных систем»*

Научный руководитель:
к. ф.-м. н., старший научный сотрудник кафедры информационно-аналитических систем
Е. Г. Михайлова

Консультант:
ассистент кафедры информационно-аналитических систем Г. А. Чернышев

Рецензент:
инженер-стажёр ООО “Юнидата” А. Н. Смирнова

Санкт-Петербург
2023

Saint Petersburg State University

Shchukin Ilia

Bachelor's Thesis

Semantic type detection for columnar data using machine learning

Education level: bachelor

Speciality *02.03.03 "Software and Administration of Information Systems"*

Programme *CB.5006.2019 "Software and Administration of Information Systems"*

Scientific supervisor:
senior researcher, C.Sc. E.G. Mikhailova

Consultant:
assistant G.A. Chernishev

Reviewer:
intern engineer "Unidata" A.N. Smirnova

Saint Petersburg
2023

Оглавление

Введение **4**

1. Постановка задачи **5**

2. Обзор **6**

2.1. Существующие решения 6

2.2. Индустриальные продукты 8

3. Разработка архитектуры **11**

3.1. Архитектура модели Doduo 11

3.2. Выбор языковой модели 13

4. Набор данных **14**

4.1. Процесс обработки данных 14

4.2. Фильтрация данных 16

4.3. Выбор меток для сущностей 16

5. Обучение модели **18**

5.1. Получение токенов из табличных данных для модели . . 18

5.2. Расширение вокабуляра токенайзера 19

5.3. Аугментация данных 20

5.4. Процесс обучения 20

5.5. Результаты обучения 21

Заключение **23**

Список литературы **24**

Введение

Определение семантических типов колонок в табличных данных является важной задачей в области профилирования. Под семантическим типом понимают связь между содержимым колонки и какой-то сущностью из реального мира, например: “Имя”, “Возраст”, “Страна”. При этом определение семантических типов значительно сложнее определения атомарных таких как: int, string, date, так как одному атомарному типу может соответствовать несколько семантических. Например семантические типы “Страна”, “Столица” и “Область” соответствуют атомарному string.

Для автоматизации поиска семантических типов исследователями было предложено множество подходов. Классические решения данной задачи используют регулярные выражения, словари и статистики. При этом они обладают малой точностью и сильно ограничены количеством распознаваемых типов [7]. Применение машинного обучения позволило значительно повысить качество распознавания и увеличить число распознаваемых сущностей.

Успешное решение данной задачи открывает возможности для решениях смежных задач, создавая для них основу. Полученную модель возможно применить для извлечения эмбеддингов, векторов представляющих семантику таблицы. Получаемые эмбеддинги далее можно использовать для решения задач интерпретации, поиска, заполнения пропущенных значений, расширения таблиц новыми столбцами и строками [11, 12].

Существующие на данный момент решения не подходят для работы с данными, содержащими русский язык, поскольку используют для обучения данные, которые в основном содержат только английский язык. Также отсутствует набор данных, который позволил бы обучить модель, способную решать поставленную задачу. В данной работе описан процесс создания соответствующего набора данных и разработки модели машинного обучения для определения семантических типов колонок.

1 Постановка задачи

Целью данной ВКР является разработка модели машинного обучения, способной определять семантические типы колонок в русскоязычных табличных данных. Для достижение данной цели были сформулированы следующие задачи.

1. Провести обзор существующих решений для определения семантических типов колонок.
2. Разработать архитектуру модели машинного обучения.
3. Подготовить набор данных на основе данных из корпуса RWT.
4. Обучить модель на подготовленном наборе данных и провести экспериментальное исследование.

2 Обзор

2.1 Существующие решения

2.1.1 Sherlock

Первым решением данной задачи, которое использовало машинное обучение является модель Sherlock [7]. Для предсказания типа из колонки извлекаются 1588 различных характеристик (распределения символов, глобальные статистики и другие). После этого для уменьшения размерности получаемого вектора, эти значения разделяются на группы, которые обрабатываются отдельными подсетями, Feature-specific Subnetwork на Рис.1. После обработки выходы этих подсетей конкатенируются и подаются на вход основной сети, Primary Nework на Рис.1. На выходе сети расположен softmax слой, размер которого соответствует числу возможных типов.

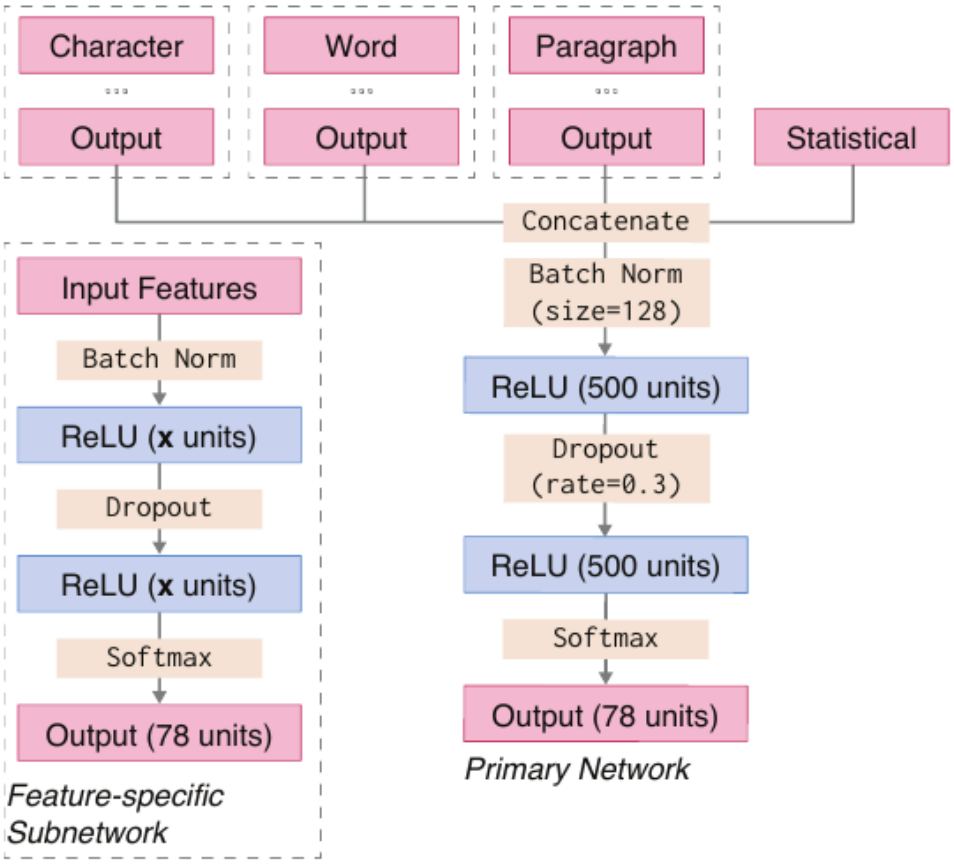

Рис. 1: Архитектура модели Sherlock. Источник [7].

2.1.2 SeLaB

Авторы данной работы [9] отмечают, что все методы, которые применялись ранее, для решения этой задачи не используют контекст таблицы и работают только с колонками по отдельности. Из-за этого модели упускают зависимости между данными в разных колонках, которые могли бы быть полезны для определения типов.

Чтобы решить данную проблему была предложена модель SeLaB, которая выполняет две итерации при обработке данных. На первом этапе модель находит контекстно-независимые предсказания для каждого столбца. Из предсказанных типов получается векторное представление для контекста таблицы. На втором этапе модель получает на вход, помимо значений из колонки, контекст таблицы.

Благодаря использованию контекста таблицы SeLaB смогла достичь лучшей точности, чем все решения известные ранее.

2.1.3 Doduo

На данный момент модель Doduo представленная в [1] является лучшим решением данной задачи для английского языка. Основной идеей решения является применение предобученной языковой модели, которой на вход подается таблица целиком, а не отдельными колонками. Это позволяет использовать контекст таблицы при определении типа колонки. При этом типы определяются для всех колонок одновременно без использования дополнительных итераций.

В качестве языковой модели Doduo используется BERT [2], модель с архитектурой “трансформер”. Использование предобученной модели позволяет существенно снизить объем данных, необходимый для обучения.

Архитектура модели представлена на Рис.2. Подробно архитектура данной модели рассмотрена в 3.1.

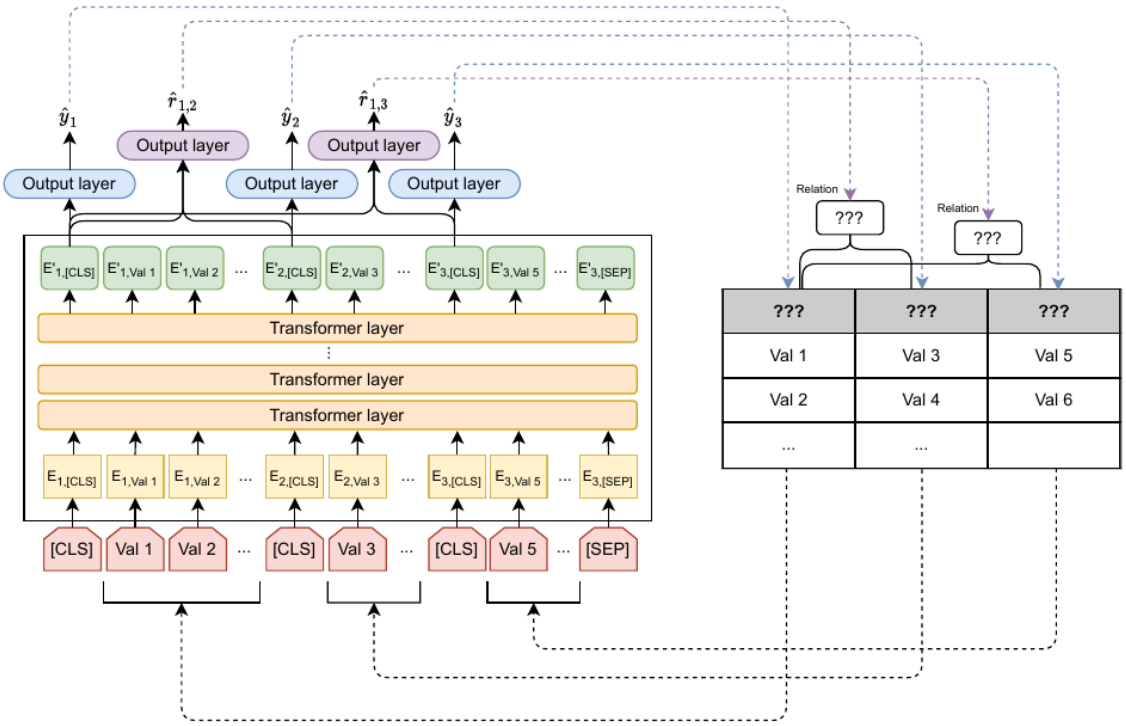

Рис. 2: Архитектура модели Doduo. Источник [1].

Также авторы данной работы показали возможность эффективного применения модели для кластеризации колонок. В данном сценарии модель выступает в роли энкодера эмбеддингов для столбцов, которые можно использовать для кластеризации.

2.2 Индустриальные продукты

На данный момент известно о как минимум трёх инструментах профилирования данных, обладающих возможностью предсказывать семантические типы колонок: SAS Viya[1], Tableau[2] и Talend[3] [4].

2.2.1 SAS Viya

SAS Viya — это облачная система для анализа и управления данными из разных источников. Sas Viya предоставляет множество инструментов для построения аналитических моделей, машинного обучения и

[1] https://www.sas.com/en_us/software/viya.html
[2] https://www.tableau.com/
[3] https://www.talend.com/

больших данных. Одной из функцией данного инструмента является определение семантических типов колонок в профилируемых данных. Поддерживаемые SAS Viya типы приведены в таблице 1.

Таблица 1: Поддерживаемые SAS Viya типы сущностей.

NAME	ORGANIZATION	ADDRESS	CITY
STATE/PROVINCE	POSTALCODE	COUNTRY	PHONE
EMAIL	DATE	UNKNOWN	URL
GENDER	MATCHCODE	PERSONAL_ID	ORGANIZATION_ID
GENERIC_ID	COUNTY	MARITAL_STATUS	

2.2.2 Tableau

Tableau — это инструмент для визуализации и профилирования данных. Tableau также способен автоматически определять типы столбцов в анализируемых данных, но при этом поддерживает сильно ограниченное количество типов: текст, даты, время, числа, boolean и географические названия.

2.2.3 Talend

Talend — платформа для интеграции, управления и обработки больших объёмов данных. С помощью регулярных выражений Talend способна определять 89 семантических типов. В таблице 2 перечислены типы, поддерживаемые Talend.

Таблица 2: Поддерживаемые Talend типы сущностей.

Address Line	Airport	Airport Code	Amex Card
Animal	Answer	AT VAT Number	Bank Routing Transit Number
BE Postal Code	Beverage	BG VAT Number	CA Province Territory
CA Province Territory Code	City	Civility	Color Hex Code
Company	Continent	Continent Code	Country
Country Code ISO2	Country Code ISO3	Currency Code	Currency Name
Data URL	DE Postal Code	DE Phone	Email
EN Month	EN Month Abbrev	EN Weekday	First Name
File URL	FR Postal Code	FR VAT Number	FR Phone
FR Insee Code	FR Commune	FR Departement	FR Region
FR Region Legacy	FR Social Security Number	Full Name	Gender
Geographic Coordinate	Geographic Coordinates	Geographic Coordinates (degree)	HDFS URL
HR Department	Industry	Industry Group	IBAN
IPv4 Address	IPv6 Address	ISBN-10	ISBN-13
Job Title	Language	Language Code ISO2	Language Code ISO3
Last Name	Measure Unit	MailTo URL	Money Amount (EN)
Money Amount (FR)	MAC Address	MasterCard	Month
Museum	MX Estado	MX Estado Code	Organization
Passport	SE Social Security Number	SEDOL	Street Type
UK Phone	UK Postal Code	UK Social Security Number	US Phone
US Postal Code	US Social Security Number	URL	US County
US State	US State Code	Visa Card	Web Domain
Weekday			

3 Разработка архитектуры

Для реализации решения описанного в данной работе были выбраны: язык программирования Python версии 3.10, фреймворк машинного обучения PyTorch и библиотека Transformers, предоставляющая интерфейс для предобученных моделей.

3.1 Архитектура модели Doduo

В качестве основы для данного решения была выбрана архитектура модели Doduo, поскольку Doduo показала наилучшие результаты на бенчмарках: VizNet [10] и WikiTables-TURL [8].

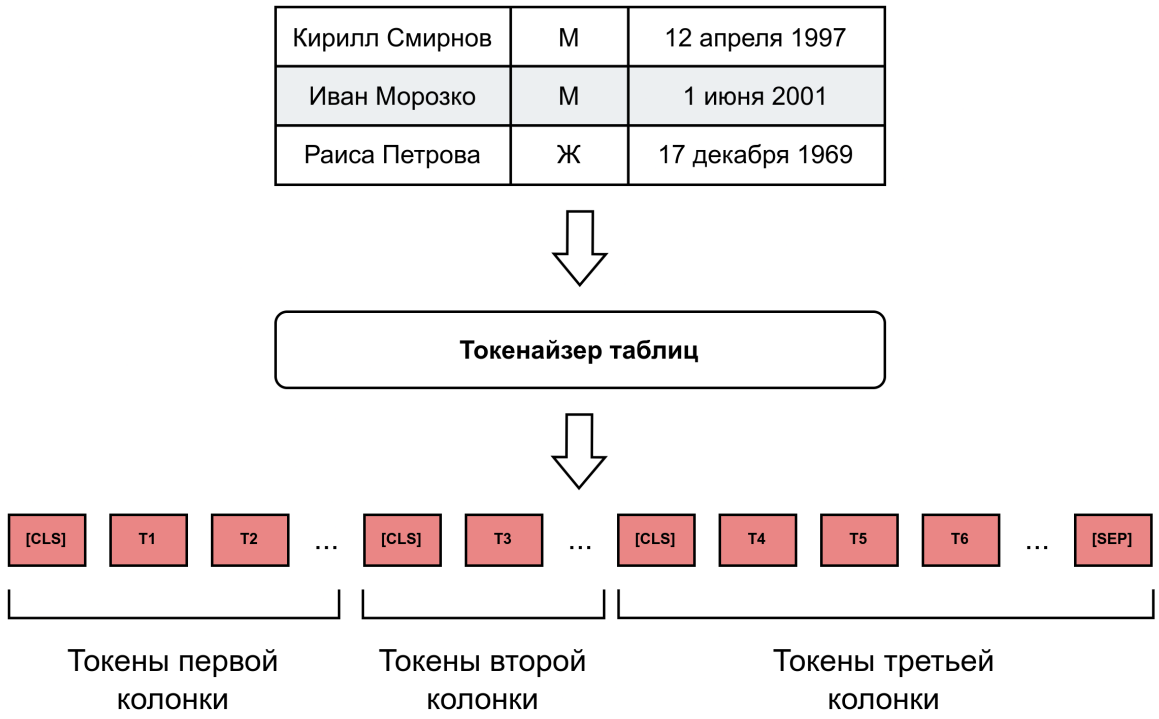

Рис. 3: Процесс токенизации таблицы.

Модель состоит из двух частей: языковой модели и классификатора. Также важной частью данного решения является токенайзер, который позволяет получать представление таблицы в виде последовательности токенов — чисел, каждому из которых соответствует эмбеддинг в первом слое модели (на Рис. 4 отмечены голубым цветом). В процессе

токенизации, Рис. 3, сначала находятся последовательности токенов для отдельных колонок, которые потом конкатенируются в одну. При конкатенации между полученными последовательностями вставляется специальный токен [CLS], который для модели обозначает начало колонки.

После токенизации языковая модель получает на вход последовательность токенов и преобразовывает их в векторные представления (на Рис. 4 отмечены зелёным цветом). Из них отбираются вектора соответствующие токенам [CLS] — они являются эмбеддингами для колонок. Полученные эмбеддинги затем используются классификатором для определения типов колонок.

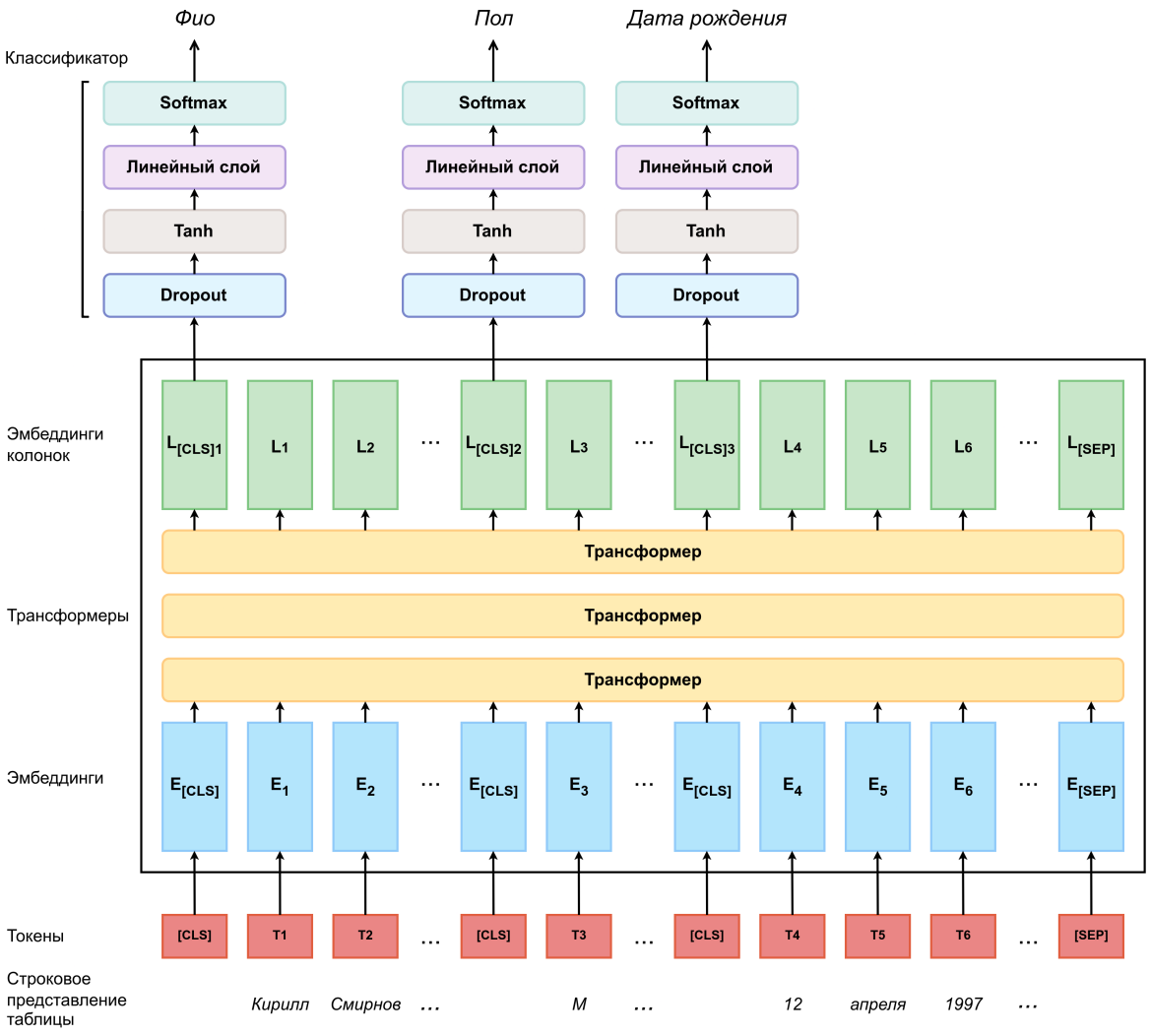

Рис. 4: Архитектура модели.

Модель Doduo обладает дополнительным классификатором для определения отношения между колонками (на Рис. 2 Output layer фиолетового цвета), например местоположение для человека может являться его местом рождения. Поскольку определение отношений между столбцами является отдельной задачей и для неё нет необходимых данных (требуется дополнительная ручная разметка данных), этот классификатор отсутствует в реализации модели в данной работе.

Также в классификаторе типов колонок в Doduo содержится два линейных слоя, а в реализации описанной в данной работе только один. Использование двух линейных слоёв приводило к ухудшению результатов, поэтому было принято решение использовать только один линейный слой.

3.2 Выбор языковой модели

Поскольку используемая в Doduo модель BERT обучалась на английском корпусе Wikipedia, она не подходит для fine-tuning'а, то есть её не выйдет качественно дообучить, и её необходимо заменить на аналогичную обученную на русскоязычном корпусе. Задача усложняется существенно ограниченными вычислительными ресурсами доступными для обучения. Возможным решением данной проблемы может быть использование дистилированной модели. Дистилляция позволяет создавать модели меньшего размера, при этом в большей степени сохраняя качество предсказаний больших моделей.

По итогам практического сравнения существующих русскоязычных моделей была выбрана rubert-tiny2[4]. Rubert-tiny2 является русскоязычной дистилированной версией BERT на основе моделей: RuBERT, LaBSE и Laser. В данной модели всего 12 миллионов параметров и три блока трансформеров, что позволило обучать её на видеокарте с малым количество видеопамяти.

[4]Репозиторий с моделью cointegrated/rubert-tiny2 на hugging face

4 Набор данных

В качестве исходного набора данных был выбран корпус русскоязычных таблиц RWT [3], состоящий из 1,2 миллиона таблиц, полученных с веб-сайта Wikipedia. В статьях на сайте Wikipedia используются таблицы разных видов: они могут иметь заголовок, как сверху, так и слева, могут иметь объединенные ячейки или обладать более сложной структурой. Пример неподходящих таблиц приведён на Рис. 5. При этом для решения задачи распознавания типов нужны таблицы, имеющие заголовок только сверху и структуру без объединенных ячеек. Из-за этого потребовалось создать новый набор данных, состоящий из отобранных таблиц, которые подходят для обучения.

Free the Universe [скрыть]	
Совокупная оценка	
Источник	**Оценка**
Metacritic	(68/100)[1]
Оценки критиков	
Источник	**Оценка**
AllMusic	★★★★★[2]
The Guardian	★★★★★[3]
NME	(7/10)[4]
Pitchfork Media	(5.7/10)[5]
Rolling Stone	★★★★½[6]
Spin	(7/10)[7]
Alternative Press	★★★½☆[8]
Drowned In Sound	(6/10)[9]

Имя	Возраст	Пол
Ваан (яп. ヴァン *Ван*)	17	мужской[7]

Основная статья: ***Ваан***

Ваан — главный герой *Final Fantasy XII*, семнадцатилетний сирота, потерявший родителей во время чумы, когда ему было 12 лет[8]. Его единственный брат Рекс погиб за два года до начала событий игры, во время вторжения империи Аркадия в Далмаску. Ваан зарабатывает себе на жизнь, работая помощником в магазине Мигело, выполняя различные поручения. Кроме того, он занимается карманными кражами, воруя при этом только у солдатов Аркадии — по его мнению, так он возвращает то,

Рис. 5: Примеры таблиц, которые не подходят для обучения модели, поскольку в них присутствуют объединенные ячейки.

4.1 Процесс обработки данных

Для избежания проблем с воспроизводимостью результатов и проблемы “Pipeline Jungles” [5], возникающей по мере усложнения процесса обработки данных, было принято решение о создании полноценного описания шагов обработки в среде Apache Airflow.

На Рис. 6 изображен ациклический ориентированный граф, где в вершинах располагаются наборы данных (например `raw_wikitables`) и действия (например `filter_dirty_tables`). Стрелки входящие в дей-

ствия обозначают зависимость их результатов от данных, а выходящие указывают на данные производимые действием.

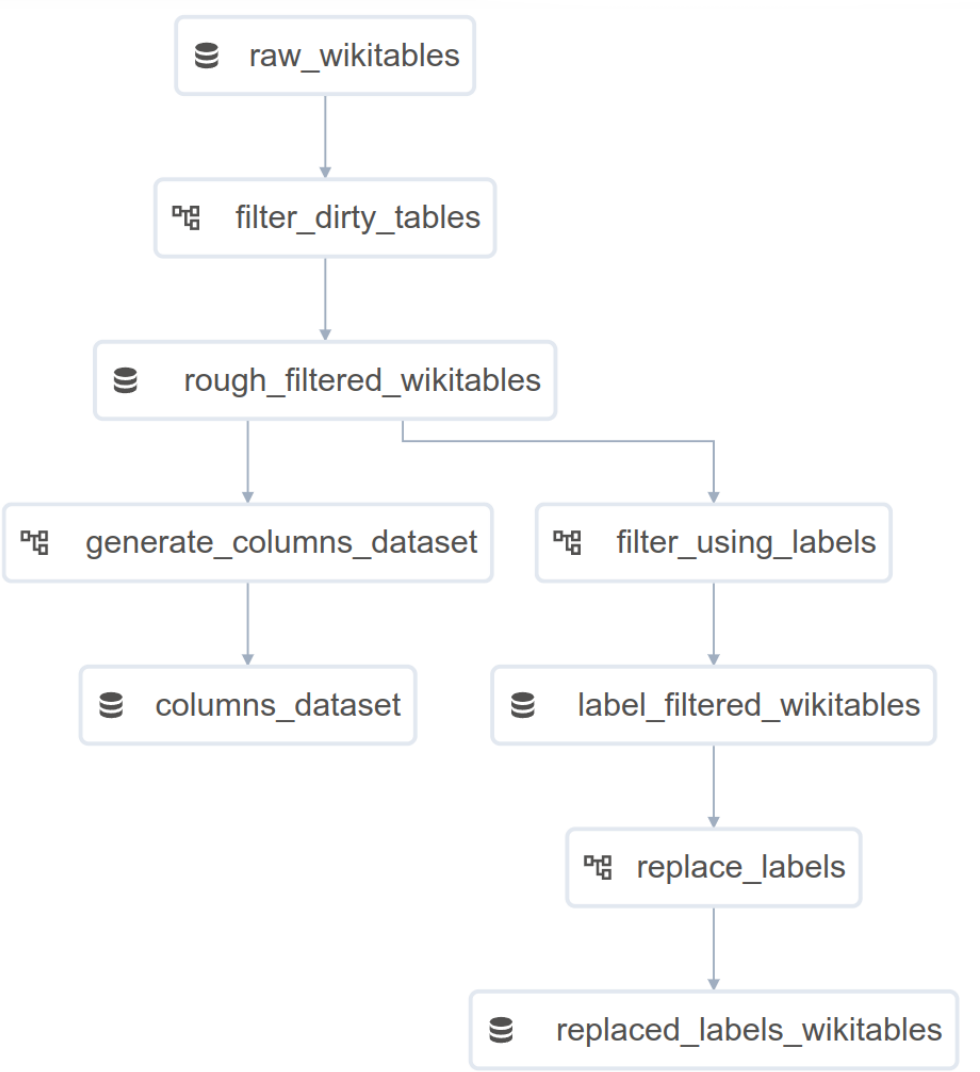

Рис. 6: Представление процесса обработки данных в виде ациклического графа в Apache Airflow.

Хранение промежуточных наборов данных позволило сэкономить время необходимое для получения финального набора данных, при внесении изменений в логику отдельных шагов. При обновлении какого-либо из наборов данных происходит автоматический запуск шагов, зависящих от этих данных. Это гарантирует, что никакие шаги обработки не будут пропущены в случае ошибки, а также, что конкретные версии наборов данных будут воспроизводимы.

4.2 Фильтрация данных

На первом этапе обработки данных с помощью регулярных выражений происходит фильтрация таблиц, имеющих заголовки слева и содержащих объединенные ячейки. Также отсеиваются таблицы без заголовков и таблицы, содержащие артефакты парсинга веб таблиц. На данном этапе отбрасывается половина таблиц исходного датасета. Дополнительно на этапе фильтрации происходит удаление характерных для таблиц из Wikipedia ссылок на источники вида: “[123]”, т.к. они являются шумом в данных и не обладают важной для задачи семантикой.

4.3 Выбор меток для сущностей

Набор меток был вручную отобран из наиболее часто встречающихся заголовков колонок в наборе данных. Также он был обогащён русскими метками, которые соответствуют использованным в SATO [6] английским меткам из корпуса наборов данных VizNet [10]. Например: language → язык, company → компания и другие.

Для создания набора данных `label_filtered_wikitables`, Рис. 6, используется расширенный набор меток, содержащий синонимичные. Например: “население”, “население, человек” и “численность населения”. Таблицы, содержащие метки, не входящие в отобранный набор меток, отсеиваются. При формировании конечного набора таблиц, все синонимичные метки заменяются одной меткой. В таблице 3 приведен список из 138 используемых меток без синонимичных.

Отдельно создается набор данных состоящий исключительно из колонок, которые имеют подходящие метки. Этот набор содержит в себе больше колонок, чем набор с таблицами, поскольку отсеиваемые таблицы могут могут содержать в себе колонки с подходящими метками. Данный набор применялся для расширения вокабуляра, описанного в 5.2.

Таблица 3: Список выбранных меток без синонимичных.

html	isbn	авиакомпания	автомобиль
автор	административный центр	адрес	актёр
актриса	альбом	английское название	артист
бортовой номер	версия	вес	вес, кг
вид спорта	владелец	возраст	время
год	года	год выпуска	годы
город	государство	гражданство	группа
дата	дата премьеры	дата рождения	день
диаметр, км	дистанция	длина	длительность
должность	жанр	звание	игрок
издание	издатель	имя	исполнитель
канал	категория	класс	классификация
клуб	код	кодовое имя	количество
команда	компания	компонент	континент
координаты	лига	лидер	материал
машина	место	местонахождение	место проведения
место рождения	месяц	награда	название
наименование	население, человек	научное название	национальность
неделя	номер	область	объём
описание	организация	оригинальное название	отдел
перевод	персонаж	песня	платформа
позиция	пол	порядок	провинция
продажи	продолжительность	продукт	продюсер
проект	производитель	процент	работа
размер	ранг	регион	режиссёр
результат	рейтинг	религия	роль
рост	русское название	сайт	сезон
сеть	символ	система	скорость ветра
событие	создатель	сокращение	состояние
спортсмен	ссылка	ссылки	стадион
стадия	статус	страна	сценарист
тип	титул	транскрипция	условные цветовые обозначения
фильм	фио	формат	цвет
частота	численность	число	штат
язык	языки		

5 Обучение модели

После применения к исходному набору данных всех необходимых преобразований был получен новый набор данных из 53783 таблиц, который подходит для обучения модели. Полученные данные были разделены на три части: учебную, валидационную и тестовую в пропорции: 70:10:20 соответственно.

5.1 Получение токенов из табличных данных для модели

Для модели Doduo токенизация происходит следующим образом: для каждого столбца таблицы получается его представление в виде строки, затем для каждой строки происходит токенизация, при этом число токенов ограничено сверху. Далее полученные последовательности токенов конкатенируются в одну.

У такого подхода есть существенный недостаток. Никак не учитывается число колонок в таблице, то есть для таблицы с 10 столбцами на вход будет подано до 320 токенов, но если колонок всего две, то до 64 токенов, при условии что на одну колонку выделяется до 32 токенов. Это неэффективно, поскольку на этапе обучения из-за выравнивания в батчах (подмножество выборки, на котором считается аппроксимированный градиент) модель будет получать много данных не несущих никакой информации, а на этапе предсказания получит меньше данных, чем могла бы. Для решения этой проблемы были предложены две модификации для процесса токенизации таблиц. На Рис.7 изображены распределения числа токенов между столбцами в зависимости от метода токенизации таблицы.

В качестве первой модификации изменим число токенов которые выделяются одному столбцу, теперь оно будет вычисляться по формуле:

$$\text{число токенов для колонки} = \frac{\text{максимальное число токенов}}{\text{число столбцов в таблице}}$$

Такая модификация позволяет модели получать на вход больше токенов,

если таблица имеет мало столбцов.

Поскольку число токенов, получаемых из строкового представления колонки зависит не только от числа строк в таблице, но и от типа данных (например для дат требуется больше токенов), возможна ситуация, где для одного столбца токенизация может дать небольшое число токенов, а для другого число превышающее выделенное ему число токенов. При этом останутся неизрасходованными токены первого столбца. Поэтому во второй модификации те токены, которые не были израсходованы распределяются между оставшимися столбцами. Новая формула выглядит следующим образом:

$$\text{число токенов для колонки} = \frac{\text{оставшееся число токенов}}{\text{число оставшихся столбцов в таблице}}$$

Данная модификация позволила улучшить результаты работы модели. Метрика macro F1 увеличилась с 0.823 до 0.885.

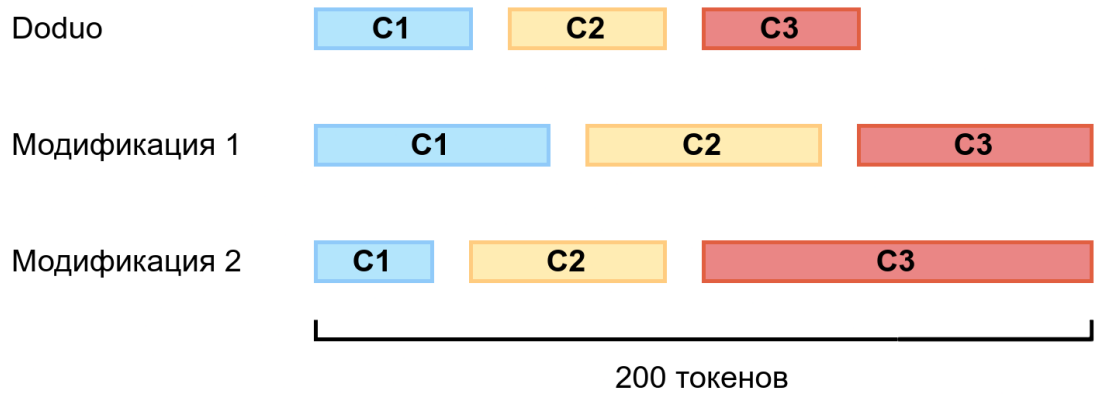

Рис. 7: Распределение числа токенов между колонками в зависимости от метода токенизации таблицы. Под Cn следует понимать токены колонки n.

5.2 Расширение вокабуляра токенайзера

В работе [13] авторы показывают улучшение работы модели BERT на задачах из медицинского домена после добавления новых токенов для

специфичных для этой области терминов. При рассмотрении результатов токенизации для отдельных лейблов было обнаружено, что изначального вокабуляра токенайзера мало для эффективной токенизации. Из-за этого происходит дробление слов на части, что ухудшает результат работы модели. Добавление новых токенов для меток: “вид спорта”, “столица”, “жанр” и “актёр” позволило повысить F1 для каждой из этих меток и общий macro F1 с 0.885 до 0.89.

Для нахождения новых токенов используется отдельный набор данных, состоящий из колонок. В колонках, соответствующих необходимым меткам, подсчитываются частоты для встречаемых слов. Наиболее часто встречаемые слова затем добавляются в вокабуляр. Также, поскольку каждому токену должен соответствовать свой эмбеддинг, увеличивается размер матрицы эмбеддингов внутри модели на число новых токенов.

5.3 Аугментация данных

Для того, чтобы уменьшить переобучение модели возникла необходимость в аугментации данных. Исходя из предположения, что изменение порядка строк и столбцов в таблице не изменяет её семантику в целом, был реализован следующий метод аугментации. На этапе обучения модель на вход получает таблицу из исходного набора данных, где случайным образом были переставлены столбцы и строки. Необходимо отметить, что аугментация не используется на этапах валидации и тестирования модели для воспроизводимости экспериментов.

5.4 Процесс обучения

Модель обучалась на компьютере со следующими характеристиками: 64 ГБ ОЗУ, Intel Xeon E5-2680 v4, NVIDIA GeForce GTX 1650. Оптимальные гиперпараметры для обучения были выбраны на основе проведенных экспериментов, также проведенные эксперименты показали результативность предложенных улучшений. Для записи результатов использовалась система MLflow, позволяющая собирать и анализировать

метрики и параметры моделей. Всего было сделано 68 запусков цикла обучения.

Для обучения использовался оптимизатор `AdamW` с параметром `learning rate` равным 2e−3. В качестве шедулера был выбран `OneCycleLR` с параметрами `pct_start` = 0.1 и `max_lr` = 2e−3. Модель обучалась 20 эпох. Для повышения стабильности при обучении использовалась аккумуляция градиентов, градиенты получаемые на батчах суммируются и сбрасываются при обновлении весов, которое происходит раз в четыре батча. В процессе обучения сохранялись веса для модели с лучшим macro F1 на валидационном наборе данных. После окончания обучения лучшая модель проверялась на тестовом наборе.

На Рис. 8 представлены графики изменения значений функции потерь на обучающей и валидационной выборках, а также изменение метрики macro F1 в процессе обучения.

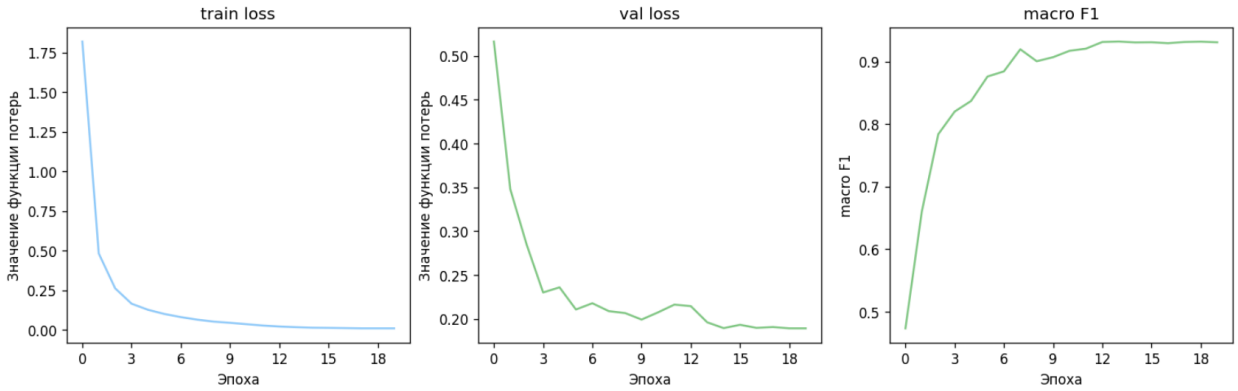

Рис. 8: Значения функции потерь на учебном и валидационном наборах, а также метрики macro F1 в процессе обучения.

5.5 Результаты обучения

В таблице 4 приведены результаты экспериментов, показавшие эффективность предложенных модификаций. В таблице 5 приведены значения метрик модели после завершения обучения,

Таблица 4: Значения метрик macro F1, micro F1 и weighted F1 на данных для разных модификаций решения.

Метрика / Вариант решения	Macro F1	Micro F1	Weighted F1
Без модификаций	0.823	0.942	0.942
Модифицированный токенайзер	0.885	0.962	0.961
Модифицированный токенайзер и расширенный вокабуляр	0.890	0.968	0.967

Таблица 5: Значения метрик macro F1, micro F1 и weighted F1 на валидационной и тестовых выборках после завершения обучения.

Метрика / Выборка	Macro F1	Micro F1	Weighted F1
Валидационная	0.932	0.971	0.971
Тестовая	0.890	0.968	0.967

Заключение

В ходе работы были достигнуты следующие результаты.

1. Выполнен обзор существующих на данный момент исследовательских решений Sherlock, SeLaB, Doduo и индустриальных решений: SAS Viya, Tableau и Talend для решения задачи определения семантического типа колонки. По результатам обзора была выбрана архитектура модели Doduo в качестве основы, поскольку Doduo на сегодняшний день решает аналогичную задачу для английского языка наилучшим образом по сравнению с другими моделями.

2. Разработана архитектура модели на основе архитектуры модели Doduo. Полученная модель адаптирована для работы с русским языком, а также способна работать в условиях ограниченных вычислительных ресурсов.

3. Подготовлен набор данных из 53 тысяч таблиц на основе RWT (корпуса русскоязычных таблиц), полученных из Wikipedia. Автоматизирован процесс обработки новых данных.

4. Выполнено обучение модели, проведены эксперименты, выбраны оптимальные гиперпараметры. Предложены модификации решения, способные улучшить конечный результат, а именно: предложен новый подход к токенизации таблиц и расширение вокабуляра.

5. Код модели опубликован на github.com, веса модели доступны на huggingface.co, датасет на kaggle.com. Демонстрационный стенд доступен на платформе streamlit.app.

Список литературы

[1] Annotating Columns with Pre-trained Language Models / Yoshihiko Suhara, Jinfeng Li, Yuliang Li et al. // Proceedings of the 2022 International Conference on Management of Data. — Association for Computing Machinery, 2022. — ISBN: 9781450392495. — URL: https://doi.org/10.1145/3514221.3517906.

[2] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova // CoRR. — 2018. — Vol. abs/1810.04805. — arXiv : 1810.04805.

[3] Fedorov Platon, Mironov Alexey, Chernishev George A. Russian Web Tables: A Public Corpus of Web Tables for Russian Language Based on Wikipedia // CoRR. — 2022. — Vol. abs/2210.06353. — arXiv : 2210.06353.

[4] Hameed Mazhar, Naumann Felix. Data Preparation: A Survey of Commercial Tools // SIGMOD Rec. — 2020. — dec. — Vol. 49, no. 3. — P. 18–29. — URL: https://doi.org/10.1145/3444831.3444835.

[5] Hidden Technical Debt in Machine Learning Systems / D. Sculley, Gary Holt, Daniel Golovin et al. // Advances in Neural Information Processing Systems / Ed. by C. Cortes, N. Lawrence, D. Lee et al. — Vol. 28. — Curran Associates, Inc., 2015. — URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/86df7dcfd896fcaf2674f757a2463eba-Paper.pdf.

[6] Sato: Contextual Semantic Type Detection in Tables / Dan Zhang, Yoshihiko Suhara, Jinfeng Li et al. // CoRR. — 2019. — Vol. abs/1911.06311. — arXiv : 1911.06311.

[7] Sherlock: A Deep Learning Approach to Semantic Data Type Detection / Madelon Hulsebos, Kevin Hu, Michiel Bakker et al. // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge

Discovery & Data Mining. — KDD '19. — New York, NY, USA : Association for Computing Machinery, 2019. — P. 1500–1508. — URL: https://doi.org/10.1145/3292500.3330993.

[8] TURL: Table Understanding through Representation Learning / Xiang Deng, Huan Sun, Alyssa Lees et al. // CoRR. — 2020. — Vol. abs/2006.14806. — arXiv : 2006.14806.

[9] Trabelsi Mohamed, Cao Jin, Heflin Jeff. Semantic Labeling Using a Deep Contextualized Language Model // CoRR. — 2020. — Vol. abs/2010.16037. — arXiv : 2010.16037.

[10] VizNet: Towards a large-scale visualization learning and benchmarking repository / Kevin Hu, Neil Gaikwad, Michiel Bakker et al. // Proceedings of the 2019 Conference on Human Factors in Computing Systems (CHI). — ACM, 2019.

[11] Zhang Li, Zhang Shuo, Balog Krisztian. Table2Vec: Neural Word and Entity Embeddings for Table Population and Retrieval // Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. — SIGIR'19. — New York, NY, USA : Association for Computing Machinery, 2019. — P. 1029–1032. — URL: https://doi.org/10.1145/3331184.3331333.

[12] Zhang Shuo, Balog Krisztian. Ad Hoc Table Retrieval Using Semantic Similarity // Proceedings of the 2018 World Wide Web Conference. — WWW '18. — Republic and Canton of Geneva, CHE : International World Wide Web Conferences Steering Committee, 2018. — P. 1553–1562. — URL: https://doi.org/10.1145/3178876.3186067.

[13] exBERT: Extending Pre-trained Models with Domain-specific Vocabulary Under Constrained Training Resources / Wen Tai, H. T. Kung, Xin Dong et al. // Findings of the Association for Computational Linguistics: EMNLP 2020. — Online : Association for Computational Linguistics, 2020. — . — P. 1433–1439. — URL: https://aclanthology.org/2020.findings-emnlp.129.