ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Московский физико-технический институт
(национальный исследовательский университет)»
(МФТИ, Физтех)

На правах рукописи

Сомов Олег Дмитриевич

Методы разработки text-to-SQL систем в условиях сдвига обучающей выборки

Специальность: 1.2.1. Искусственный интеллект и машинное обучение

ДИССЕРТАЦИЯ
на соискание учёной степени
кандидата технических наук

Научный руководитель:
доктор компьютерных наук
Тутубалина Елена Викторовна

Долгопрудный — 2024

Оглавление

Стр.

Стр.

Введение

Актуальность темы. Задача text-to-SQL является одним из направлений исследований в области обработки естественного языка, в частности одной из подзадач семантического
парсинга. Семантические парсеры представляют собой программные системы, которые преобразуют естественный язык в
структурированное логическое представление, понятное компьютеру. Практическим примером работы семантических пасеров, является преобразование из команд или вопросов на естественном языке в код (Python, Java), в запрос к базе данных (SQL, SPARQL) или в логическое выражение (Lambda calculus, LISP).
Семантические парсеры, как ключевой элемент в системах обработки естественного языка, находят применение в различных областях, таких как информационный поиск, диалоговые системы и анализ текстов [1, 2, 3, 4].

С развитием глубокого обучения семантические парсеры превратились из комплексных, многокомпонентных систем, зачастую основанных на детерминированных алгоритмах, в интегральные решения, основанные на моделях машинного обучения. На текущий момент прикладные семантические парсеры, решающие задачу преобразования текста в код или запрос превратились в такие направления NLP, как задачи text-to-SQL, text-to-code, text-to-logical form [5, 6]. Как представлено на Рисунке 1, вместо разработки и обучения множества независимых компонент стало актуально обучение одной text-to-text модели машинного обучения на парах: **выражение на естественном языке** x **- выражение на формальном языке** y.

Это превращение решило проблему каскадности в семантических парсерах (последовательное накопление системой ошибок компонент), но из-за этого же, в задаче стали более явно представлены главные проблемы машинного обучения - отсутствие данных, переобучение и слабая генерализация моделей [7, 8].

Задача text-to-SQL - одна из самых популярных среди остальных задач семантического парсинга. Рост популярности связан с общей тенденцией в области решений на основе искусственного интеллекта справляться с глобальными и сложными задачами. Если в 2010-ых годах, машинное обучение

использовалось для простых атомарных задач, таких как проставление семантических тегов или выделение именованных сущностей, то в 2020-ых годах появляются комплексные модели, способные решать сложные задачи - например, определение ключевых идей научной статьи или генерация рабочего SQL запроса. Сегодня актуальность text-to-SQL задачи обоснована необходимостью обращаться напрямую к корпоративным данным на естественном языке. В первую очередь, это увеличивает скорость принятия решений. Также это влияет на качество решений, так как принимающие их сотрудники могут сами исследовать базу данных компании, без привлечения третьих лиц в качестве аналитиков.

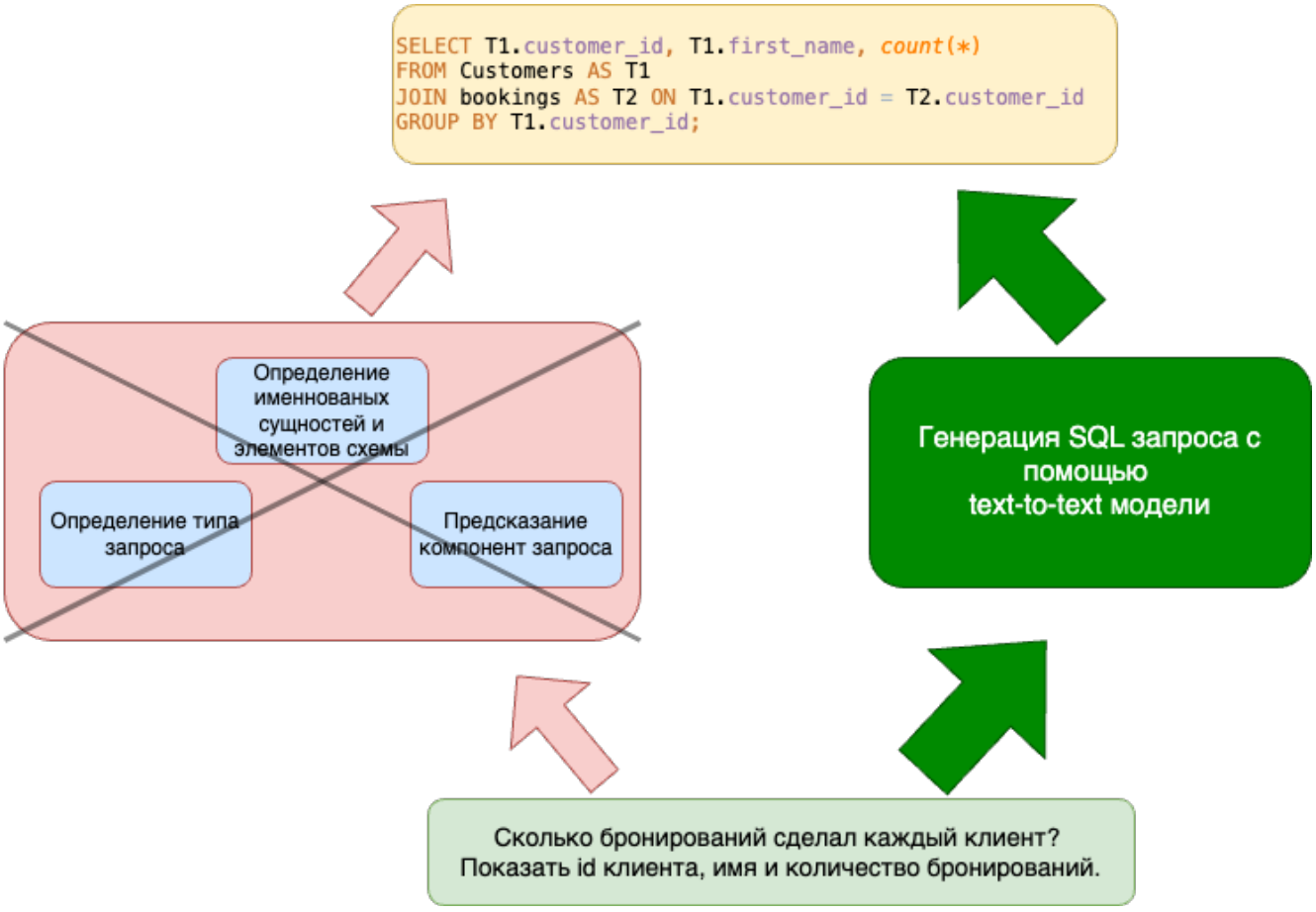

Рисунок 1 — Переход от многокомпонентных text-to-SQL моделей к однокомпонентным text-to-text моделям.

Переход от правиловых и многокомпонентных решений более остро обнаружил проблемы машинного обучения в рамках решения задачи text-to-SQL. Рассмотрим ключевые принципы машинного обучения и их

отражение в необходимых задачах для построения эффективных и надежных text-to-SQL систем:

1. Для обучения используются исторические данные;
2. Знания не передаются в явном алгоритме, а в виде параметризируемой функции f, параметры θ которой подбираются на обучающей выборке;
3. Машинное обучение использует обнаруженные закономерности для принятия решений в условиях неопределенности.

Первый принцип определяет формат обучающей выборки задачи - text-to-SQL модели обучаются на парах x и y, это обучение с учителем. Данные должны быть корректны и должны покрывать все сценарии использования text-to-SQL систем. Эффективный и качественный сбор данных для text-to-SQL моделей является актуальной темой, так как сбор данных для семантических парсеров в целом является дорогостоящей и скрупулёзной процедурой. Такую разметку обычно выполняют высококвалифицированные специалисты, знакомые с языком SQL. Если собранные данные в обучающей выборке однообразны (например, синтетически сгенерированы) или просто некорректны, это приведет к существенному снижению точности и эффективности text-to-SQL моделей при использовании на реальных данных [9].

Второй принцип подсказывает, что функция f, чьи параметры θ подобраны на обучающей выборке, должна быть способна делать точные предсказания за пределами обучающей выборки - на тестовой выборке. Способность к обобщению на новые данные называется генерализацией. Модели машинного обучения с учителем обучаются по принципу минимизации эмпирического риска. Условие успешного применения принципа минимизации эмпирического риска состоит в **одинаково и равномерно распределенных данных обучения и тестирования**. Но если это не так и данные обучающей выборки отличаются от тестовой – модель проявляет переобучение и слабую генерализацию [10].

Одна из причин проявления слабой генерализации в моделях машинного обучения – **сдвиг обучающей выборки (distribution shift)**. Сдвиг обучающей выборки – это серьезная проблема, поскольку модели, обученные на данных, которые не отражают реальные условия применения, при тестировании проявляют низкую способность к генерализации [10].

Домен семантических парсеров очень склонен к сдвигу обучающей выборки. Обычно проблема сдвига решается дополнительным сбором данных, но в семантическом парсинге сбор данных трудоемок - надо сформировать корректную команду на естественном языке и написать выражение на формальном языке. А для задачи text-to-SQL для верного SQL запроса надо иметь доступ к соответствующей базе данных [11, 12]. Среди ключевых сдвигов обучающей выборки в text-to-SQL задаче - композиционный, доменный и ковариационный [13]. Композиционный и доменный сдвиги относятся к полному виду сдвига. В этих случаях данные зависимой и независимой переменных обучающей и тестовой выборки значительно отличаются друг от друга. Для программ генерации любого кода самым распространённым сдвигом является композиционный сдвиг и доменный сдвиг.

Способность к композиционному обобщению в контексте семантических парсеров относится к их умению эффективно обрабатывать и понимать новые составные структуры и выражения на основе ранее изученных элементарных компонент. Свойственная задаче семантического парсинга композиционность не представляет проблемы для человека, так как по принципу композиционности - значения сложных словосочетаний определяются значениями их частей, поэтому полностью определяются и понятия новых сочетаний знакомых элементов [14, 15]. Например, человек узнав слово «арапник» (длинный охотничий кнут, используемый в охоте) без проблем сможет понять это слово в других контекстах уже известных слов и составлять новые предложения с этим словом. Для модели машинного обучения принцип композиционности языка SQL представляет собой сдвиг обучающей выборки и является проблемой [16]. Доменный сдвиг проявляется, когда модели необходимо генерализировать на новые, невиданные ранее элементы схемы, новые базы данных и верно учитывать их в целевом SQL запросе.

Ковариационный сдвиг - сдвиг независимой переменной. В задаче text-to-SQL он проявляется как в появлении новых формулировок вопросов к известным структурам SQL, так и в появлении вопросов, на которые нельзя ответить без привлечения внешних знаний или в рамках данных базы данных.

В данной работе исследуется поведение text-to-text моделей в представленных сдвигах, разрабатывается метод с лучшей композиционной генерализацией и предлагаются методы разработки text-to-SQL систем в условиях сдвига обучающей выборки.

Третий принцип обращает наше внимание на оценку уверенности моделей машинного обучения в контексте обучения text-to-SQL моделей. При использовании text-to-SQL моделей в производстве, пользователь должен быть специфичен в обращении к базе знаний, понимать ее атрибуты и представленные таблицы - иначе существует высокий риск генерации некорректного SQL запроса, который предоставит ложный ответ. text-to-SQL модели обучаются на парах выражение на естественном языке x - SQL запрос y. Примеров запросов, когда на данный вопрос x SQL запроса не существует, в датасете машинного обучения нет, так как таких примеров можно собрать бесконечное количество и их наличие приведет больше к переобучению и падению качества на целевой задаче генерации SQL, чем к способности модели уметь отвергать неподходящие вопросы [17].

Следовательно, вопрос, несоответствующий базе данных, является **out-of-distribution (o.o.d.)** примером для text-to-SQL модели. Как упоминалось выше, эти примеры являются примером ковариационного сдвига – даже если добавить неотвечаемые пары в обучающую выборку, всегда найдутся неотвечаемые вопросы для любой вопросно-ответной системы (например, text-to-SQL система не сможет ответить на вопрос – *Почему трава зеленая?*).

Одним из методов детекции o.o.d. примеров в моделях является метод оценки уверенности модели (например, на основании оценки неопределенности рассчитаной с помощью выходного softmax распределения целевой переменной или скрытых представлений нейронной сети). Ожидается, что качественная text-to-SQL модель будет иметь следующее поведение в данных сценариях:

- Если вопрос относится к базе данных и модель уверена в его генерации, она, совместно с самим запросом, выдает высокую оценку уверенности;
- Если модель не уверена в генерации запроса или вопрос не относится к используемой базе данных, она, совместно с самим запросом, выдает низкую оценку уверенности.

Интерпретируемая оценка уверенности модели в своем предсказании делает шаг навстречу надежным и безопасным системам искусственного интеллекта [18].

Подводя итог актуальности темы, необходимо обратить внимание на выделенные проблемы и мотивацию их решения в контексте всей области искусственного интеллекта и, в частности, семантических парсеров.

Технологический прогресс сегодня движется к построению моделей искусственного интеллекта, способных не только решать базовые задачи, но и способных к рассуждению, созданию новых идей. Такая модель должна обладать высокой способностью к различным типам генерализации, быть надежной и интерпретируемой. Для обучения таких моделей необходимо огромное количество данных, которые часто бывают некачественными и противоречивыми. Более того, бенчмарки для оценки этих способностей моделей требуют согласованности между исследователями и самим обществом. В тоже время, область семантических парсеров обладает готовыми средствами верификации результатов моделей. Прогресс в этой области позволит перенести способности частных моделей на общие модели искусственного интеллекта и строить качественные и надежные системы.

Объектом исследования в диссертации выступают text-to-SQL модели. **Предметом** исследования является методы разработки устойчивых и надежных text-to-SQL моделей в условиях сдвига обучающей выборки.

Целями диссертационной работы являются:

– Определение закономерностей поведения предобученных языковых моделей в условиях сдвига обучающей выборки с точки зрения генерализации и интерпретируемости в задаче text-to-SQL для формирования выводов об области применимости таких систем в производстве;
– Разработка качественной и надежной системы text-to-SQL в условиях, приближенных к производству.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Разработать качественный text-to-SQL датасет для достоверности экспериментов и оценки сложности разработки такого набора данных;
2. Разработать сдвиги обучающей выборки для оценки генерализации и интерпретации предобученных языковых моделей в задаче text-to-SQL в условиях сдвига;
3. Определить метрики оценки уровня генерализации и интерпретации языковых моделей и провести эксперименты на наборе предобученных языковых моделей;
4. Разработать text-to-SQL систему, близкую к использованию в производстве, с устойчивостью к различным видам сдвига.

Научная новизна:

- Разработан первый русскоязычный text-to-SQL датасет PAUQ и определены правила будущей адаптации text-to-SQL датасетов на русский язык;
- Разработаны методы искусственного разбиения данных, специфичные домену SQL, для оценки композиционной генерализации text-to-SQL моделей, которые позволяют более точно, нежели предшествующие методы разбиения [11, 16], оценить генерализацию;
- Сформированы новые выводы о поведении больших языковых моделей в аспектах мультиязычной, многозадачной, доменной и композиционной генерализации и об интерпретируемости моделей в условиях сдвига обучающей выборки в задаче text-to-SQL.

Практическая значимость. Результаты работы могут быть полезны широкому кругу специалистов в области обработки естественного языка и баз данных:

- Разработанный датасет PAUQ позволит организациям и исследователям обучать и тестировать системы перевода с русского языка на SQL;
- Методы эмуляции сдвига могут быть применены для тестирования генерализации text-to-SQL моделей в условиях сдвига обучающей выборки;
- Определенные закономерности поведения языковых моделей в условиях сдвига обучающей выборки помогут исследователям и разработчикам лучше понимать ограничения современных больших языковых моделей, оптимальным образом собирать text-to-SQL данные для обеспечения лучшего качества систем перевода;
- Разработанная text-to-SQL система может являться примером для создания качественных и надежных систем перевода с естественного языка на SQL;
- Предложенный метод дообучения больших языковых моделей может быть внедрен в процесс обучения text-to-query моделей для улучшения композиционной генерализации (улучшение способности модели создавать новые формы запросов);
- В результате исследований в виртуальный ассистент “Салют” компании “Сбер” была внедрена вопросно-ответная система перевода вопроса

в запрос, что позволило виртуальному ассистенту отвечать на фактологические вопросы.

Методология и методы исследования. В данной диссертационной работе применялись нейросетевые методы обработки естественного языка.

Основные положения, выносимые на защиту:

1. Разработан русскоязычный датасет text-to-SQL PAUQ;
2. Предложены методы контролируемых сдвигов распределения для эмулирования различных видов сдвига в задаче text-to-SQL и определены закономерности поведения предобученных языковых моделей в условиях сдвига обучающей выборки с точки зрения генерализации и интерпретируемости;
3. Разработан новый метод улучшения композиционной генерализации text-to-SQL и text-to-SPARQL моделей;
4. Разработана надежная и устойчивая text-to-SQL система в условиях сдвига обучающей выборки.

Апробация работы. Основные результаты работы докладывались на следующих конференциях и семинарах:

1. Международная конференция по компьютерной лингвистике и интеллектуальным технологиям “Диалог” (онлайн, 17-20 июня, 2020 г.);
2. Всероссийская конференция по искусственному интеллекту “Лето с AIRI” (Сириус, Россия, 17-27 июля, 2022 г.);
3. Международная конференция “Empirical Methods in Natural Language Processing” (Абу-Даби, ОАЭ, 7-11 декабря, 2022 г.);
4. Семинар “Generalization in NLP” международной конференции “Empirical Methods in Natural Language Processing” (Сингапур, 6-10 декабря, 2023 г.);
5. 66-ая конференция МФТИ, (Москва, 1-6 апреля, 2024 г.);
6. Семинар “Clinical NLP” международной конференции “North American Chapter of the Association for Computational Linguistics” (Мексика, 16-21 июня, 2024 г.);
7. Семинар AIRI по NLP “ИИшница” (Москва, 29 февраля, 2024 г.);
8. Семинар Sber AI Community (Москва, 26 июня, 2024 г.);
9. Научный семинар кафедры Банковских информационных технологий, МФТИ (Москва, 11 июля, 2024 г.);

10. Научный семинар НИВЦ МГУ (Москва, 15 октября, 2024 г.);
11. Научный семинар ИПУ РАН (Москва, 16 октября, 2024 г.).

Достоверность подтверждается экспериментами, проведенными в соответствии с общепринятыми стандартами, взаимосвязью данных экспериментов и научных выводов, сделанных в работе, квалифицированной апробацией на международных и российских научных конференциях. Достоверность выводов и качество исследований подтверждается разработкой и запуском вопросно-ответной системы в виртуальном ассистенте “Салют” компании “Сбер”, которая позволяет виртуальному ассистенту отвечать на фактологические вопросы.

Личный вклад. Автором проведено исследование предметной области, выполнен основной объем теоретических и экспериментальных исследований, изложенных в диссертации, разработана программная система на основе созданных методов. В работах [19, 20, 21, 22, 23] автором проведено исследование предметной области, выполнен основной объем теоретических и экспериментальных исследований, изложенных в публикациях. Тутубалиной Е.В. принадлежит постановка задачи и практические рекомендации для выполнения работы. В работе [24] автору принадлежит постановка задачи и практические рекомендации для выполнения работы. В разработке и запуске вопросно-ответной системы внедренной в производство в виртуальном ассистенте “Салют” компании “Сбер” автор принимал ключевое участие. Основные результаты по теме диссертации изложены в 6 печатных изданиях, 3 из которых изданы в периодических научных журналах, индексируемых Web of Science и Scopus.

Глава 1. Адаптация text-to-SQL данных и моделей на русский язык

Эта глава посвящена адаптации text-to-SQL датасета и сопутствующим задачам и вопросам. В этой главе формально определена постановка задачи text-to-SQL, а также рассмотрены методы обучения text-to-SQL моделей. Рассмотрены text-to-SQL датасеты, их особенности и метрики оценки. В главе содержится описание адаптации первого русского датасета text-to-SQL PAUQ. Завершается глава ответами на исследовательские вопросы, которые позволят упростить адаптацию будущих датасетов text-to-SQL и повысить эффективность обучения text-to-SQL моделей на русском языке.

1.1 Постановка задачи text-to-SQL

text-to-SQL – является одной из подзадач задачи семантического парсинга, задачи перевода выражения на естественном языке в формальное выражение. В современном NLP задача text-to-SQL формулируется как задача языкового моделирования в формате sequence-to-sequence. Пример обучающей выборки (X,Y) представлен как набор пар размером N: выражение на естественном языке x и соответствующее ему выражение на языке SQL y. Последовательность токенов $x_1, x_2, ..., x_M$, где M — количество токенов в предложении x, обозначает x. В свою очередь, последовательность токенов $y_1, y_2, y_3, ..., y_K$, где K — количество токенов в SQL выражении y, обозначает y. Специфично задаче text-to-SQL, у каждого вопроса x_i есть дополнительная информация s_i – контент базы данных, которой адресован вопрос. Например, схема базы данных - информация о таблицах, атрибутах (столбцов/колонок таблиц), связях и типах данных атрибутов.

Реляционная база данных обозначается как D. Схема базы данных S для D включает (1) набор из N таблиц $\mathcal{T} = \{t_1, t_2, \cdots, t_N\}$, (2) набор атрибутов $\mathcal{C} = \{c_1^1, \cdots, c_{n_1}^1, c_1^2, \cdots, c_{n_2}^2, \cdots, c_1^N, \cdots, c_{n_N}^N\}$, связанных с таблицами, где n_i — количество атрибутов в i-й таблице, (3) набор внешних ключевых связей $\mathbb{R} = \{(c_k^i, c_h^j) \mid c_k^i, c_h^j \in \mathcal{C}\}$, где каждая (c_k^i, c_h^j) обозначает связь внешнего ключа между атрибутом c_k^i и атрибутом c_h^j.

Задача машинного обучения для обучения text-to-SQL sequence-to-sequence модели формулируется следующим образом:

$$L(X, Y, S, \theta) = -\frac{1}{N} \sum_{i=0}^{N} \sum_{k=0}^{K} logP(y_{i,k}|x_i, y_{i,<k}, s_i, \theta) \rightarrow min$$

Цель оптимизации – минимизация эмпирического риска функции правдоподобия P языкового моделирования, параметризованной θ, на заданной подвыборке (X, Y) размером N для последовательностей длины K. Каждая пара выборки (X,Y) соответствует своей базе D со схемой S.

Результат предсказания модели P для вопроса x и схемы s, соответствующей базе данных, представляет собой последовательность токенов $\tilde{y}$, которая состоит из SQL токенов $\tilde{y_1}$,$\tilde{y_2}$, $\tilde{y_3}$, ..., $\tilde{y_K}$. Каждый токен $\tilde{y_k}$ предсказывается с помощью моды исходного softmax-распределения по словарю $\mathcal{V}$ целевого языка (например, SQL):

$$\tilde{y}_k = \arg\max_{\tilde{y}_k \in \mathcal{V}} P(\tilde{y}_k \mid x, \tilde{y}_{<k}, s, \theta)$$

1.2 Особенности данных в задаче text-to-SQL

Сбор данных для обучения семантических парсеров играет важную роль в развитии и улучшении понимания естественного языка компьютером. Современные приложения, такие как виртуальные помощники [25], чат-боты [26], и вопросно-ответные системы [27], требуют большого количества качественных и разнообразных данных. Это подчеркивает важность сбора разнообразных и репрезентативных данных. В современном NLP наблюдается тенденция к переходу от комплексных многокомпонентных систем к интегральным малокомпонентным решениям. Это особенно заметно по современным бенчмаркам семантического парсинга - Spider[1], GrailQA [2], DS-1000[3], Spider-v2[4]. Этот переход оказывает значительное влияние на данные, используемые для обучения семантических парсеров.

[1]https://yale-lily.github.io/spider

[2]https://dki-lab.github.io/GrailQA/

[3]https://ds1000-code-gen.github.io/model_DS1000.html

[4]https://spider2-v.github.io/#leaderboard

Комплексные многокомпонентные системы обычно включают множество взаимосвязанных компонент. Каждый из них требует своего набора обучающих данных, таких как данные для извлечения отношений, сопоставления сущностей и классификации шаблонов логических выражений. Распространенным явлением является зависимость обучающих данных одной компоненты от обучающих данных других компонент. При этом данные собираются таким образом, чтобы точно отражать уникальный набор задач и сценариев, ожидаемых от конкретной компоненты. Таким образом, высокая связность компонент приводит к усложнению процесса сбора и управления данными, а также разработки таких решений.

Интегральные малокомпонентные решения, напротив, стремятся к упрощению архитектуры, объединяя множество компонент в одну. Примерами трансформированных задач из комплексных многокомпонентных систем семантического парсинга стали задачи text-to-query (например, text-to-SQL и text-to-SPARQL задачи) и text-to-code, представляющие собой пары **выражение на естественном языке** x и **выражение на формальном языке** y. Специфика сбора данных для этих задач также изменилась. Можно выделить следующие характеристики датасетов такого типа:

- Соответствие выражений на естественном языке x реальным вопросам пользователей. Это помогает моделям лучше обучаться на естественном стиле вопросов и повышает их применимость к реальным сценариям.
- Обучающие данные должны охватывать различные сценарии использования целевого языка семантического парсера.
- Наиболее полное охватывание грамматики целевого языка запросов датасетом – наличие в данных различных шаблонов запросов y, которые включают разнообразные операторы и функции языка. Это обеспечивает способность моделей обрабатывать широкий спектр выражений на естественном языке после обучения.
- Пары обучающих данных должны проходить экспертную ревизию для обеспечения точности, качества и эффективности логических выражений, связанных с каждым выражением на естественном языке.

Такие характеристики обучающих данных позволяют создать высококачественные семантические парсеры.

1.3 text-to-SQL наборы данных

text-to-SQL датасеты ориентированы на преобразование запросов на естественном языке в структурированный язык запросов SQL. Пара вопрос-запрос выглядит следующим образом:

– x: Сколько глав департамента старше 56 лет?

– y: select count (*) from head where age $>$ 56

В дополнение к каждой паре есть соответствующая база данных D, относительно которой исполняется SQL запрос. Схема базы данных задана с помощью DDL-выражения, описывающего схему базы данных S, отношения между таблицами и типы атрибутов.

Наиболее известные text-to-SQL датасеты представлены на английском языке. Существуют аналоги на других языках, но они являются переводами существующих наборов данных. Все актуальные датасеты представлены в диалекте SQLite [28] для SQL выражений и формата баз данных. Далее приведены ключевые text-to-SQL датасеты:

– **Spider** [8] - text-to-SQL датасет, аннотированный 11 студентами Йельского университета. Размер датасета составляет 10,181 пар. Особенность датасета – в тестовой выборке присутствуют базы данных, которых не было на этапе обучения. В датасете равномерно представлены запросы разной сложности, с несколькими `JOIN` операциями, вложенными запросами с операциями `UNION`, `INTERSECT`.

– **WikiSQL** [29] - text-to-SQL датасет из 80,654 аннотированных вручную пар вопросов на естественном языке, SQL-запросов и таблиц SQL, извлеченных из 24,241 HTML-таблиц Википедии. Является первым обретшим популярность text-to-SQL датасетом. В WikiSQL представлены простые запросы к одной таблице с несколькими `WHERE` условиями и операциями агрегации.

– **BIRD** [30] - text-to-SQL датасет с большими базами данных. BIRD включает более 12,751 уникальных пар вопрос-SQL, 95 больших баз данных общим объемом 34 ГБ. Он охватывает более 37 профессиональных областей, таких как блокчейн, хоккей, здравоохранение и образование. Особенность датасета в построении не просто корректных SQL запросов, но и эффективных с точки зрения

исполнения. Несмотря на большую актуальность для исследования для text-to-SQL задачи, в BIRD датасете примерно 30% ошибок и вопросов, на которое нельзя однозначно ответить с помощью базы данных, что делает использование этого датасета во время написания диссертации невозможным [31, 32]. В BIRD представлены более сложные запросы по сравнению со SPIDER датасетом с увеличенной длинной запросов и количеством `JOIN` операций.

- **EHRSQL** [33] - text-to-SQL датасет состоящий из реальных запросов медицинских работников. Он включает в себя 222 шаблона вопросов сотрудников больницы и 6,287 пар примеров. Особенность этого датасета в том, что в тестовой выборке есть вопросы, на которые невозможно ответить без внешних знаний или только на основании базы данных, что требует от модели интерпретации уверенности в предсказании. Пары вопросов, которые соответствуют базе данных, относятся к одной базе – объединению MIMIC-III [34] и eICU [35]. В SQL запросах присутствует от 1 до 5 операций `JOIN`, а сами запросы обогащены операциями, связанными со временем и датой (`strftime` и `datetime`), что специфично реальным вопросам работников больницы.
- **ATIS** [36], **Restaurants** [37], **Scholar** [38], **Restaurants** [37], **Academic** [39], **Yelp и IMDB** [40] – набор небольших датасетов, не превышающих 1000 примеров. Датасеты направлены на работу с одной базой во время обучения и тестирования, а сами SQL-запросы по сложности не превышают пары условий и нескольких операций `JOIN`. В современных моделях text-to-SQL эти датасеты не оцениваются и считаются устаревшими [41, 42, 43].

В данной работе эксперименты проводятся на датасетах SPIDER, WikiSQL и EHRSQL, как наиболее качественных и распространненых для исследования text-to-SQL систем.

1.4 Методы оценки качества text-to-SQL моделей

Оценка text-to-SQL моделей проводится несколькими способами:

– **Exact match** – метрика, используемая для оценки точного построчного совпадения сгенерированного выражения $\tilde{y}$ с истинным выражением y. Поскольку запросы часто могут иметь разный формат, перед сравнением строк применяются методы стандартизации текста:
  – Стандартизация регистра.
  – Унификация непостоянных частей запроса.
  – Форматирование пробелов.

  Эта метрика в большей степени применяется к text-to-SQL в тех случаях, когда иными способами верифицировать корректность запроса невозможно – нет базы данных, для исполнения запроса или сам запрос не является финальным SQL-запросом (например, замаскированы значения или элементы схемы).

– **Execution match** – метрика, оценивающая соответствие исполнения предсказанного запроса и истинного запроса в задаче text-to-SQL. Запрос исполняется на базе данных D, для которой он был сгенерирован. Если результаты совпадают, считается, что сгенерированный запрос корректен. В оценку может быть добавлено условие, чтобы порядок возвращаемых значений был одинаков для обоих запросов, если в запросе присутствует синтаксическая конструкция `ORDER BY`.

$$\hat{y}_i = \begin{cases} 1 & \text{if } \mathrm{M}(g_i) == \mathrm{M}(p_i) \\ 0 & \text{if } \mathrm{M}(g_i) \neq \mathrm{M}(p_i) \end{cases} \tag{1.1}$$

Для обеих метрик, корректность $\hat{y}_i$ для пары истинного запроса g_i и предсказанного запроса p_i определяется формулой 1.1. M соответствует либо построчному сравнению **Exact Match**, либо сравнению результата исполнения **Execution Match**.

1.5 Методы обучения text-to-SQL моделей

В этой главе будут рассмотрены text-to-SQL модели и способы их построения. Все рассмотренные модели относятся к sequence-to-sequence подходу, поскольку на вход подается последовательность и на выходе генерируется последовательность. Это делает задачу text-to-SQL близкой к задаче машинного перевода. В основном используются три подхода к обучению моделей: классический sequence-to-sequence подход языкового моделирования, шаблонный подход и подход, основанный на генерации грамматического дерева языка SQL.

Модели text-to-SQL в постановке sequence-to-sequence обучаются методом минимизации отрицательного логарифма правдоподобия, как и любая другая задача языкового моделирования:

$$L(X, Y, S, \theta) = -\frac{1}{N} \sum_{i=0}^{N} \sum_{k=0}^{K} logP(y_{i,k}|x_i, y_{i,<k}, s_i, \theta) \rightarrow min \tag{1.2}$$

В качестве элементов целевой переменной $y_{i,k}$ могут выступать токены, классы и узлы в грамматическом дереве целевого словаря $\mathcal{V}$. Предсказание следующего токена в последовательности длины K относится к классическим моделям sequence-to-sequence:

$$\tilde{y}_k = \arg\max_{\tilde{y}_k \in \mathcal{V}} P(\tilde{y}_k \mid x, \tilde{y}_{<k}, s, \theta) \tag{1.3}$$

Предсказание класса характерно для шаблонного метода обучения семантического парсера, например, предсказание наличия компоненты `ORDER BY` в запросе. Предсказание следующего узла в грамматическом дереве относится к грамматическому подходу.

– **Подход sequence-to-sequence** в обучении семантических парсеров заключается в обучении модели генерировать $\tilde{y}$ с помощью языкового моделирования. Модель обучается минимизировать логарифм правдоподобия, последовательно предсказывая ожидаемые токены запроса $\tilde{y_k}$ [16, 31, 44]. В качестве архитектуры может выступать как архитектура только декодировщика [45], так и кодировщика-декодировщика [46].

– **Шаблонный подход** заключается в заполнении заданного шаблона команды $\tilde{y}$ на формальном языке его компонентами $\tilde{y_k}$, например, наличие компоненты `ORDER BY` или количество условий в `WHERE`. Шаблон команды делится на множество независимых компонент, для которых генерация каждой компоненты представляет собой отдельную задачу классификации. Однако, ввиду слабой масштабируемости, этот подход используется в современных семантических парсерах все реже [47, 48, 49]. В качестве архитектуры используется кодировщик для формирования векторного представления вопроса и схемы и последующее множество классификаторов для определения элементов шаблона.
– **Грамматический подход** состоит в предсказании дерева грамматики для целевой команды $\tilde{y}$. Процесс обучения аналогичен sequence-to-sequence, но вместо токенов $\tilde{y_1}, \tilde{y_2}, ..., \tilde{y_K}$ предсказывается последовательность контекстно-свободных грамматик, состоящих из терминалов, не-терминалов и продукций. Грамматики обычно реализуются через подход Domain Specific Language (DSL), адаптированный под пространство используемого языка. С примером создания DSL грамматики под датасет SPIDER можно ознакомится в репозитории решения RAT-SQL[5]. В задаче text-to-SQL DSL часто формируется под конкретный датасет, так как разработка полного пространства грамматик языка SQL трудозатратна. Для генерации финальной команды $\tilde{y}$ модель предсказывает грамматическое дерево, используя алгоритм построения дерева в глубину и предсказывая узлы и листья дерева $\tilde{y_k}$. На этапе предсказания модель обходит построенное дерево в глубину и собирает терминальные значения из его листьев [42, 50, 51]. Аналогично шаблонному подходу, архитектура состоит из кодировщика и декодировщика, предсказывающего грамматики.

При обучении text-to-SQL моделей первый этап включает сопоставление сущностей, указанных в вопросе на естественном языке. При работе с реляционной таблицей осуществляется сопоставление элементов схемы и значений базы данных; пример этой задачи представлен на рисунке 1.1. Для выполнения этого может быть разработана дополнительная модель [42] или задача сопоставления ложится на саму модель генерации запроса [16]. Этап

[5]https://github.com/berlino/tensor2struct-public/blob/main/tensor2struct/languages/ast/Spider.asdl

сопоставления сущностей связан с моделью генерации запроса передачей результата в этап генерации итогового запроса.

Далее будут представлены методы обучения text-to-SQL моделей на примере четырех ключевых моделей, которые наиболее популярны на бенчмарке SPIDER[6]. Выбраны именно эти методы для разбора, так как в их реализациях есть индуктивное смещение алгоритма обучения в сторону text-to-SQL задачи.

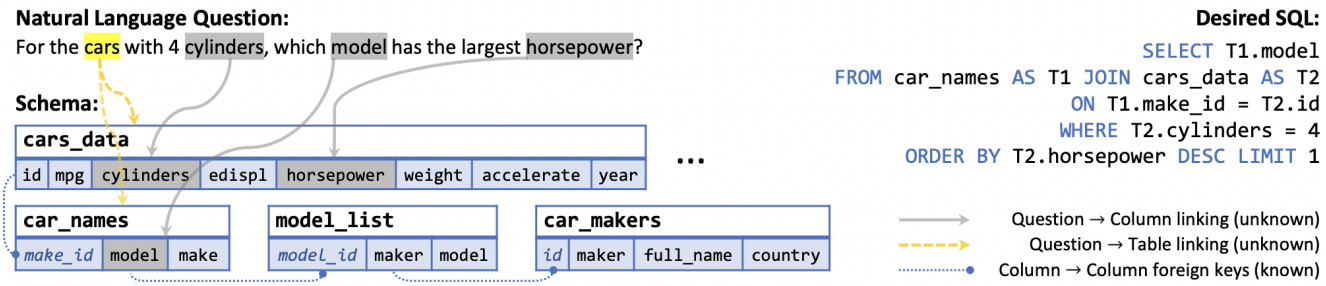

Рисунок 1.1 — Пример задачи сопоставления сущностей в text-to-SQL из датасета Spider. Модель text-to-SQL должна правильно сопоставить слово cars из входного вопроса с названиями таблиц `cars_data` и `car_names` и определить их отношение между друг другом для построения верной операции объединения, аналогичную операцию необходимо провести с атрибутами `cylinders`, `horsepower`. Также необходимо верно определить те элементы, которые нужно возвратить пользователю – `model`.

Наиболее простой вариант обучения заключается в дообучении предобученной языковой модели. Одной из распространенных моделей для дообучения является модель T5 [46]. Архитектура модели T5 представляет собой кодировщик-декодировщик, состоит из блоков трансформера [52] и предназначена для решения задач sequence-to-sequence. Модель дообучается в постановке sequence-to-sequence моделей методом минимизации отрицательного логарифма правдоподобия, как описано в начале секции и представлено в формуле 1.2.

Пример дообучения модели text-to-SQL T5 показан на рисунке 1.2. На вход модели подаются название базы данных D, вопрос QS и линеаризованная схема базы данных S, которая представляет собой список таблиц t_i и соответствующий каждой таблице список атрибутов $c^i_1, ..., c^i_{n_i}$.

[6]https://yale-lily.github.io/spider

$$X = D : QS|t_1 : c_1^1, ...c_{n_1}^1|t_2 : c_1^2, ...c_{n_2}^2|t_N : c_1^N, ...c_{n_N}^N \tag{1.4}$$

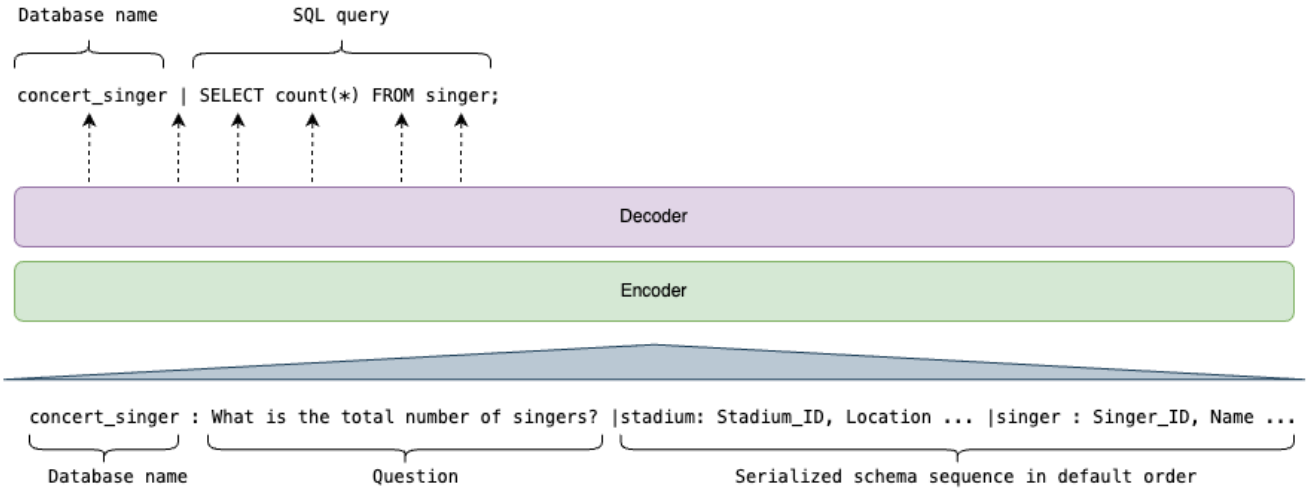

Рисунок 1.2 — Иллюстрация text-to-SQL модели, на основании sequence-to-sequence модели T5. На вход модели архитектуры кодировщик (encoder)-декодировщик (decoder) подается название базы данных D, вопрос QS и линеаризованное представление схемы S, как описано в представлении 1.4. Как в представлении 1.5, модель последовательно генерирует название базы данных и SQL запрос.

На выходе базы генерируется сначала название базы (такой формат помогает лучше генерировать последующие элементы схемы в запросе [16, 53]) и потом сам запрос QR.

$$Y = D|QR \tag{1.5}$$

Существует 3 ключевых размера T5 моделей – T5-base[7] (220 млн. параметров), T5-large (770 млн. параметров)[8], T5-3B (3 млрд. параметров)[9]. Размер модели влияет на размер контекста, который модель может обработать за один раз и на качество генерации.

BRIDGE (Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing) [43] следует форме кодировщик-декодировщик. В качестве кодировщика используется предобученная модель BERT [54], а декодировщик представляет собой нейронную сеть

[7]https://huggingface.co/google-t5/t5-base
[8]https://huggingface.co/google-t5/t5-large
[9]https://huggingface.co/google-t5/t5-3b

LSTM [55] с pointer-generator механикой предсказания следующего токена [56]. Аналогично подходу T5, BRIDGE сериализует схему базы данных в строку, состоящую из элементов таблиц t_i и атрибутов c_{n_i}, и конкатенирует её с входным вопросом QS с элементами-сепараторами – `[T], [C]`. В BRIDGE также присутствует механизм сопоставления значений и элементов входного запроса с помощью fuzzy-алгоритма сопоставления строк (например, алгоритм Левенштейна). Те значения v, которые алгоритм нашел с высоким уровнем сопоставления, добавляются к входной строке через элемент-сепаратор `[V]`. Входное представление формируется сериализацией вопроса и элементов схемы с элементами-сепараторами `[SEP]`, как показано в представлении 1.6. Пример форматирования входа в BRIDGE архитектуру приведен на рисунке 1.3.

$$X = [CLS]QS[SEP][T]t_1[C], c_1^1, ..., c_{n_1}^1, [T], t_2, [C], c_1^2, ..., c_{n_1}^2, [T], t_3 ... [SEP] \quad (1.6)$$

Кодировщик BRIDGE. Представление X кодируется кодировщиком (в решении, BRIDGE используется BERT + LSTM кодировщик), формируя эмбеддинги $h_X \in \mathbb{R}^{|X| \times n}$, где $|X|$ – количество токенов в строке X, n – размерность эмбеддинга кодировщика. Далее сегмент, относящийся к вопросу h_X, передается в следующий слой bi-LSTM, формируя представление $h_{QS} \in \mathbb{R}^{|QS| \times n}$, где $|QS|$ — токены, соответствующие именно строке вопроса в последовательности x. Скрытое представление схемы $\mathbf{h}_S \in \mathbb{R}^{|\mathcal{S}| \times n}$ формируется из скрытого представления таблиц $\mathbf{h}_S^{t_i}$ и колонок $\mathbf{h}_S^{c_{ij}}$, причем каждая колонка в том числе обогащена информацией о типе колонки $\mathbf{f}_{\text{type}}$, primary key статусе $\mathbf{f}_{\text{pri}}$ и foreign key статусе $\mathbf{f}_{\text{for}}$ с помощью полносвязной сети g с функцией активации $ReLU$, как показано в формуле 1.7.

$$\begin{aligned} \mathbf{h}_S^{t_i} &= g([\mathbf{h}_X^p; \mathbf{0}; \mathbf{0}; \mathbf{0}]), \\ \mathbf{h}_S^{c_{ij}} &= g([\mathbf{h}_X^q; \mathbf{f}_{\text{pri}}^u; \mathbf{f}_{\text{for}}^v; \mathbf{f}_{\text{type}}^w]) \\ &= \text{ReLU}(\mathbf{W}_g[\mathbf{h}_X^m; \mathbf{f}_{\text{pri}}^u; \mathbf{f}_{\text{for}}^v; \mathbf{f}_{\text{type}}^w] + \mathbf{b}_g), \\ \mathbf{h}_S &= [\mathbf{h}^{t_1}, \ldots, \mathbf{h}^{t_\mathcal{T}}, \mathbf{h}^{c_1^1}, \ldots, \mathbf{h}^{c_{n_N}^N}] \in \mathbb{R}^{|\mathcal{S}| \times n}, \end{aligned} \quad (1.7)$$

Декодировщик BRIDGE. В качестве декодировщика используется LSTM с pointer-generator механикой предсказания следующего токена с multi-head attention механизмом для обогащения эмбеддингов информацией контекста

[52]. Декодировщик инициализируется с финального состояния кодировщика вопроса. На каждом шаге декодировщик выполняет одно из следующих действий: генерирует токен из словаря $\mathcal{V}$, копирует токен из вопроса Q или копирует элемент схемы из S.

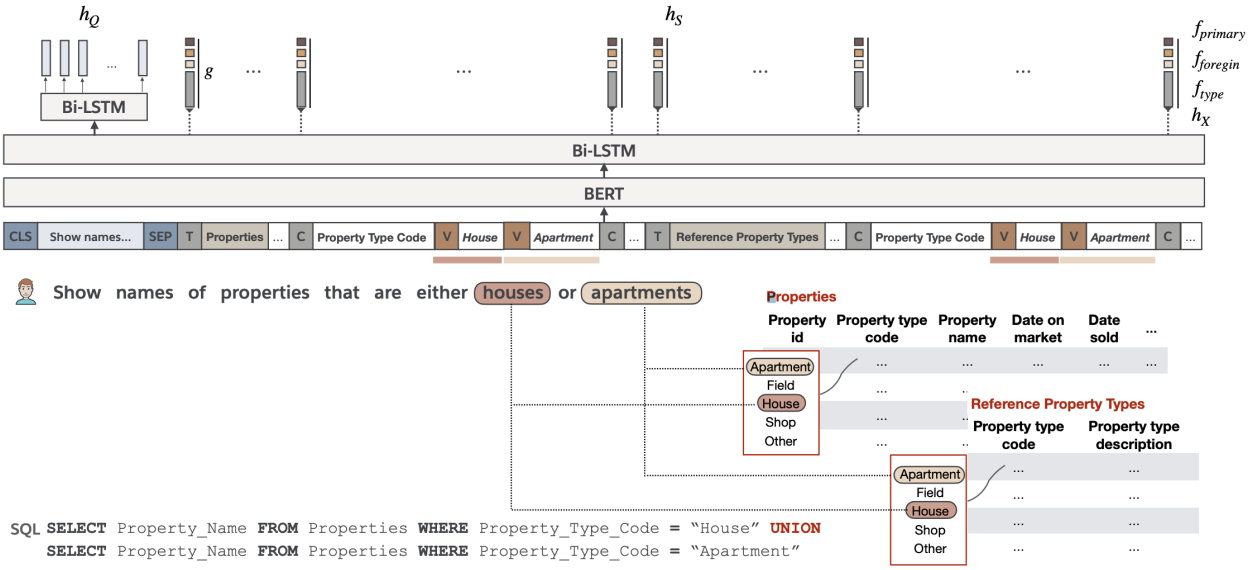

Рисунок 1.3 — Архитектура кодировщика части BRIDGE. Две сущности **house** и **apartments** во входном вопросе сопоставляются со значениями базы данных и добавляются через элемент-сепаратор к входной последовательности.

На каждом шаге t при текущем состоянии декодировщика $\mathbf{s}_t \in \mathbb{R}^{1\times n}$ и рассчитанном представлении кодировщика для каждого токена QS и S входной последовательности $h_j = [\mathbf{h}_{QS}; \mathbf{h}_S] \in \mathbb{R}^{(|Q|+|S|)\times n}$, вычисляется операция multi-head attention [52].

Multi-head attention Цель multi-head attention – обогатить векторное представление токена векторными представлениями токенов из контекста. Механизм multi-head attention состоит в создании H независимых подпространств, в каждом из которых вычисляется операция внимания. Каждому подпространству h, соответствует набор обучаемых параметров $\mathbf{W}_Q^{(h)}, \mathbf{W}_K^{(h)}, \mathbf{W}_V^{(h)} \in \mathbb{R}^{n\times d}$ для формирования промежуточных векторных представлений в подпространстве d механизма внимания. Обычно, $d << n$. Операция внимания перевзвешивает данный эмбеддинг текущего шага t ($\mathbf{s}_t$) на основании остальных эмбеддингов контекста h_j в подпространстве размера d.

Для модели BRIDGE, метод обогащения векторных представлений представлен в формуле 1.8. В данной версии multi-head attention, $\mathbf{W}_K^{(h)} = \mathbf{W}_V^{(h)}$. Сначала вычисляются коэффициенты влияния $\boldsymbol{\alpha}_{tj}^{(h)}$ каждого эмбеддинга h_j контекста длинной $|Q| + |S|$ на текущее представление (эмбеддинг) декодера s_t на шаге t. Операция softmax используется для нормализации коэффициентов, так что $\sum_{j=1}^{|Q|+|S|} \boldsymbol{\alpha}_{tj}^{(h)} = 1$. Итоговое обогащённое представление $\mathbf{z}_t^{(h)} \in \mathbb{R}^{1\times d}$ является суммой взвешенных представлений контекста в подпространстве d с коэффициентом $\boldsymbol{\alpha}_{tj}^{(h)}$. Представления $\mathbf{z}_t^{(h)}$ вычисляются для каждого $h \in H$. Рассчитанные представления конкатенируются в один вектор $z_t \in \mathbb{R}^{1\times Hd}$ – обогащенное контекстом векторное представление.

$$
\begin{aligned}
e_{tj}^{(h)} &= \frac{\mathbf{s}_t \mathbf{W}_K^{(h)} (\mathbf{h}_j \mathbf{W}_V^{(h)})^\top}{\sqrt{n/H}}; \quad \boldsymbol{\alpha}_{tj}^{(h)} = \mathrm{softmax}_j\{e_{tj}^{(h)}\}, \\
\mathbf{z}_t^{(h)} &= \sum_{j=1}^{|Q|+|S|} \boldsymbol{\alpha}_{tj}^{(h)} (\mathbf{h}_j \mathbf{W}_V^{(h)}); \quad \mathbf{z}_t = [\mathbf{z}_t^{(1)}, \ldots, \mathbf{z}_t^{(H)}],
\end{aligned}
\tag{1.8}
$$

Итоговое распределение для предсказания следующего токена $\tilde{y}_t$ определяется как сумма распределения по словарю $P_{\mathcal{V}}(\tilde{y}_t)$ (из формулы 1.10) и по элементам входной последовательности $|Q| + |S|$, которая состоит только из слов вопроса и элементов схемы таблиц $\mathcal{T}$ и атрибутов $\mathcal{C}$. Таким образом, у модели есть возможность либо предсказать из целевого словаря $\mathcal{V}$ или из входной последовательности x. Это полезно, когда среди токенов есть out-of-vocabulary (OOV) токены или токены, которые редко встречаются в обучающей выборке и модель не склонна выбирать их. Если один и тот же токен есть и в V, и в x – у него будет более высокая вероятность в итоговом распределении. В формуле 1.9, параметр p_{gen}^t на каждом шаге t рассчитывается на основании s_t и z_t из компоненты кодировщика с multi-head attention механизмом. $W_{gen}^s \in \mathbb{R}^{n\times 1}, W_{gen}^z \in \mathbb{R}^{d\times 1}, b_{gen} \in \mathbb{R}$ – обучаемые параметры, определяющие вес семплирования совместного распределения словаря V и входной последовательности. Для построения распределения по входной последовательности, используется $a_{tj}^{(H)}$ – распределение по контекстным токенам подпространства под номером H (также возможно брать среднее по всем подпространствам). Семплирование для предсказания следующего токена производится с помощью моды распределения, как в Формуле 1.3.

$$p^t_{\text{gen}} = \text{sigmoid}(\mathbf{s}_t \mathbf{W}^s_{\text{gen}} + \mathbf{z}_t \mathbf{W}^z_{\text{gen}} + \mathbf{b}_{\text{gen}}); \tag{1.9}$$

$$p^t_{\text{out}} = p^t_{\text{gen}} P_{\mathcal{V}}(\tilde{y}_t) + (1 - p^t_{\text{gen}}) \boldsymbol{\alpha}^{(H)}_{tj}. \tag{1.10}$$

RAT-SQL (Relation-Aware Transformer SQL) [42] – архитектура данной модели следует также sequence-to-sequence подходу, что и T5, BRIDGE модели с некоторыми интересными аспектами.

Формирование входной и выходной последовательности. Схема базы данных D представляется как связанный граф $\mathcal{G} = \langle \mathcal{T} \cup \mathcal{C} \cup QS, \mathcal{E} \rangle$. Узлами x_i данного графа являются названия таблиц $\mathcal{T}$, атрибутов $\mathcal{C}$ схемы базы и слова входного запроса QS, ребрами $\mathcal{E}$ – primary/foreign SQL отношения и факт высокого уровня сопоставления слов входного вопроса QS и элементов схемы $\mathcal{T}$ и $\mathcal{C}$. Выходная последовательность представлена как абстрактное синтаксическое дерево контекстно-свободных грамматик языка SQL.

Кодировщик В качестве кодировщика для формирования эмбеддинга для выражения используется последовательность блоков трансформера с модифицированным механизмом multi-head attention из формулы 1.8. В решении RAT-SQL multi-head attention модифицируется введением дополнительной переменной r_{tj} для учета информации о связях графа $\mathcal{G}$ между текущим токеном x_t и остальными токенами последовательности x_j длины L в формуле 1.11 [57]. r_{tj} – соответствует связи между элементами схемы (например, отношение таблицы и атрибута через primary/foreign key отношение).

$$\begin{aligned} e^{(h)}_{tj} &= \frac{\mathbf{x}_t \mathbf{W}^{(h)}_Q (\mathbf{x}_j \mathbf{W}^{(h)}_K + r^K_{tj})^\top}{\sqrt{n/H}}; \quad \boldsymbol{\alpha}^{(h)}_{tj} = \text{softmax}_j\{e^{(h)}_{tj}\}, \\ \mathbf{z}^{(h)}_t &= \sum_{j=1}^{L} \boldsymbol{\alpha}^{(h)}_{tj} (\mathbf{h}_j \mathbf{W}^{(h)}_V + r^V_{tj}); \quad \mathbf{z}_t = [\mathbf{z}^{(1)}_t, \ldots, \mathbf{z}^{(H)}_t], \end{aligned} \tag{1.11}$$

Декодировщик. Декодировщик в решении RAT-SQL следует древовидному декодировщику из работы [58]. Абстрактное синтаксическое дерево генерируется декодером методом обхода в глубину, пример на рисунке 1.4. Декодер предсказывает одно из двух действий a: (i) трансформировать

последний предсказанный узел дерева глубже в грамматическое правило или (ii) при предсказании листа дерева выбрать нужную таблицу или атрибут.

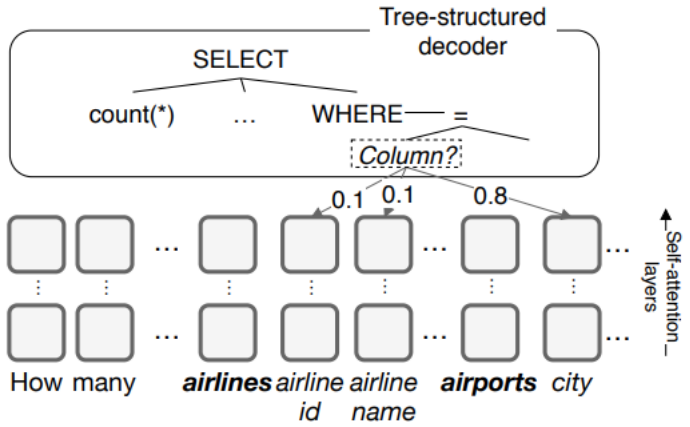

Рисунок 1.4 — Предсказание грамматического дерева в архитектуре RAT-SQL. Последовательность кодируется блоками трансформера. Декодировщик генерирует итоговую последовательность в виде бинарного дерева.

Задача генерации грамматического дерева p схожа с задачей генерации последовательности – $P(p|\mathcal{Y}) = \prod_t (a_t|a_{<t}, \mathcal{Y})$, где $\mathcal{Y} \in \mathbb{R}^{1\times h}$ — эмбеддинг графа вопроса и схемы базы данных, а $a_{<t}$ — предыдущие решения декодера. Данная последовательность в решении RAT-SQL моделируется с помощью модели LSTM, параметры моделирования описаны в формуле 1.12.

$$
\begin{aligned}
&m_t, h_t = \text{LSTM}\left([a_{t-1}, z_t, h_{p_t}, a_t, a_{f_t}], m_{t-1}, h_{t-1}\right) \\
&\quad \text{где} \\
&m_t \text{ - long-term состояние LSTM} \\
&h_t \text{ - short-term состояние LSTM} \\
&a_{p_{t-1}} \text{ - векторное представление предыдущего узла дерева} \\
&p_t \text{ - индекс узла AST дерева} \\
&\quad \text{относительного которого происходит расширение дерева} \\
&a_{p_t} \text{ - векторное представление текущего узла дерева} \\
&z_t \text{ - векторное представление контекста,} \\
&\quad \text{подсчитанное по } \mathcal{Y} \text{ относительно состояний } h_{t-1} \\
&\quad \text{механизмом multi-attention}
\end{aligned}
\tag{1.12}
$$

В качестве действий a на основании h_t в RAT-SQL есть два правила: ApplyRule – предсказание следующей контекстно-свободной грамматики для

продолжения генерации дерева, SELECTCOLUMN – предсказание атрибута в листе дерева, либо SELECTTABLE – предсказание таблицы в листе дерева. Детали реализации представлены в работе [58].

RESDSQL (Ranking-enhanced Encoding plus a Skeleton-aware Decoding framework for Text-to-SQL) [44] является SoTA дообученным решением для моделей архитектуры кодировщик-декодировщик на бенчмарке SPIDER[10]. Архитектура RESDSQL состоит из двух компонент – определения релевантных элементов схемы с помощью классификатора и генерации запроса.

Ранжировщик элементов схемы. Модель классификатора включает предобученный кодировщик и двунаправленную LSTM нейронную сеть. На вход модели подается последовательность, как показано в формуле 1.4. Для каждого токена t_i формируется эмбеддинг e_i. Поскольку каждое название схемы разбивается на несколько частей BPE токенизатором [59] (например, название колонки “direction” превратится в два токена – “direct” и “ion”), используется bi-LSTM для операции pooling, которая кодирует токенизированное представление элементов схемы в один эмбеддинг для каждого элемента – в результате такого кодирования получается эмбеддинг для каждой таблицы $t_i \in \mathbb{R}^{1\times n}$ и эмбеддинг для каждого атрибута $c_{n_i} \in \mathbb{R}^{1\times n}$, где n – размерность эмбеддинга скрытого представления кодировщика. Затем эмбеддинг названия таблицы обогащается векторными представлениями атрибутов с помощью механизма multi-head attention в эмбеддинг $\hat{t}_i \in \mathbb{R}^{1\times d}$, аналогично тому, как описано в формуле 1.8.

Предсказание для каждого элемента схемы формируется с помощью двух полносвязных нейронных сетей с нелинейностью softmax σ, как показано в формуле 1.13. Обучаемые параметры нейронной сети – $U_1^t, U_1^c \in \mathbb{R}^{d\times w}$, $b_1^t, b_1^c \in \mathbb{R}^w$, $U_2^t, U_2^c \in \mathbb{R}^{d\times 2}$, и $b_2^t, b_2^c \in \mathbb{R}^2$. Предсказания $\hat{y}_i, \hat{y}_i^k \in \mathbb{R}^2$ представляют собой распределения по двум классам – данный элемент схемы присутствует или отсутствует в итоговом запросе. Результат операции определяется с помощью $argmax$ операции.

$$\begin{aligned} \hat{y}_i &= \sigma((t_i U_1^t + b_1^t)U_2^t + b_2^t); \\ \hat{y}_i^k &= \sigma((c_{n_i}^i U_1^c + b_1^c)U_2^c + b_2^c) \end{aligned} \tag{1.13}$$

[10]https://yale-lily.github.io/spider

Focal loss. Поскольку SQL-запрос обычно касается лишь нескольких таблиц и колонок в базе данных, распределение целевых переменных y_i и y_i^k в обучающем наборе данных сильно несбалансированно (т.е. в основном элемент схемы отсутствует в целевой переменной). В результате количество отрицательных примеров многократно превосходит количество положительных примеров, что вызывает серьезное смещение в процессе обучения. В связи с этим, в качестве функции потерь классификатора используется focal loss [60], представленная в формуле 1.16, вместо кросс-энтропийной функции потерь.

Пусть p_t – вероятность положительного класса. CE – кросс-энтропийная ошибка для бинарной классификации.

$$\mathrm{CE}(p, y) = \begin{cases} -\log(p) \text{если } y = 1 \\ -\log(1-p) \text{иначе} \end{cases} \tag{1.14}$$

Далее, пусть p_t:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{иначе} \end{cases} \tag{1.15}$$

Тогда $\mathrm{CE}(p, y) = \mathrm{CE}(p_t) = -\log p_t$. Итоговая формула focal loss представляет собой перевзвешивание кросс-энтропийной функции ошибки с коэффициентом фокусировки $\boldsymbol{\gamma}$. Этот параметр определяет степень, с которой фокусировка будет осуществляться на неверно предсказанных примерах, с высокой степенью уверенности. Для примера, если $\boldsymbol{\gamma} = 0$, то ошибка эквивалентна кросс-энтропийной функции ошибки CE. Если $\boldsymbol{\gamma} = 2$, то верно предсказанный пример с вероятностью $p_t = 0.9$, снижает значение focal-loss ошибки в 100 раз, а если $p_t = 0.968$ – в 1000 раз. В то время, как примеры, в которых модель ошибочно не уверена, снизят ошибку всего лишь в 4 раза (например, если $p_t \leqslant 0.5$ и $\boldsymbol{\gamma} = 2$) – таким образом, дав больший вес ошибки сложным примерам.

$$FL(p_t) = -(1-p_t)^{\boldsymbol{\gamma}} \log(p_t) \tag{1.16}$$

Финальная функция потерь модели рассчитывает ошибку предсказания релевантного запросу атрибута и таблицы по следующей формуле, где N – количество таблиц базы данных, а $M = \sum_{i=1}^{N} n_i$ – количество атрибутов базы данных:

$$L(y_i, \hat{y}_i, y_k^i, \hat{y_k^i}) = \frac{1}{N}\sum_{i=1}^{N} FL(y_i, \hat{y}_i) + \frac{1}{M}\sum_{i=1}^{N}\sum_{k=1}^{n_i} FL(y_k^i, \hat{y}_k^i) \tag{1.17}$$

FL – focal loss, y_i - значение целевой переменной для таблицы i. $y_i = 1$, если таблица упомянута в SQL запросе, иначе $y_i = 0$. y_k^i - значение целевой переменной атрибута k в таблице i. $y_k^i = 1$, если атрибут упомянут в SQL запросе, иначе $y_k^i = 0$.

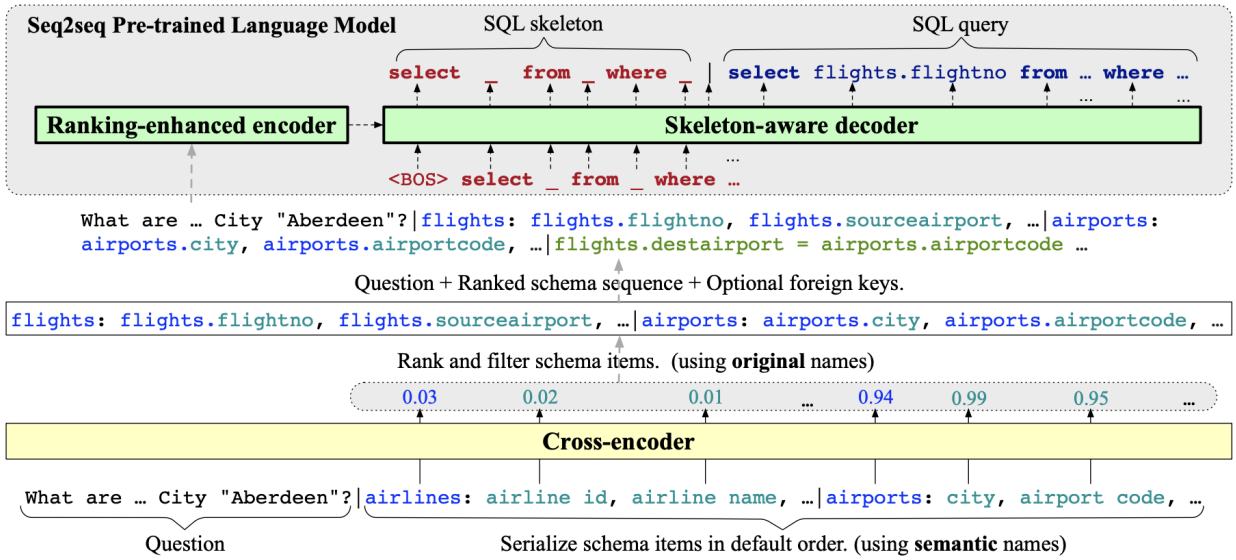

Рисунок 1.5 — Иллюстрация семантического парсера RESDSQL решения. Cross-encoder обогащает векторные представления элементов схемы семантикой вопроса и другими элементами схемы и ранжирует элементы схемы базы данных. На вход text-to-SQL модели подается исходный вопрос, наиболее вероятные названия таблиц и атрибутов, присутствующих в запросе, и информация о primary/foreign ключах. text-to-SQL модель сначала генерирует шаблон вопроса (SQL Skeleton) и далее сам целевой вопрос (SQL query).

Декодировщик RESDSQL. Результаты предсказания для элементов схемы ранжируются по вероятности. Затем выбираются k_1 наиболее подходящих таблиц и k_2 колонок для данного вопроса и передаются на вход модели генерации T5, которая описана в sequence-to-sequence подходе 1.5. На этапе генерации модель сначала учится сгенерировать шаблон SQL запроса (SQL skeleton), с пропущенными элементами схемы и значениями, а после и итоговый

SQL запрос, как показано на рисунке 1.5. Таким образом, в начале генерируется не очень сложная задача предсказания синтаксической структуры запроса, а потом, при наличии более богатого контекста, генерируется истинный SQL запрос с заполненными элементами схемы.

```
### Complete sqlite SQL query only and with no
↳ explanation
### SQLite SQL tables, with their properties:
#
# continents(ContId, Continent)
# countries(CountryId, CountryName, Continent)
#
### How many continents are there?
SELECT
```

Рисунок 1.6 — Метод организации контекста для большой языковой модели. На вход модели подаются инструкция по задаче, схема реляционной базы, инструкция и сама задача на генерацию.

```
/* Some example questions and corresponding SQL queries
↳ are provided based on similar problems: */
/* Answer the following: How many authors are there? */
SELECT count(*) FROM authors

/* Answer the following: How many farms are there?. */
SELECT count(*) FROM farm

${TARGET_QUESTION}
```

Рисунок 1.7 — Метод организации контекста для большой языковой модели в формате in-context-learning для решения DAIL-SQL. На вход модели подается инструкция по задаче - генерация SQL запроса при данном вопросе и пары схожих пар вопрос-SQL запрос. Схожие пары определяются по мере близости (например, по косинусному расстоянию между эмбеддингами вопросов) данного вопроса к вопросам, для которых уже известен SQL (из обучающей выборки, например).

С развитием больших языковых моделей стало актуально в том числе контекстное обучение (in-context learning (ICL)) – метод обучения без обновления параметров нейронных сетей, а с помощью подачи на вход инструкций и схожих примеров с результатом обучения, как контекст языковой модели.

$$\tilde{y_k} = \arg\max_{\tilde{y_k} \in \mathcal{V}} P(\tilde{y} \mid I, s, \text{demo}, \tilde{y}_{<k}, QS, \theta) \tag{1.18}$$

Организация контекста для большой языковой модели (например, OpenAI ChatGPT) представлена на рисунках 1.6 и 1.7 для решения DAIL-SQL [61]. На вход модели подаются инструкция по задаче I, схема реляционной базы s, примеры истинных пар $demo = [(QS_1, QR_1), (QS_2, QR_2), ...]$ и сам вопрос QS к базе данных D. Задача few-shot in-context-learning по предсказанию итогового запроса $\tilde{y}$ для языковой модели LLM P, параметризованной θ, представлена в 1.18.

1.6 Русский text-to-SQL датасет PAUQ

PAUQ [19] является локализованным на русский язык и улучшенным вариантом английского датасета text-to-SQL Spider. В рамках работы были переведены на русский язык вопросы x, запросы y, а также обновлён контент баз данных D. Кроме того, был исправлен исходный датасет SPIDER на английском языке. С датасетом можно ознакомиться в репозитории[11].

1.6.1 Принципы адаптации и разметка text-to-SQL датасета

Для адаптации датасета text-to-SQL Spider с учётом существующих адаптаций на другие языки (китайский, португальский) [62, 63], необходимо разработать принципы локализации исходных английских запросов с учётом русского языка, культуры и общепринятых правил взаимодействия с базой

[11] https://github.com/ai-spiderweb/pauq

данных. Также требуется изменить контент базы знаний для корректной оценки точности исполнения запросов, сгенерированных моделью.

Принципы локализации англоязычного датасета Spider включают следующее:

- Русские значения в запросах не должны быть прямыми переводами английских значений. Значения в SQL должны быть сформулированы либо на английском, либо на русском языке, или возможно, быть комбинацией обоих языков.
- Если в SQL-запросе присутствуют только названия таблиц/атрибутов и отсутствуют значения, такой запрос и соответствующий текстовый вопрос не подлежат локализации или модификации — они сразу переводятся. В случае, если SQL-запрос содержит значения, этот запрос, а также соответствующий текстовый вопрос изменяются на основе нескольких критериев. Прежде всего, русские значения не должны быть буквальными переводами английских, а должны быть близкими аналогами, отсутствующими в исходном содержании базы данных. Принципы локализации значений:
  - **Аббревиатуры:** если существует широко используемый аналог на русском языке аббревиатуры, то она переводится и изменяется. Если найдена неизвестная аббревиатура, она сохраняется в оригинальной форме, без транслитерации.
  - **Названия фильмов/песен/метрик/событий:** такие названия локализуются, то есть заменяются аналогом, известным в русской культуре (который знаком носителю языка из средств массовой информации/культуры). Однако название должно оставаться на английском языке, если оно используется обычно в оригинальной форме.
  - **Собственные имена:** личные собственные имена заменяются аналогами на русском языке. Названия компаний и брендов не переводятся.
  - **Адреса:** локализуются.
  - **Бинарные значения (например, “да/нет”, “истина/ложь”):** такие значения должны быть переведены на русский язык.

– Перевод с английского на русский выполняется профессиональным переводчиком. По сравнению с машинным переводом данный подход дает значительный прирост качества.
– Для баз данных реального мира в России, используются английские названия таблиц и атрибутов, хотя содержание атрибутов может быть на русском языке. По этой причине все названия таблиц и атрибутов в SQL запросах и схемах баз данных остаются на английском языке.
– Все локализованные/изменённые значения добавлены в таблицы базы данных. Без этого честная оценка моделей с точностью выполнения была бы невозможна, так как для вопросов, содержащих отсутствующие значения, фактический результат и предсказанные запросы — как неправильные, так и правильные — возвращали бы одинаковый результат при выполнении.

Соответственно задача разметки выглядит так:

1. Сравнение SQL-запросов из англоязычного датасета и переведённого на русский:
    а) В случае, если запросы идентичны, в них не фигурируют значения полей или значения полей совершенно различны — пропускается дальнейшее действие;
    б) Если в запросах присутствуют значения полей и в русской версии запроса значение является переводом английского значения, необходимо обратиться к соответствующей базе данных и заменить значение из русскоязычного запроса на подходящее по смыслу уникальное значение;
    в) Если это невозможно, следует заменить соответствующее английское значение на русскоязычное во всей базе данных.
2. Были обновлены исходные базы данных SPIDER:
    а) Все пустые таблицы SPIDER баз данных заполнены значениями;
    б) Все атрибуты, упомянутые в SQL запросах и которые не содержат значений, были заполнены.

Исправленные запросы исполняются на соответствующей базе данных и проверяется результат. Если причиной пустого результата является невыполнимый набор условий – такой запрос корректен. Иначе, перепроверяется контент базы данных и добавляются нужные значения в

базу данных, для не пустого исполнения запроса. Перевод на естественном языке каждого вопроса и запросы со значениями перепроверяются вторично независимым разметчиком на соответствие правилам русского языка.

1.6.2 Статистика датасета

В таблице 2 приведены ключевые статистики PAUQ, локализованного и улучшенного датасета text-to-SQL, при оригинальном SPIDER разбиении на обучение и тестирование. В результате адаптации было переведено на русский язык и адаптировано 9876 примеров text-to-SQL. Для адаптации контента баз данных были проведены следующие изменения:

– Английские вопросы были переведены на русский язык;
– В случае наличия значений в запросах они были адаптированы на русский язык;
– Соответствующие русские значения были добавлены в базы данных.

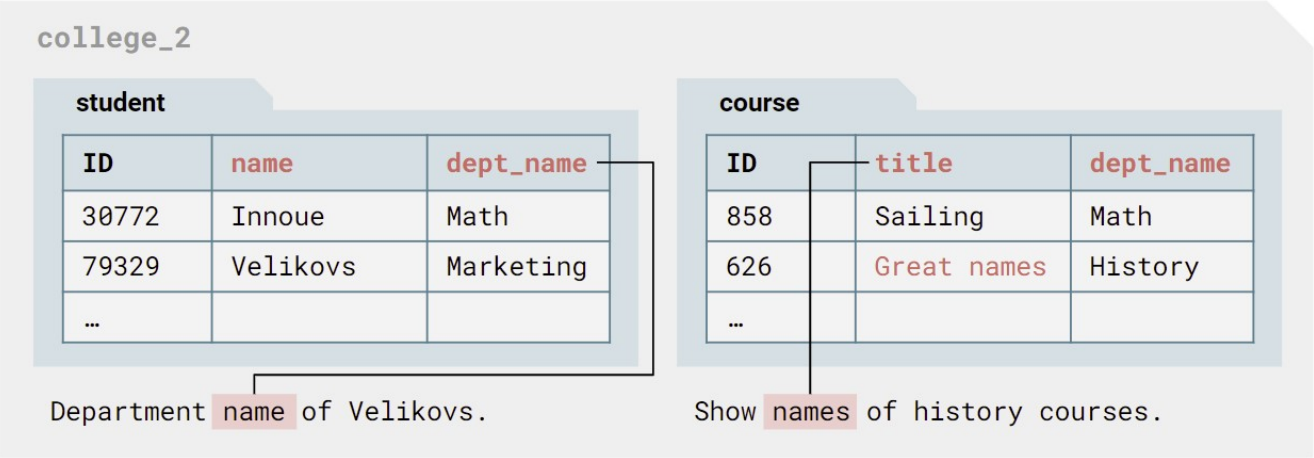

Рисунок 1.8 — Пример базы данных, содержащей разные сущности с одинаковыми именами. Сложность заключается в определении соответствующей таблицы в данном контексте. Например, в таблице `student` слово **name** в вопросе соответствует названию атрибута `dept_name`, а не атрибуту `name` - с которым текстовое сравнение ближе. В таблице `course`, слово **names** соответствует названию `title`.

Таблица 1 — Пересечение элементов по токенам внутри баз данных.

	БД Spider	**БД PAUQ**	**Запросы Spider**	**Запросы PAUQ**
Количество 2ух и более пересечений элементов найденных в:				
...сущностях из соот. базы данных	7261	8637↑	1538 (21.2%)	1952 (22.6%)
...названиях таблиц из соот. базы данных	227	227	42 (18.5%)	101 (44.5%) ↑
...названиях атрибутов из соот. базы данных DB	1425	1425	156 (10.9%)	215 (15.1%) ↑
...значениях из соот. базы данных	5659	7070↑	154 (2.7%)	234 (3.3%) ↑

Сопоставление частей вопроса и сущностей базы данных является основной и самой сложной частью систем перевода текста в SQL [64, 65, 66]. Таким образом, вопросы, содержащие токены, которые присутствуют в нескольких сущностях одновременно, особенно сложны. Например, такой случай приведен на рисунке 1.8 – чем больше таких пересечений в одной базе, тем она сложнее.

Почти каждая четвёртая сущность SPIDER (имя таблицы, имя атрибута, сущность (город Нью-Йорк, команда Манчестер-Юнайтед) или значение атрибута (бинарное значение (М\Ж\1\0), дата, число) содержит токен из какого-то другого элемента, но только восьмая часть из них используется в каком-либо запросе. Добавление русских значений и исправление текущих примеров привело к увеличению пересечений между вопросом и контентом базы данных и внутреннему пересечению элементов внутри базы. Количество пересекающихся разных элементов базы данных между SPIDER и PAUQ датасетами приведено в таблице 1. Количество сущностей в сравнении в SPIDER датасетом в целом представлено на рисунке 1.9.

По сравнению с оригинальным датасетом SPIDER, PAUQ содержит следующие ключевые улучшения:

– Пустых возвратов стало в 8 раз меньше (1665 → 231);

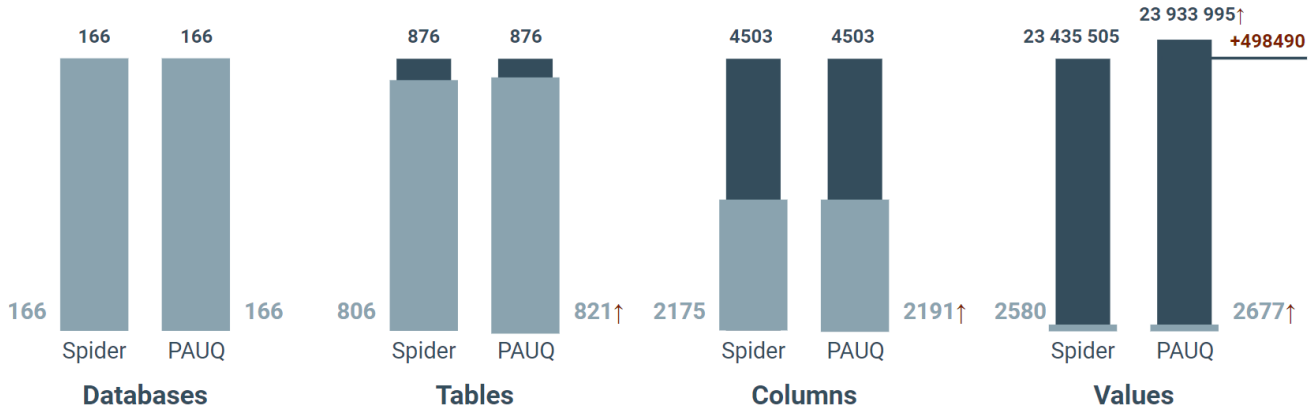

Рисунок 1.9 — Покрытие сущностей баз данных в вопросах. Левый столбец соответствует SPIDER, правый — PAUQ. Тёмный цвет соответствует сущностям из баз данных, светлый — сущностям, использованным в запросах.

Таблица 2 — Ключевые статистики PAUQ, локализованного и улучшенного датасета text-to-SQL на обучении (Train) и тестировании (Test) на русском и английском языках. Длина вопросов и запросов в словах и элементах синтаксиса.

	Train		**Test**	
	Русский	Английский	Русский	Английский
Длина вопросов	12.0	8.9	12.3	9.0
Длина запросов	26.3	26.3	22.4	22.5
Количество таблиц на запрос	2		2	
Количество атрибутов на запрос	3		3	
Количество пар примеров	8800		1076	

– Нулевых возвратов агрегационных запросов стало в 4 раза меньше (379 → 85);
– Синтаксические ошибки, присутствовавшие в оригинальном датасете Spider, были устранены (3 → 0);
– Увеличено количество пересечений между сущностями для усложнения задачи сопоставления элементов схемы с исходным вопросом;
– Как видно из рисунка 1.9, количество сущностей, использованных в вопросах выросло на 4%, а количество значений баз данных на 2%

(это приближает бенчмарк PAUQ к условиям реального использования text-to-SQL систем [30]).

Из таблицы 2 видно, что, несмотря на относительно короткие вопросы на естественном языке, SQL-запросы являются длинными. С точки зрения сложности баз данных бенчмарка SPIDER — в среднем на каждую базу данных приходится 5 таблиц, а в каждой таблице в среднем 4 колонки.

1.6.3 Эксперименты

Для датасета PAUQ были адаптированы несколько английских text-to-SQL моделей. Цель экспериментов по адаптации на русский язык состояла в том, чтобы ответить на три исследовательских вопроса:

1. **RQ-1:** Каковы отличия в результатах обучения между русскими, английскими и мультиязычными моделями?
2. **RQ-2:** Насколько сильно квалифицированный перевод влияет на результат обучения моделей в сравнении с машинным переводом?
3. **RQ-3:** В каких компонентах сгенерированного SQL-запроса модели чаще всего ошибаются?

Таблица 3 — Метрики Execution Match BRIDGE и RAT-SQL на PAUQ. EN - английская версия датасета PAUQ, RU - русская версия. MT RU - машинный перевод на русский язык. HT RU - ручной перевод на русский язык.

Train	Test	BRIDGE	RAT-SQL
EN	EN	0.60	0.63
MT RU	RU	0.40	0.37
MT + HT RU	RU	0.49	0.50
RU	RU	0.48	0.49
RU + EN	EN	0.65	0.65
RU + EN	RU	0.50	0.53

Эксперименты были проведены на двух моделях text-to-SQL из бенчмарка SPIDER. Модели были выбраны с учётом следующих правил:

– Для разносторонней оценки были выбраны модели из различных семейств обучения из главы 1.5;
– Поскольку в PAUQ также изменяются значения базы данных, по крайней мере одна из моделей была выбрана с учётом работы со значениями для оценки эффекта;
– Для исключения предвзятости к процессу обучения обеих моделей использовался один вид кодировщика.

С учетом этих требований были выбраны две модели — BRIDGE и RAT-SQL. Параметры обучения этих моделей представлены в Приложении А. Ключевой метрикой оценки экспериментов была Execution Match метрика. Для экспериментов была использована официальная реализация для оценки решений бенчмарка SPIDER[12]. Каждый эксперимент проводился от 3 до 5 раз. Отклонения в результатах не превышают 1% по метрике Execution Match.

В таблице 3 представлены результаты обучения на различных версиях PAUQ: русская версия PAUQ (**RU** - финальная, полностью переведенная разметчиками), английская версия PAUQ (**EN** - исправленный и обновленный датасет SPIDER), обучение на переведенных данных SPIDER полностью с помощью машинного перевода (machine translated) (**MT RU**), обучение на вручную (human translated) переведенных примерах PAUQ со значениями и переведённых примерах без значений с помощью машинного перевода (**MT + HT RU**). Для оценки влияния многоязычного обучения были объединены русская и английская версия PAUQ (**RU+EN**) и оценены на русской версии PAUQ (**RU**) и английской версии PAUQ отдельно (**EN**).

Для обучения на русскоязычном датасете и на двух языках одновременно был использован кодировщик mBERT-base[13]. Этот кодировщик был выбран, так как он сопоставим по количеству параметров с оригинальным BERT-base[14], использованным в представленных оригинальных моделях. Также, поскольку схема баз данных не переводилась на английский язык, требовалась поддержка многоязычности. Обучение английских моделей проводилось на архитектуре BERT-base.

RQ-1: Вывод об отличиях в результатах моделей, обученных на разных языках Как видно в таблице 3, модели обученные на английском

[12]https://github.com/taoyds/test-suite-sql-eval
[13]https://huggingface.co/google-bert/bert-base-multilingual-uncased
[14]https://huggingface.co/google-bert/bert-base-uncased

языке с использованием кодировщика BERT-base, демонстрируют лучшие метрики по сравнению с обучением на русском языке с использованием mBERT-base. В то же время обучение на двух языках одновременно значительно улучшает метрики на тестовых наборах данных русского и английского языка.

RQ-2: Вывод о влиянии квалифицированного перевода на решение задачи text-to-SQL. В таблице 3 видно, что MT RU против профессионального перевода ухудшается на 8-13%. В качестве инструмента машинного перевода был использован Yandex Translate [15]. Анализ ошибок показывал, что основные источники ошибок — примеры с использованием значений не соответствуют значениям в золотых запросах на SQL. Если примеры со значениями в запросе перевести с помощью профессионального перевода, а без значений оставить в формате машинного – качество моделей будет сопоставимо с полным ручным переводом. К примеру, в Spider 40% вопросов и SQL запросов не имеют значений. Значит с помощью машинного перевода можно сэкономить время и деньги на разметку, не жертвуя при этом качеством.

1.6.4 Анализ ошибок

В третьем вопросе этой главы рассматриваются отличия в предсказании различных компонент итогового запроса между русским и английским языками для ответа на **RQ-3: В каких компонентах сгенерированного SQL запроса модели чаще всего ошибаются?**

Для ответа на этот вопрос были рассмотрены компоненты SQL запроса и корректность сопоставления значений схемы. Компонентой считается часть запроса, соответствующая части SQL запроса. Например, компонента `WHERE` соответствует части запроса выбора атрибутов из полученного результата – `SELECT count(cars)`, а `JOIN` – `JOIN table-1 on table-1.a=table-2.b`. Были рассмотрены ошибки соответствия компонент SQL запроса - `SELECT`, `GROUP`, `ORDER` и тд. Эти компоненты соответствуют структуре запроса и в

[15] https://yandex.cloud/ru/services/translate

этом исследовании проведен анализ точности определения моделью логики построения запроса.

Таблица 4 — Пропорция ошибок по компонентам для BRIDGE (слева) и RAT-SQL (справа) среди исследуемых разбиений.

	SELECT	WHERE	JOIN	AND/OR
EN	0.04 / 0.04	0.04 / 0.04	0.35 / 0.27	0.10 / 0.10
MT + HT RU	0.03 / 0.05	0.04 / 0.04	0.39 / 0.34	0.10 / 0.09
RU	0.04 / 0.04	0.05 / 0.06	0.32 / 0.38	0.08 / 0.10
RU + ENG	0.03 / 0.04	0.03 / 0.04	0.34 / 0.34	0.08 / 0.10

	IUEN	ORDER	GROUP
EN	0.10 / 0.11	0.18 / 0.18	0.26 / 0.21
MT + HT RU	0.11 / 0.12	0.24 / 0.23	0.32 / 0.33
RU	0.11 / 0.11	0.26 / 0.23	0.33 / 0.33
RU + ENG	0.09 / 0.10	0.23 / 0.20	0.29 / 0.27

Таблица 5 — Пропорция ошибок по предсказанию элементов схемы для BRIDGE (слева) и RAT-SQL (справа) для исследуемых разбиений.

	SELECT (без аггр.)	WHERE (схема)	WHERE (знач.)
EN	0.20 / 0.18	0.17 / 0.16	0.23 / 0.31
MT + HT RU	0.32 / 0.29	0.22 / 0.31	0.57 / 0.57
RU	0.29 / 0.28	0.25 / 0.22	0.55 / 0.57
RU + EN	0.29 / 0.24	0.28 / 0.28	0.56 / 0.57

	FROM	GROUP BY	ORDER BY
EN	0.08 / 0.07	0.20 / 0.21	0.16 / 0.13
MT + HT RU	0.10 / 0.08	0.28 / 0.32	0.35 / 0.29
RU	0.09 / 0.09	0.29 / 0.30	0.22 / 0.23
RU + EN	0.08 / 0.07	0.29 / 0.24	0.22 / 0.19

Точность заключается в верном построении ожидаемых синтаксических конструкций и сопутствующих операторов, например агрегирующих функций

и операторов сравнения. В качестве выбранных компонент рассмотрены `WHERE` (только операции сравнения), `SELECT` (только операции агрегации), `GROUP`, `ORDER`, `AND/OR`, `IUEN`, `JOIN`. Метрика оценки – точность сопоставления строк. Из истинного запроса g_i извлекается компонента, из нее удаляются элементы схемы и значения, аналогично извлекается из p_i. Если обе строки эквивалентны, то точность совпадения для данной компоненты для запроса i равна 1 иначе 0. В то же время, порядок операторов сравнения и аггрегирующих функций не влияет на корректность запроса.

Таким образом оценивается не только наличие данной компоненты в генерируемом запросе, но и корректность генерации данной компоненты (например, корректно предсказали `ORDER BY atr-1 DESC`, а не `ASC`). Данное значение суммируется для каждой компоненты каждого g_i по тестовой выборке и делится на общее количество таких компонент в истинных запросах в тестовом наборе данных. В таблице 4, указана именно пропорция ошибок, поэтому полученная точность вычитается из 1.

Как видно в таблице 4, и русские и английские модели чаще всего ошибаются в предсказании компонент `JOIN`, `ORDER` и `GROUP`. То есть модели не понимают, что им надо сделать `JOIN` операцию таблиц и добавить условие сортировки или группировки по атрибуту. Ожидаемо, на русском качество хуже чем на английском, как следует из интегральных метрик таблицы 3. Разница в предсказании компонент (`SELECT`, `WHERE`, `AND/OR`, `IUEN`) между моделями незначительна.

Аналогично был проведен анализ сопоставления схемы и значений. Из компонент `SELECT` , `WHERE`, `FROM`, `GROUP BY`, `ORDER BY` были извлечены элементы схемы (атрибуты и таблицы) и значения (из компоненты `WHERE`). Из каждой компоненты истинного запроса g_i извлечены элементы схемы и значения и сопоставлены с элементами предсказанного запроса p_i. Например, из компоненты `SELECT` были извлечены ожидаемые к возврату атрибуты, из компоненты `WHERE` были отдельно возращены атрибуты сравнения и значения базы данных. При точном совпадении извлеченных элементов из каждой компоненты для истинного и предсказанного запроса, точность для данной компоненты равна 1 иначе 0. Данное значение суммируется для каждой компоненты каждого g_i по тестовой выборке и делится на общее количество таких компонент в истинных запросах в тестовом наборе данных. В таблице

указана именно пропорция ошибок, поэтому полученная точность вычитается из 1.

Результаты исследования показали в таблице 5, что модели на русском и английском демонстрируют схожую схему распределения ошибок, с разницей, что ошибок на английском меньше. Но видно, что средний процент ошибок, связанных со связыванием сущностей, значительно выше, чем среднее количество ошибок, связанных с пониманием компоненты и логики. То есть, ключевой источник ошибок связан именно с пониманием схемы базы данных, нежели с построением корректного синтаксического запроса. Заметно, что все модели хорошо справляются с предсказанием имен таблиц базы данных: компоненты `FROM` содержат лишь 8% ошибок. Модели на русском демонстрируют высокий процент ошибок по предсказанию элементов схемы: компоненты `SELECT`, `WHERE` (столбцы), `ORDER BY`, `GROUP BY` составляют в среднем 23% ошибок. Прогнозы, сделанные на английском наборе лучше с точки зрения сопоставления элементов, в среднем у них 18% ошибок. Сопоставление сущностей базы и вопроса (`WHERE (значения)`) особенно сложно для русских моделей – модель ошибается практически каждый второй раз. Влияние индуктивного смещения алгоритма обучения BRIDGE модели в сторону сопоставления значений видно аналогично предыдущему анализу в английском языке - ошибок в компоненте `WHERE` с подстановкой значений меньше, чем в модели RAT-SQL.

RQ-3: Вывод о наиболее затруднительных элементах предсказания SQL запроса. Сложности в предсказании итоговых запросов можно разделить на два класса - сложность в предсказании компонент запроса и сложность в предсказании элементов схемы. В предсказании компонент запроса, основная сложность состоит в верном предсказании компонент `JOIN`, `ORDER`, `GROUP`. Для предсказания элементов схемы – `SELECT`, `WHERE` (атрибуты), `ORDER BY`, `GROUP BY`. Особенную сложность для русского языка представляет задача сопоставления сущностей базы и вопроса.

1.7 Выводы

В данной главе введена формальная постановка задачи обучения text-to-SQL моделей в архитектуре sequence-to-sequence, рассмотрены методы, решающие text-to-SQL задачу, представлены принципы адаптации и описан процесс перевода первого русского text-to-SQL датасета PAUQ. В рамках адаптации англоязычных моделей был проведен набор исследований для оптимизации будущей адаптации text-to-SQL решений на русский язык.

Научный вклад и ключевые выводы исследований данной главы:

- Адаптирован и локализован первый русский датасет text-to-SQL с помощью профессиональных переводчиков со значимыми исправлениями и доработками исходного английского датасета;
- Адаптированы под русский язык и оценены англоязычные text-to-SQL модели BRIDGE и RAT-SQL;
- Исследования показали, что машинный перевод пар запросов без значений (те, что в компоненте `WHERE`) и ручной перевод примеров со значениями, позволяет получить качество на русском языке сопоставимое с полным ручным переводом датасета. SQL запросы без значений представляют 40% исходного датасета – такая оптимизация позволит в будущем быстрее и дешевле адаптировать text-to-SQL датасеты;
- Исследования показали, что совместное обучение на английском и русском языках позволяет улучшить результаты на этих языках по отдельности на 2%-5% при обучении моделей с многоязычными кодировщиками. Таким образом, при адаптации нового text-to-SQL датасета можно сразу обучать модель на двух языках и получить лучшее качество, чем только при обучении на русском;
- Определены ключевые источники ошибок в text-to-SQL моделях – связанные с компонентами запроса и с задачей сопоставления сущностей. Для лучшего качества моделей при создании text-to-SQL датасета требуется уделить особое внимание именно этим элементам SQL запросов.

В следующей главе будет рассмотрена проблема сдвига распределения в NLP, определены ключевые сдвиги в text-to-SQL задаче и проведены исследования по оценке генерализации современных text-to-SQL моделей.

Глава 2. Оценка генерализации text-to-SQL моделей

2.1 Генерализация в NLP

Генерализация нейронных сетей, также называемая способностью к обобщению (далее будет использоваться именно термин “генерализация”), представляет собой способность модели успешно переносить представления, знания и стратегии, полученные из прошлого опыта, на новые данные. Эта характеристика является ключевым показателем качества нейронных сетей [67, 68, 69, 70]. Однако данная характеристика может быть интерпретирована по-разному [71].

С одной стороны, смысль генерализации заключается в обеспечении устойчивого, надежного и справедливого поведения моделей при предсказании на данных, которые отличаются от тех, на которых модели обучались, что является критически важным при их применении в реальном мире. С другой стороны, хорошая генерализация неразрывно связана с хорошей производительностью и без нее модель не способна выполнить поставленную задачу. С третьей – хорошая генерализация заключается в том, что модели должны вести себя по-человечески, так как люди, как известно, обладают хорошей способностью к обобщению.

Генерализация в машинном обучении – многогранное понятие, и в отдельных ситуациях, нужно приоритизировать различные ее аспекты. Принято, что генерализация оценивается по тому, насколько хорошо модель работает на тестовом наборе данных, учитывая его соотношение с данными, на которых модель была обучена. В течение долгого времени существовало одно простое ограничение на это соотношение: обучающие и тестовые данные должны были быть разными. Это достигается путем случайного разделения доступных данных на обучающие и тестовые части, предполагая, что данные являются **независимыми и одинаково распределенными (i.i.d.)**. Таким образом, генерализация оценивается путем обучения и тестирования моделей на разных, но одинаково отобранных данных. За последние 20 лет наблюдается значительный прогресс в применении таких случайных разделений данных на обучение и тестирование в различных приложениях.

С момента первого выпуска Penn Treebank [72], показатели F1 для парсинга синтаксических деревьев с метками выросли с более 80% в конце прошлого века [73] до почти 90% в первые десять лет нынешнего века [74] и достигли показателей вплоть до 96% в последние годы [75].

Оценки по популярному набору тестов GLUE выросли с показателей между 60 и 70 на момент его выпуска в 2018 году [76] до значений выше 90 менее чем через год [**devlin2019bert**], с производительностью на широком спектре задач, достигающей и превосходящей уровень человеческих результатов к 2019 году (например, [**devlin2019bert**, 77]). Начиная с 2020 года крупные языковые модели [45, 78] показали впечатляющие результаты на почти всех существующих тестах понимания естественного языка с независимыми и одинаково распределенными данными (i.i.d.).

С прогрессом пришло понимание того, что для модели обработки естественного языка (NLP) достижение очень высоких или сопоставимых с человеческими результатов на тестовом наборе данных с независимыми и одинаково распределенными данными (i.i.d.) не гарантирует, что модель надежно обобщает широкий спектр различных сценариев так, как это делают люди. В последние годы появилось множество исследований, указывающих на сбои в генерализации нейронных моделей, которые демонстрируют передовые результаты на случайных разбиениях данных на обучение и тестирование [79, 80, 81, 82, 83, 84, 85]. Эти сбои происходят ввиду сдвига обучающей выборки, который может проявляться в множестве сценариев. Исследования показывают, что модели, которые хорошо справляются с i.i.d. тестами, могут полагаться на простые эвристики, которые не обеспечивают надежное обобщение в широком спектре не i.i.d. сценариев [86, 87], чрезмерно зависят от стереотипов [88, 89] или склонны к запоминанию, а не к обобщению [85, 90]. В случае со сдвигом обучающей выборки, такие эвристики предсказания не работают и поэтому модели сталкиваются с проблемами.

Сценарии сдвига обучающей выборки разнообразны - производительность модели снижается, если данные для оценки отличаются от данных для обучения по жанру, домену или теме [91, 92], или если они представляют различные субпопуляции [79, 93]. Также исследования указывают на неспособность моделей обобщать композиционно [81, 82, 94, 95], структурно [96, 97], на более длинные последовательности [98, 99] или на немного измененные формулировки одной и той же задачи [89].

2.2 Аспекты сдвига обучающей выборки

Существуют 6 аспектов генерализации нейронных сетей в NLP - эти аспекты обычно проявляются при сдвиге обучающей выборки [71]. Примеры генерализации приведены на рисунке 2.1. Считается, что модель обладает данным аспектом генерализации, если она верно поняла новые данные, полученные в результате сдвига и/или сопоставила нужную форму такому выражению.

- **Межъязыковая генерализация** – примерами межъязыковой генерализации является способность модели уметь работать с новым языком, которого не было в обучающей выборке [**devlin2019bert**, 100, 101] или способность модели к улучшению генерализации за счет совместного обучения на нескольких языках [102, 103, 104].
- **Доменная генерализация** – примером исследований доменной генерализации являются работы [105, 106] где рассматривается, как модель анализа настроений, обученная классифицировать отзывы о конкретных продуктах, обобщает свои знания на новые коммерческие продукты, которые не были представлены в ее обучающих данных. В [79] исследуется как модель, обученная на данных, собранных от одной демографической группы, обобщает на другую популяцию. В свою очередь, [92] исследуется как модель машинного перевода, обученная на каноническом тексте, обобщает свои знания на шумные данные с интернет-платформ.
- **Устойчивость** – способность моделей обучаться на основную задачу, без переобучения на ложные корреляции, которые могут появляться в данных. Например, популярные датасеты такие как Stanford NLI [107] и MultiNLI [108] страдают от случаев ложной корреляции. К примеру, в работах [109, 110] было показано, что модели могут делать правильные предсказания для NLI примеров, рассматривая только утверждения-гипотезы (одна из частей объекта x). Ложные корреляции в датасете проявляются в выборе слов и грамматических особенностях составленных предложений (например, наличие отрицания, указывающего на класс противоречия). Таким

образом, модель не делает требуемых логических заключений, а просто использует явные признаки утверждения-гипотезы.

- **Композиционная генерализация** – способность модели комбинировать известные компоненты (токены, слова) для сопоставления новых композиций [69]. Например, способность сопоставлять выражения на естественном языке с выражением на формальном языке [11, 16, 81, 111].
- **Структурная генерализация** – определяет, насколько модели могут обрабатывать или генерировать структурно (грамматически) правильные формы, в противоположность предыдущим аспектам генерализации которые заключаются в сопоставлении выражениям правильных интерпретаций (как например, в задаче text-to-SQL). В отличие от композиционной генерализация, эта степень генерализации может быть оценена на задаче языкового моделирования. Существуют две категории структурной генерализации. Синтаксическая генерализация - способность модели обобщать на новые синтаксические структуры или новые синтаксические элементы в известных синтаксических структурах. Например, в работах [97, 112] удалялись определенные формулировки предложений из обучающих данных, а затем проверялось, могут ли модели тем не менее понимать такие формулировки на тестовой выборке. Морфологическая генерализация обычно проявляется в задачах морфологической флексии [113, 114, 115, 116].
- **Генерализация к задачам** – способность моделей адаптироваться к нескольким NLP задачам. Одно из направлений это многозадачное обучение, где модель обучается сразу на несколько задач [117]. Примеры многозадачных бенчмарков – GLUE [76] и SuperGLUE [118]. Также более современные бенчмарки формулируются как sequence-to-sequence задачи и могут быть решены с помощью одной большой языковой модели [46, 119]. В больших языковых моделях также рассматривается другая сторона генерализации к задачам – способность предобученной модели к адаптации к новой задаче путем дообучения [**devlin2019bert**, 120, 121].

В эру больших языковых моделей стало популярно направление контекстного обучения, без самого процесса обучения с изменением параметров языковой модели. В качестве затравки модели демонстрируются примеры схожих задач и их соответсвующих решений(cat – 3, chicken – 7, bird – 4), а модель на основании этой информации решает требуемую задачу (tiger – ?) [45, 122]. Такой подход был рассмотрен в методах решения text-to-SQL в первой главе.

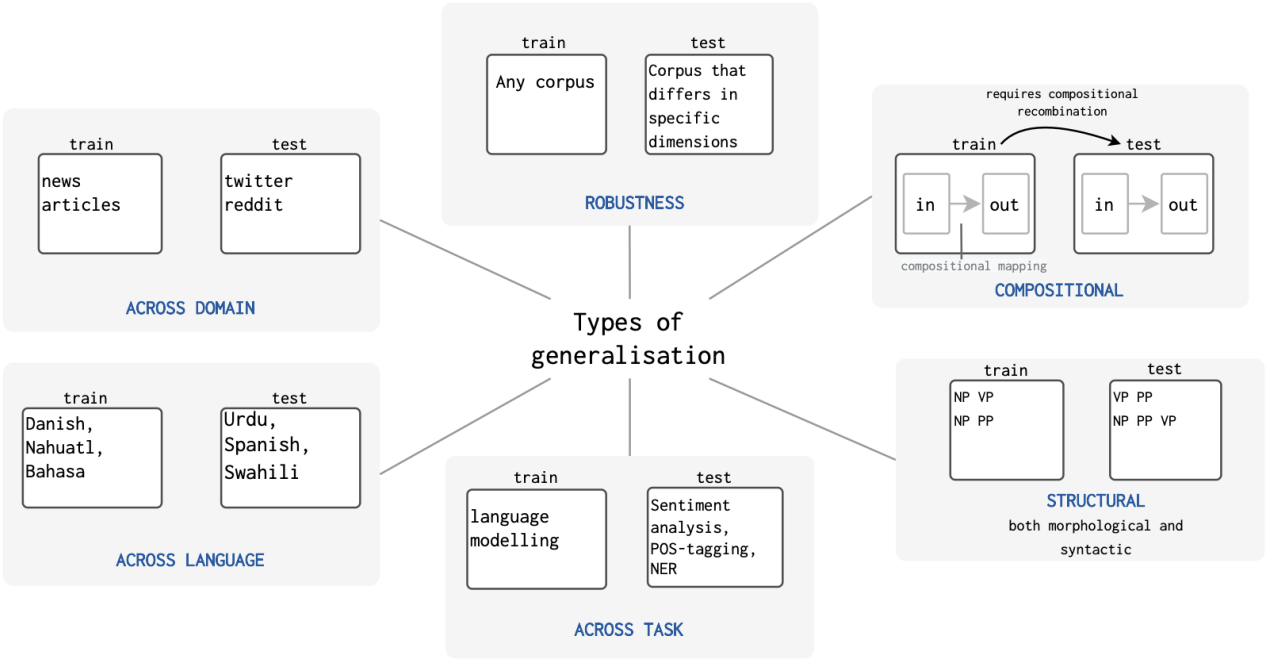

Рисунок 2.1 — Шесть аспектов генерализации в NLP. Across domain - Доменная генерализация. Robustness - устойчивость. Compositional - Композиционная генерализация. Structural - Структурная генерализация. Across task - Генерализация к задачам. Across Language - Межъязыковая генерализация.

Среди всех аспектов генерализации, в контексте семантического парсинга особое внимание необходимо уделить аспектам композиционной генерализации. Важный вопрос - **как оценивать композиционную генерализацию моделей?** На рисунке 2.2 представлено 3 ключевых функциональных теста [13]:

– Тест систематичности – оценка способности модели комбинировать известные компоненты языка, чтобы формулировать новые

композиции. Для оценки систематичности требуется, что два слова a и b не встречаются вместе в одном контексте (но при этом такой контекст возможен), и слова a и b оба присутствуют в различных контекстах в обучающей выборке.

– Тест продуктивности – оценка продуктивности определяет способность модели генерировать последовательности длиннее тех, что были в обучающей выборке.
– Тест замещаемости – оценка замещаемости определяет способность модели к корректной интерпретации синонимов в рамках одного и того же контекста.

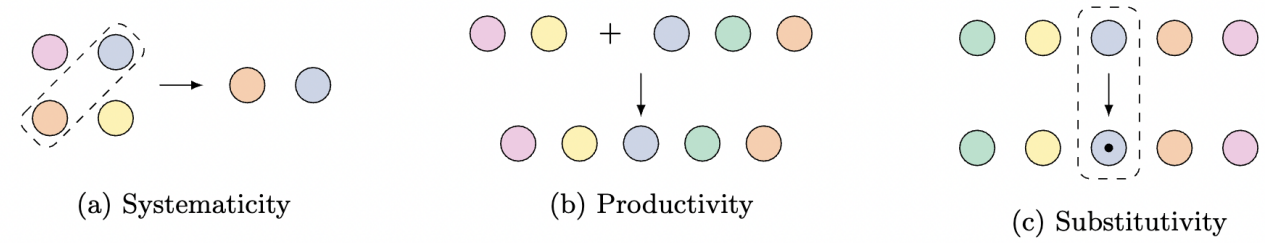

Рисунок 2.2 — Схематические изображения трех тестов для проверки композиционности моделей нейронных сетей. (a) **systematicity** - систематичность (b) **productivity** - продуктивность. (c) **substitutivity** - замещаемость.

Представленные аспекты генерализации в обучении нейронных сетей являются примером сдвига обучающей выборки. Пусть совместное распределение данных представлено выражением 2.1, где $p(x)$ распределение независимой переменной x, а $p(y|x)$ – распределение целевой переменной y с учетом x.

$$p(x,y) = p(x)p(y|x) \tag{2.1}$$

Сдвиг распределения – это ситуация, когда $p(x_{tr},y_{tr}) \neq p(x_{ts},y_{ts})$, где tr - обучающая выборка, ts - тестовая выборка.

Существует 3 вида сдвига обучающей выборки:

– Ковариационный сдвиг ($p(x_{tr}) \neq p(x_{ts})$ и $p(x_{tr},y_{tr}) = p(x_{ts},y_{ts})$) - сдвиг, при котором происходит сдвиг входных данных $p(x)$, но распределение выходных данных остается таким же. Примерами бенчмарков с

ковариационным сдвигом являются HANS [83], PAWS [123], COGS [81].

– Сдвиг целевой переменной ($p(x_{tr}) = p(x_{ts})$ и $p(x_{tr},y_{tr}) \neq p(x_{ts},y_{ts})$) – сдвиг при котором меняется распределение целевой переменной. Сдвиг целевой переменной может произойти в случае несогласия разметчиков в разметке данных (например, разметка SQL запросов экспертами по MySQL и PostgreSQL может привести к различным результатам).
– Полный сдвиг ($p(x_{tr}) \neq p(x_{ts})$ и $p(x_{tr},y_{tr}) \neq p(x_{ts},y_{ts})$) – одновременное изменение распределения входных и выходных данных. Примерами полного сдвига являются задачи семантического парсинга, например задача text-to-SQL в условиях доменного или композиционного сдвига [8, 20, 30].

Наконец, последней компонентой сдвига обучающей выборки является источник сдвига. Источник сдвига определяет, насколько экспериментатор контролирует обучающие и тестовые данные и какие выводы можно сделать из эксперимента по генерализации [71]. Можно выделить четыре источника сдвига: естественный сдвиг, искусственно разделенные естественные корпуса, сгенерированный сдвиг и полностью сгенерированные наборы данных. Пусть есть параметр τ – интересующее свойство данных, в отношении которого проверяется способность к обобщению. Тогда распределение данных для исследования будет задано как $p(x, y, \tau)$, где x, y – исходное распределение данных. С точки зрения исследователя, используется операция разделения $f : (\tau) -> \{\text{true}, \text{false}\}$ – разделение исследователем данных на обучающие и тестирующие по какой-то логике (например, положить более длинные примеры в тестовую выборку):

$$\begin{aligned} y(x_{tr}, y_{tr}) &= p(x,y|f_{tr}(\tau) = \text{true}) \\ y(x_{ts}, y_{ts}) &= p(x,y|f_{ts}(\tau) = \text{true}) \end{aligned} \tag{2.2}$$

Существуют 4 типа источника сдвига [71]:

– **Естественный сдвиг** – сдвиг, при котором экспериментатор не имеет полного контроля над исходным распределением и правилом разделения $f(\tau)$. Примерами естественного сдвига являются источники данных от разных аннотаторов [124], различных доменов [125], популяций [93], временных периодов [126] или разных процессов, направленных на решение одной и той же задачи [76].

– **Искусственное разделение естественных данных** – использование естественных данных, но таких, которые искусственно разделены по определенным параметрам – то есть экспериментатору известно правило разделения $f(\tau)$ (например, длина данных [127] или синтаксическая сложность [97]) выборки. Таким образом, экспериментаторы не контролируют сами данные, но они контролируют схему разбиения $f(\tau)$.
– **Сгенерированный сдвиг** - представляют собой случай, в которых обучающая выборка представляет собой полностью естественный корпус, а тестовая разработана с учетом определенных свойств для решения интересующего аспекта обобщения. Например, тестовая выборка может быть сконструирована с учетом наличия или отсутствия определенных шаблонов [128, 129], нарушая определенные правила по которым была сконструирована обучающая выборка [130] или была сконструирована в состязательной манере [131].
– **Полностью сгенерированное распределение** – такое распределение данных является полностью синтетическим, где экспериментатор имеет полный контроль над исходным распределением $p(x, y)$ и правилом разбиения τ. Такое разбиение является наиболее подходящим для исследования генерализации, но при этом предлагает очень ограниченное распределение языка. Примерами таких распределений и сдвигов являются полностью синтетические данные, такие как [13, 111].

Введеные понятия определяют сдвиг обучающей выборки в NLP и какие аспекты генерализации нейронных сетей позволяют бороться с этим. На основании этих понятий, далее будет сформирована постановка эксперимента для оценки генерализации в text-to-SQL задаче.

2.3 Оценка генерализации в text-to-SQL задаче

Модели text-to-SQL, как частный пример моделей семантического парсинга, должны обладать двумя важнейшими аспектами генерализации для успешного использования в производстве – **доменная генерализация**

и **композиционная**. Устойчивость, как свойство модели, уметь выделять ключевые признаки в данных для принятия решения, а не основываться на ложной корреляции – актуальна для всех моделей машинного обучения, поэтому она не является специфичным аспектом text-to-SQL моделей.

Таблица 6 — Статистика PAUQ XSP разбиений для русского (Ru PAUQ XSP) и английского языков (En PAUQ XSP) . Train соответствует обучающей выборке. Test – тестовой.

	Ru PAUQ XSP		**En PAUQ XSP**	
	Train	Test	Train	Test
Размер разбиения	8800	1076	8800	1076
Средняя длина вопроса	8.95	9.05	12.01	12.31
Средняя длина запроса	24.79	21.28	24.79	21.28
Средняя длина шаблона	21.04	16.79	24.51	21.14
% тестовых шаблонов в Train	9		8	
% тестовых вопросов в Train	0		0	

Тестирование доменных и композиционных аспектов генерализации требует формирования полных сдвигов из секции 2.2. Для исследования генерализации text-to-SQL моделей были разработаны и применены алгоритмы **искусственного разделения естественных данных**. Был выбран именно такой тип сдвига для оценки генерализации, так как SQL запросы и соответствующие вопросы сложно качественно сгенерировать в достаточном количестве для корректной оценки. Разделение исходного датасета по выбранной характеристике позволяет оценить генерализацию text-to-SQL моделей. В качестве исходного датасета был выбран датасет PAUQ.

Для оценки доменной генерализации text-to-SQL моделей данные датасета PAUQ были разделены в XSP (cross-database setting) манере [132]. Таким образом, в обучающей выборке датасета есть один набор баз данных D_{tr}, а в тестовой другой D_{ts}, так что $D_{tr} \cap D_{ts} = \varnothing$. Такое разбиение далее будет определяться как **PAUQ XSP**. В противоположность XSP разбиению имеется SSP (single database setting) постановка, когда обучающая выборка нацелена на тот же набор баз данных, что и тестовая выборка. Ключевые характеристики

PAUQ XSP разбиения приведены в таблице 6. Для оценки композиционной генерализации text-to-SQL моделей предлагаются следующие разбиения:

– **Разбиение для оценки систематичности** формируется за счет разделения датасета на две части на основании шаблонов запросов. Для этого над SQL запросами была проведена операция шаблонизации - маскирование элементов схемы и значений и приведение к одному регистру всех запросов. Таким образом, исходные SQL запросы стали представлены в формате
`SELECT ATRIBUTE FROM TABLE WHERE ATRIBUTE = NUMERIC`.

Таблица 7 — Статистики композиционных разбиений датасета PAUQ. Train соответсвует обучающей выборке. Test – тестовой. Все статистики приведены для английского языка.

	Random SSP		**Template SSP**		**Paraphr. SSP**	
	Train	Test	Train	Test	Train	Test
Размер разбиения	3829	986	3446	986	1814	226
Ср. длина вопроса	12.87	12.69	13.03	12.06	10.62	10.51
Ср. длина запроса	22.96	23.06	23.55	21.08	14.36	15.17
Ср. длина шаблона	22.79	22.85	23.38	20.88	14.2	14.99
% тест. шаблонов в Train	44		0		100	
% тест. вопросов в Train	10		0		0	

	Long train SSP		**Long test SSP**	
	Train	Test	Train	Test
Размер разбиения	3428	981	3444	986
Ср. длина вопроса	13.72	9.79	12.13	15.22
Ср. длина запроса	27.19	8.69	17.13	43.44
Ср. длина шаблона	27.01	8.56	16.98	43.15
% тест. шаблонов в Train	0		0	
% тест. вопросов в Train	0		0	

Для оценки систематичности были предложены два разбиения на основе шаблонов - **Template SSP** и **Train long SSP**. **Template SSP**

- разбиение по шаблонам, так что шаблоны не пересекаются между обучением и тестированием в постановке SSP. **Train long SSP** – аналогичное разбиение по шаблонам, но с условием, что шаблоны в обучении длиннее шаблонов при тестировании.

– **Разбиение для оценки продуктивности** распределяет более длинные шаблоны в тестовую выборку, а более короткие в обучающую. Данное разбиение далее обозначается, как **Test long SSP**.

– **Разбиение для оценки замещаемости** распределяет парафразы, соответствующие одному и тому же шаблону, между обучающей и тестовой выборкой без пересечения парафраз. Данное разбиение далее обозначается как **Paraphrase SSP**.

Каждое композиционное разбиение формируется в постановке SSP и следует принципу композиционности из секции 2.2 - каждый SQL токен из тестовой выборки хотя бы раз присутствует в обучающей. Для сравнения уровня генерализации было также добавлено **Random SSP** разбиение – случайное (i.i.d.) разбиение в формате SSP. Статистики разбиений приведены в таблице 7. Реализации алгоритмов по разбиению данных для датасета PAUQ для оценки генерализации приведены в данном репозитории[1].

2.4 Эксперименты по оценке генерализации text-to-SQL моделей

Для исследования генерализации на разработанных разбиениях было выбрано 5 популярных text-to-SQL моделей: кодировщик-декодировщик T5 (T5-base и T5-3B), декодировщик Llama3-8B (в формате PeFT (Parameter Efficient Fine-tunining) [133] обучения и полного дообучения SFT (supervised finetuning)) и решения с индуктивным сдвигом в сторону задачи text-to-SQL: RESDSQL, RAT-SQL, BRIDGE. Параметры обучения и экспериментов каждой модели приведены в Аппендиксе А. Каждый запуск модели проводился от 3 до 5 раз. Отклонения в результатах не превышают 1% по метрике Execution Match.

В таблице 8 приведены результаты обучения в SSP постановке задачи на английском языке. Из таблицы видно, что в сравнении с **Random SSP** разбиением хуже всего модель способна обобщать в случае **Test long**

[1]https://github.com/runnerup96/splitting-strategies

SSP разбиения с качеством ниже 50% – у предобученных моделей слабая способность к продуктивности, генерации более длинных примеров, нежели в обучающей выборке. У моделей, обученных на **Train long SSP** и **Template SSP**, наблюдается более высокий уровень способности к систематичности – модели способны верно создавать новые запросы, шаблонов которых не было в обучающей выборке.

Таблица 8 — Execution Match на композиционных разбиениях данных SSP на английском языке. Зеленым цветом выделено разбиение с лучшим качеством. Random SSP разбиение приведено для сравнения с i.i.d. разбиением.

Разбиение	T5-base	T5-3B	RESDSQL	Llama3 LoRA	Llama3 SFT
Random SSP	**0.66**	**0.95**	**0.92**	**0.94**	**0.96**
Train long SSP	0.55	0.78	0.80	0.79	0.81
Test long SSP	0.13	0.25	0.28	0.28	0.31
Template SSP	0.52	0.73	0.69	0.70	0.75
Paraphrase SSP	**0.68**	**0.94**	**0.93**	**0.93**	**0.95**

Обобщение модели, обученной на разбиении **Train long SSP** дает понять, что модели гораздно лучше генерализируют при обучении на сложных примерах, нежели на простых (**Test long SSP**). Размер модели тоже имеет значение – самая большая модель Llama3 с 8 млрд. обучаемых параметров имеет лучшее качество даже по сравнению с моделью RESDSQL с индуктивным смещением алгоритма обучения.

Таблица 9 расширяет результаты экспериментов, представленных в первой главе. В дополнение к экспериментам на RAT-SQL и BRIDGE модели в постановке XSP были обучены на объединенных датасетах (русский и английский) PAUQ. Результаты показывают, что для модели кодировщик-декодировщик T5-base качество на русском языке также улучшается. Результаты на RESDSQL модели ухудшаются – причина заключается в компоненте классификации элементов схемы – использованный многоязычный кодировщик решает задачу с низким качеством предсказания элементов схемы по F1 метрике, около 80%. В тоже время одноязычный кодировщик решает эту задачу с качеством 99%. Таким образом, решение с индуктивным сдвигом в сторону text-to-SQL задачи не масштабируется на несколько языков в отличие от моделей, которые не используют многоязычный

кодировщик для решения специфической задачи сопоставления сущностей, а решают общую задачу text-to-SQL с помощью многоязычного кодировщика.

Таблица 9 — Execution Match метрика для оценки доменной генерализации моделей в PAUQ XSP разбиении. En - английский язык, Ru - русский язык, M - объединение En и Ru версий датасета PAUQ.

Train	Test	T5-base	RESDSQL	RAT-SQL	BRIDGE
En	En	0.63	0.74	0.63	0.60
M		0.63	0.68	**0.65**	**0.65**
Ru	Ru	0.57	0.43	0.49	0.48
M		**0.63**	0.42	**0.53**	**0.53**

Таблица 10 — Распределение ошибок, усредненное по моделям T5-base, T5-3B, RESDSQL, RAT-SQL, BRIDGE и Llama3-8B (PeFT и SFT).

Разбиение	Ошибка предсказания схемы	Синтаксическая ошибка	Ошибка предсказания значений
Template SSP	0.72	0.08	0.2
Train long SSP	0.76	0.04	0.2
Test long SSP	0.71	0.04	0.25
Paraphrase SSP	0.59	0.09	0.32
En PAUQ XSP	0.69	0.08	0.23
Ru PAUQ XSP	0.71	0.08	0.21

Если сравнить результаты SSP обучения из таблицы 8 и результаты XSP обучения из таблицы 9 по моделям T5-base и RESDSQL, то видно что SSP разбиение значительно проще для моделей. Например разница в обучении RESDSQL на XSP и SSP составляет 18%.

На основании результатов обучения был также произведен анализ источников ошибок для исследуемых разбиений в таблице 10. Ключевое наблюдение – проблема генерализации в большей степени проявляется в ошибке предсказания элементов схемы базы данных. Модели менее чем в

10% случаях генерируют некорректный SQL запрос. Предсказание значений (в компоненте `WHERE`) тоже является значительным источником ошибок, с пропорцией ошибочных генераций от 20% до 32%. С примерами некорректных генераций можно ознакомится в Аппендиксе B.

2.5 Многозадачное обучение в задаче text-to-query

Эксперименты из секции 2.4 второй главы показали, что у предобученных моделей слабая композиционная генерализация. Один из источников ошибки – некорректное предсказание синтаксиса запроса. Далее представлен многозадачный подход к моделированию последовательности, исследованный конкретно на предсказании корректного шаблона запроса к базе данных.

Известно, что многозадачный подход к обучению позволяет учить модели более устойчивые к различным видам сдвига в области NLP. Интуиция эффективности подхода заключается в том, что одна модель обучается на разных источниках данных, которые обычно дополняют друг друга. Таким образом, модель может с разных сторон посмотреть на задачу и избежать проблемы переобучения, в том числе проблемы ложной корреляции. В NLP многозадачное обучение имеет два применения – решение одновременно множества семантически схожих задач (выделение именованных сущностей и их сопоставление [134]) или разложение сложной проблемы на части, где каждая задача – это решение глобальной подзадачи (задача семантического парсинга [135]). В этой секции исследован многозадачный подход к разложению целевого запроса на несколько компонент. Концепция компонент была представлена ранее в главе, посвященной анализу ошибок в SQL запросах. В данной главе разработана контекстно-свободная грамматика для двух языков структурированных запросов — SQL (WikiSQL [29]) и SPARQL (LC-QuAD [136]) — и использована для разложения целевого языка запросов в компоненты. Созданные компоненты выступают в качестве задач для многозадачного обучения, которое реализовано с помощью предобученной языковой модели T5. Эффективность многозадачной модели оценена в задаче преобразования текста в запрос в сравнении с классическим дообучением T5 модели на различных разбиениях данных, которые имитируют различные

сдвиги: случайное разбиение (i.i.d.) и 2 вида композиционных разбиений. Главный исследовательский вопрос данной секции – **является ли стратегия многозадачного обучения более устойчивой к композиционным сдвигам при преобразовании вопроса в запрос?** Код обучения, методы разбиения данных и анализ представлены в репозитории[2].

2.5.1 Многозадачное обучение в text-to-query задаче

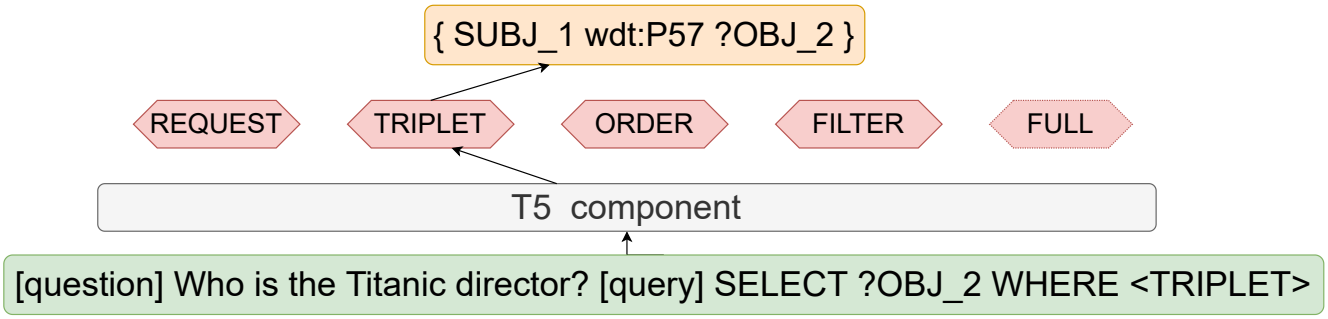

Рисунок 2.3 — Схема многозадачного обучения для языка SPARQL.

Автором разработана многозадачная модель на основе предобученной модели T5 [46] в архитектуре кодировщик-декодировщик. В исследуемой постановке, для обучения модели генерации запроса, исходный запрос разбивается на несколько компонент и каждый выходной слой моделирования языковой последовательности генерировал свою часть запроса. Также добавляется дополнительный слой, чтобы модель умела генерировать запрос полностью.

Моделирование композиций, как задач обучения Композиционная генерализация – одна из важнейших аспектов генерализации моделей семантического парсинга, позволяющая модели комбинировать известные синтаксические элементы в соответствии с входным вопросом. В контексте композиционного обобщения будут использованы термины *токены* и *композиции*. Термин **композиция** будет использоваться в том же понимании,

[2]https://github.com/runnerup96/Multi-task-for-text-to-query

что и введенный ранее термин **компонента** – часть структуры SQL/SPARQL запроса (например, часть `SELECT`).

Существуют токены набора данных `SELECT`, `COUNT`, `SUM`, `ATR_1`, `ATR_2` (`ATR_1` и `ATR_2` являются атрибутами таблицы). Последовательность произвольной длины этих токенов, например `SELECT COUNT ATR_1` называется *композицией*. Примером композиционного обобщения является случай, когда композиции `SELECT COUNT ATR_1`, `SELECT SUM ATR_2` наблюдались во время обучения, а модель тестировалась на ранее невстречающейся композиции `SELECT SUM ATR_1`.

Понятие композиций в семантическом парсинге помогает сформировать задачу обучения для многозадачного обучения. Логический запрос формируется из композиций, которые можно описать с помощью контекстно-свободной грамматики. Автором разработана контекстно-свободная грамматика для каждой композиции для языков SQL и SPARQL. Для обучения маскирована одна из композиций запроса и модель обучается предсказывать замаскированную композицию, учитывая вопрос и остальную часть запроса (которая не вошла в маску) в качестве контекста. Получается, что генерации каждой композиции соответствует отдельная задача и за ее решение отвечает отдельный линейный слой. В качестве основной модели, кодирующей последовательность, в данной постановке используется предобученная модель архитектуры кодировщик-декодировщик без последнего линейного слоя, отображающего скрытое представление в словарь токенов.

Во время обучения используется стратегия префиксации T5 [46] — вопрос, контекст запроса и схему (если она присутствует) базы знаний обозначаются с помощью служебных токенов `[question]`, `[query]`, `[schema]` . Каждая композиция маскируется названием соответствующей маски, например маска для триплета – `<TRIPLET>`. Также в модели существует отдельный слой для предсказания полного запроса, когда модели задан только вопрос и префикс запроса с маской `<FULL>`. Для многозаданого обучения определены следующие задачи:

- `REQUEST` - линейный слой отвечающий за выбор запрашиваемых атрибутов запроса и возможных агрегатных функций. Например, `select ?OBJ_3` в SPARQL и `SELECT COUNT ATR_1` в SQL;
- `TRIPLET` - линейный слой отвечающий за создание триплетов языка SPARQL, например `SUBJ_1 p:P97 ?OBJ_2.`;

- `FILTER` - линейный слой отвечающий за создание операции фильтрации в запросах SPARQL,
  например `filter ( ?OBJ_2 < NUM_VALUE_1 );`
- `ORDER` - линейный слой отвечающий за создание операции упорядочивания в запросах SPARQL,
  например `order by asc ( ?OBJ_3 );`
- `COMPARISON` - линейный слой отвечающий за моделирование условий запроса в SQL, например `ATR_1 = STR_VALUE_2`. Условия могут включать `AND` или `OR` операторы;
- `FULL` - линейный слой отвечающий за моделирование полной последовательности запроса с контекстом и префиксом `<FULL>`.

Algorithm 1 *

Алгоритм многозадачного обучения

Инициализация $(\beta, \{w^t\}_{t \in t_1, t_2, ...t_n})$
$m \leftarrow min(|t_1|, |t_2|, ...|t_n|)$
for шага в m **do**
  for $t \in t_1, t_2, ...t_n$ **do**
    $(x, y) \leftarrow GetBatchFromTask(t)$
    $\varphi \leftarrow \beta(x)$
    $p \leftarrow w^t(\varphi)$
    $L \leftarrow Loss(p, y)$
    $\beta \leftarrow \beta - \mu \nabla_\beta L$
    $w^t \leftarrow w^t - \mu \nabla_{w^t} L$
  end for
end for

В итоге, в многозадачное обучение входят следующие задачи (i) `REQUEST`, `COMPARISON` и `FULL` для обучения генерации SQL запроса; (ii) `REQUEST`, `TRIPLET`, `ORDER`, `FILTER` и `FULL` для обучения генерации SPARQL запроса.

Многозадачное обучение с планировщиком Разразработанная модель состоит из двух частей – предобученная модель β, следующая архитектуре модели T5, состоящая из трансформеров архитектуры кодировщик и декодировщик и линейный слой w, который отображает внутреннее представление нейронной сети β в пространство токенов V. Как показано

на рисунке 2.3, для обучения sequence-to-sequence модели линейный слой $w \in R^{h \times V}$(h - размерность скрытого слоя декодировщика, V - размерность словаря языковой модели) заменяется множеством линейных слоев $w^{t_1}, w^{t_2}, ..., w^{t_n}$ для каждой задачи t, каждый из которых инициализирован случайными весами.

Во время обучения алгоритмом round-robin обучаются соответствующие линейные слои $w^{t_1}, w^{t_2}, ..., w^{t_n}$ для задач t_i, а также сама предобученная модель β. В качестве функции потерь L используется отрицательный логарифм правдоподобия для предсказания следующего токена $y_{i,k}$, учитывая предыдущую подпоследовательность токенов $y_{i,<k}$ в наборе данных размером N с длиной последовательности K и контекст обучения (вопрос и незамаскированная часть запроса в многозадачном случае) c_i. Многозадачная модель $[\beta, w^{t_1}, w^{t_2}, ..., w^{t_n}]$ или классическая модель $[\beta, w]$ параметризованы весами θ.

$$\mathcal{L}(X, Y, \theta, C) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \log P(y_{i,k} | x_i, y_{i,<k}, \theta, c_i). \tag{2.3}$$

Градиенты от функции $\nabla_{w^t}, \nabla_\beta$ потерь используются для последовательного обновления модели β и соответствующего задаче t линейного слоя w^t в режиме round-robin методом градиентного спуска. Выше представлен алгоритм обучения. Для каждого вопроса случайным образом выбирается одна из задач t. Данные подготовлены для задачи таким образом, что примеры для каждой задачи не пересекаются друг с другом.

2.5.2 Наборы данных

LC-QuAD [136] – это набор пар выражений и соответствующих SPARQL запросов к WikiData[3] графу знаний. WikiSQL – это набор пар выражений и соответствующих SQL запросов, где каждому запросу соответствует своя таблица с данными. WikiSQL датасет составлен в формате SSP. Для предварительной обработки данных были маскированы элементы схемы и текстовые, и численные значения в LC-QuAD и WikiSQL с помощью `TABLE`, `ATRIBUTE`, `NUM_VALUE` и `STR_VALUE` масок. В LC-QuAD были также

[3]https://www.wikidata.org/wiki/Wikidata:Main_Page

нормализованы субъектные и объектные сущности: известные сущности языка SPARQL были замаскированы с помощью `SUBJ/OBJ` маски, неизвестные объекты — с помощью `?SUBJ/?OBJ` маски. Чтобы модель также училась верно определять позиции сущностей в SPARQL запросах, ко всем замаскированным сущностям был добавлен индекс (например, `SUBJ_1`).

Разбиения данных Для оценки композиционности во время разбиения данных требуется, чтобы каждый токен из тестового набора хотя бы один раз появлялся в наборе для обучения. И Lc-Quad, и WikiSQL направлены на одну базу знаний, поэтому каждое разбиение называется SSP. Было подготовлено 3 разбиения данных для оценки метода обучения:

- **Random SSP**: оно представляет собой случайное разбиение данных с обязательным условием, что обучающий и тестовый набор имеют полное пересечение с точки зрения токенов.
- **Разбиение по длине запроса (Train/Test long SSP)**: исходные датасеты разделяется по длине запроса, так что тестовый набор данных содержит запросы другой длины, чем те, что находятся в обучающем наборе. В связи со спецификой набора данных LC-QuAD, были помещены более длинные запросы в обучающий набор, а в наборе данных WikiSQL были помещены более длинные запросы в тестовый набор.
- **Разбиение по шаблонам (Template SSP)**: исходные датасеты разделяются по шаблонам запроса, так что шаблоны запроса из тестовой выборки не встречаются в шаблонах запросов обучающей выборки.

Анализ разбиений Каждый вопрос в каждом разбиении уникален, и между вопросами в обучающем и тестовом наборах нет пересечения. Целевые запросы этих наборов могут частично перекрываться. В частности, Random SSP LC-QuAD содержит пересечение 36% в целевых запросах, а разделение Template SSP – 17%, целевые запросы не пересекаются в разбиении по длине (Train/Test long SSP). Аналогично, разбиения WikiSQL Random SSP и Template SSP содержат пересечение 23% и 18% соответственно, целевые запросы не пересекаются в разбиении по длине. С точки зрения длины запроса разбиения LC-QuAD Random SSP и Template SSP средняя длина запроса составляет 17

токенов как в обучающем, так и в тестовом наборе. При разбиении Target Length SSP средняя длина обучающих и тестовых запросов в Lc-QuAD датасете составляет 23 и 12 токенов соответственно. В WikiSQL длина запросов в обучающем и тестовом наборе составляют 9 и 14 токенов соответственно.

Метод оценки В качестве оценки обучения использована метрика **Exact Match** точного сопоставления истинного и сгенерированного запросов. Также использована метрика BLEU для оценки качества нграммного пересечения (4 нграммы) и NGRAM F1 для оценки точности предсказания токенов, которые должны быть в запросе.

Таблица 11 — Метрики для классической T5 модели и многозадачной модели на датасетах WikiSQL и Lc-QuAD.

Разбиение	WikiSQL		
	Точность	BLEU	NGRAM F1
Random SSP	0.74	0.92	0.97
	0.89	**0.95**	**0.98**
Train/Test long SSP	0.17	0.69	0.87
	0.46	**0.88**	**0.95**
Template SSP	0.61	0.85	0.95
	0.75	**0.89**	**0.97**

Разбиение	LC-QuAD		
	Точность	BLEU	NGRAM F1
Random SSP	0.71	0.91	0.97
	0.78	**0.95**	**0.98**
Train/Test long SSP	0	0.16	0.68
	0.75	**0.93**	**0.98**
Template SSP	0.37	0.76	0.92
	0.85	**0.96**	**0.99**

Таблица 12 — Покомпонентная точность для WikiSQL и LC-QuAD. Если ожидаемый и предсказанные запросы не содержат определенную композицию - это засчитывается за верный ответ.

Разбиение	WikiSQL	
	REQUEST	COMPARISON
Random SSP	0.85	0.9
	0.94	**0.94**
Train/Test long SSP	0.58	0.54
	0.88	**0.76**
Template SSP	0.76	0.82
	0.85	**0.87**

Разбиение	LC-QuAD			
	REQUEST	FILTER	TRIPLET	ORDER
Random SSP	0.88	0.97	**0.87**	1
	1	**1**	0.84	1
Train/Test long SSP	0.18	1	0.08	1
	1	1	**0.85**	1
Template SSP	0.73	0.99	0.61	1
	1	**1**	**0.89**	1

2.5.3 Результаты экспериментов

В сравнительном анализе метрик генерации запросов, представленных в таблице 11, наблюдается значительное улучшение по Exact Match по сравнению с обычной моделью для разбиения Random SSP (в среднем увеличение на 11% среди WikiSQL и LC-QuAD). Значительное улучшение наблюдается в предсказании на Train/Test long SSP – особенно заметно, что на LC-QuAD MT T5 модель начинает генерировать новые шаблоны с увеличенной способностью к продуктивности.

Способность к систематичности также заметно улучшается: на Template SSP разбиении метрики улучшаются на 50% на датасете LC-QuAD и на 14% на датасете WikiSQL. NGRAM F1 показывает высокие оценки на каждом разбиении, хотя Exact Match метрика низкая, так как модели предсказывают токены, которые должны быть в истинном запросе, но не способны их верно структурировать. Результаты в таблице 12 подтверждают результаты высокого Exact Match – многозадачная модель с высоким качеством предсказывает каждую компоненту. Модель в основном ошибается в компоненте `WHERE` в WikiSQL, а в LC-QuAD в `COMPARISON` и `TRIPLET`.

Таблица 13 — Точность предсказания на ранее не встречающихся в обучающей выборке композициях.

Разбиение	Модель	REQUEST WikiSQL	COMPARISON WikiSQL	TRIPLET LC-QuAD
Random SSP	T5	0.59	**0.75**	**0.48**
	MT T5	**0.64**	0.73	0.34
Train/Test long SSP	T5	0.33	0.26	0.02
	MT T5	**0.52**	**0.50**	**0.83**
Template SSP	T5	0.49	0.62	0.27
	MT T5	**0.58**	**0.66**	**0.85**

В таблице 13 для каждого разбиения показано, как экспериментальные модели работают с композициями, которых нет в обучающем наборе. Приведенный компонентный анализ демонстрирует, что многозадачная модель MT T5 способна обобщать на новые компоненты со средним увеличением на 23% по всем компонентам.

Ошибки в предсказании для обеих моделей проявляются при неправильном предсказании предикатов в SPARQL запросах и неправильном выборе количества сравнений, типов сравнений и агрегационных функциях в SQL запросах.

2.5.4 Обсуждение результатов

Данный метод был разработан и исследован на упрощенной задаче text-to-query – генерации шаблонов запросов. Метод показал себя успешно и улучшил результаты генерации шаблонов по сравнению с дообучением обычной модели T5. Как было исследовано ранее, основная сложность в text-to-SQL домене состоит предсказании элементов схемы. В данной работе не исследуется эффективность данного метода при предсказании элементов схемы запроса. Дальнейшая работа над многозадачным методом языкового моделирования является перспективным исследованием для борьбы с ложной корреляцией в больших языковых моделях.

2.6 Выводы

В главе представлены аспекты генерализации в NLP, рассмотрены типы сдвигов, где проявляется генерализация и определено, что является источником сдвига. Были определены ключевые аспекты генерализации в text-to-SQL моделях, определен набор экспериментов и проведен анализ результатов поведения моделей в условиях сдвига обучающей выборки.

Главными выводами исследования данной главы является:

– Самым сложным разбиением для text-to-SQL задачи является сдвиг по длине, когда в тестовой выборке лежат запросы более длинные, нежели в обучении. Это значит, что у современных text-to-text моделей слабая способность к продуктивности и в практическом применении, таких характеристик обучающей выборки надо избегать. В тоже время, разбиение по парафразам показывает лучшее качество по сравнению с другими композиционными разбиениями – значит, что способность к замещаемости у совеременных языковых моделей хорошая.
– text-to-SQL в XSP постановке является более трудной задачей для языковых моделей нежели в SSP. В некоторых случаях разрыв в качестве составляет более 20%.

- Главный источник ошибок text-to-SQL – ошибки в предсказании элементов схемы в запросах. Несмотря на то, что композиционность связана с синтаксисом языка, этих ошибок в современных моделях меньше всех. Таким образом, в будущих исследованиях стоит уделить внимание имеенно задаче сопоставления сущностей;
- Наличие параллельного корпуса text-to-SQL данных позволяет улучшить качество text-to-SQL моделей на целевом языке.
- Метод многозадачного обучения показывает свою эффективность по сравнению с классическим дообучением модели T5 на простых text-to-SQL и text-to-SPARQL датасетах для предсказывания шаблона запроса.

В следующей главе будет описана реализованная text-to-SQL система на бенчмарке вопросов работников больницы к реляционным базам данных.

Глава 3. Методы разработки text-to-SQL решений в условиях сдвига обучающей выборки

В этой главе будут рассмотрены методы создания text-to-SQL систем на основе бенчмарка EHRSQL, который посвящён решению задачи text-to-SQL в контексте обращений работников больницы к реляционной базе медицинских данных. Электронные медицинские записи (EHR, **E**lectronic **H**ealth **R**ecords) содержат в себе информацию о пациентах, их диагнозах и методах лечения. Для работников больницы очень критично, чтобы был быстрый доступ к информации по пациенту для незамедлительного принятия верного решения. text-to-SQL система – эффективное решение этой проблемы.

3.1 Описание EHRSQL бенчмарка

EHRSQL представляет собой набор вопросов медицинских работников к базе данных о пациентах. Существует две версии этого датасета – EHRSQL [33] и EHRSQL Shared Task [137]. EHRSQL Shared Task является измененной и улучшенной с помощью ChatGPT версией датасета – эта версия рассматривается в работе.

Особенности датасета следующие:

1. **Широкий спектр вопросов на естественном языке:** В университетской больнице был проведен опрос для сбора вопросов, часто задаваемых по медицинским данным. Всего в опросе участвовало 222 человека с различным опытом работы в своей профессии. Полученные вопросы были отфильтрованы и шаблонизированы. Шаблонизированным вопросам был сопоставлен SQL запрос. Полученные шаблоны охватывают широкий спектр вопросов, включая извлечение историй пациентов (например, жизненные показатели и стоимость пребывания в больнице) и сложные операции группировки (например, определение наиболее часто назначаемых лекарств после постановки диагноза и определение выживаемости в госпитале).

Пример шаблона вопроса – *Count the number of hospital visits of patient* ***{patient_id}***.

2. **Вопросы, связанные со временем:** На основании опроса было выявлено, что реальные вопросы медицинских работников часто содержат вопросы насчет времени тех или иных событий. Для отражения этого в датасете время было классифицировано по нескольким типам выражений – единичным (например, дата визита в больницу, месяц и день события) и интервальным (например, с/до момента и в течение определенного времени). Затем типы временных выражений были объединены с шаблонами вопросов из первого пункта. Пример итогового вопроса с внедрением временных условий – *Is the heart rate of patient 2518 measured at* ***2105-12-31 12:00:00*** *less than the value measured at* ***2105-12-31 09:00:00****?*
3. **Надежные QA системы:** text-to-SQL системы должны предоставлять только точные ответы и отказываться от ответов на вопросы, на которые невозможно ответить на основании базы данных или без использования внешних знаний. Неотвечаемые вопросы включены в тестовый датасет EHRSQL бенчмарка для оценки способности воздержания от ответа text-to-SQL систем.

Статистики бенчмарка представлены в таблицах 14 и 15 . Данный text-to-SQL датасет представлен в SSP формате - на весь датасет есть одна база данных MIMIC-IV. В базе данных MIMIC-IV [34] 17 таблиц, а в среднем на каждую таблицу приходится по 6 атрибутов. 20% выборки в обучающем, валидационном и тестовом наборах данных составляют неотвечаемые вопросы. Само разбиение сконструировано таким образом, что несмотря на SSP постановку, в тестовой выборке есть как новые SQL шаблоны отвечаемых вопросов, так и обращения к новым элементам схемы, к которым ранее не обращались (хотя информация о наличии этих элементов в процессе обучения была). Таким образом, тестируется и композиционная генерализация моделей, и доменная.

Таблица 14 — Статистика датасета EHRSQL.

	Train	**Dev/Test**
Длина вопросов	15.5	15.3
Длина запросов	83.7	88.7
Количество таблиц на запрос	3	3
Количество аттрибутов на запрос	5	5
Количество пар	5124	1167

Таблица 15 — Статистика разбиений для EHRSQL. В тестовой (Test) и валидационной (Dev) выборках есть новые 34 шаблонов вопросов.

	Train	**Dev**	**Test**
Кол-во отвечаемых SQL шаблонов	100	134 (100 исходных + 34 новых)	134 (100 исходных + 34 новых)
Кол-во отвечаемых вопросов	4674	931	934
Количество неотвечаемых вопросов	450	232	233
Общее количество вопросов	5124	1163	1167

3.1.1 Метрика оценки EHRSQL

Для соревнования была сконструирована модифицированная метрика Execution Match. Ключевая модификация – добавление штрафа за некорректные генерации. Сама метрика называется **Reliability Score (RS)**.

$$RS_c(x) = \begin{cases} 1 & \text{if } x \in Q_{\text{ans}}; \quad g(x) = 1; Acc(x) = 1, \\ 0 & \text{if } x \in Q_{\text{ans}}; \quad g(x) = 0; \\ -c & \text{if } x \in Q_{\text{ans}}; \quad g(x) = 1; Acc(x) = 0, \\ -c & \text{if } x \in Q_{\text{una}}; \quad g(x) = 1, \\ 1 & \text{if } x \in Q_{\text{una}}; \quad g(x) = 0. \end{cases} \tag{3.1}$$

Функция $g(x) = 1$ указывает, что text-to-SQL система выбирает сгенерированный SQL как окончательный ответ, тогда как $g(x) = 0$ означает, что система воздерживается от ответа. Воздержание от ответа в работе будет обозначаться **NULL**. Показатель $Acc(x)$ указывает на корректность сгенерированного SQL, основанную на Execution Match метрике. Метрика надёжности (RS) включает пять различных случаев для расчёта оценки:

- Оценка 1 присваивается, если SQL корректно сгенерирован системой для отвечаемых вопросов.
- Оценка 0 присваивается, если система воздерживается от генерации SQL для отвечаемых вопросов.
- Оценка $-c$ присваивается, если система предсказывает некорректный SQL для отвечаемых вопросов.
- Оценка $-c$ присваивается, если система пытается предсказать SQL для неотвечаемых вопросов.
- Оценка 1 присваивается, если система точно определяет неотвечаемые вопросы, воздерживаясь от ответа.

Общая метрика RS рассчитывается как среднее арифметическое от оценок тестового датасета. Значение штрафа c выбирается в зависимости от требований надёжности модели. Штраф в размере 0 (RS_0) означает отсутствие наказания за некорректные предсказания – в таком случае метрика равна просто Execution Match. Штраф в размере 10 (RS_{10}) представляет собой умеренно строгий сценарий, а штраф в размере N (RS_N), где N соответствует размеру тестового датасета, является самым строгим сценарием, в котором даже одна ошибка перевешивает все правильные предсказания и воздержания. Максимально возможное значение RS составляет 100%, а минимальные возможные оценки варьируются в зависимости от штрафов: 0 для $c = 0$; -1000% для $c = 10$; $-100N\%$ для $c = N$. Основной метрикой оценки для соревнования был выбран RS_{10} – где каждые десять точных

предсказаний равнозначны одному ошибочному предсказанию. В данной главе лучшие text-to-SQL системы также будут выбираться по этой метрике.

3.2 Разработка text-to-SQL системы

В данной секции представлена text-to-SQL система, вошедшая в топ-5 решений на международном соревновании EHRSQL Shared Task[1]. В главе представлена система в общем и ее отдельные компоненты, которые были сконструированы и исследованы в процессе разработки такой системы для соревнования. Код системы доступен в репозитории[2].

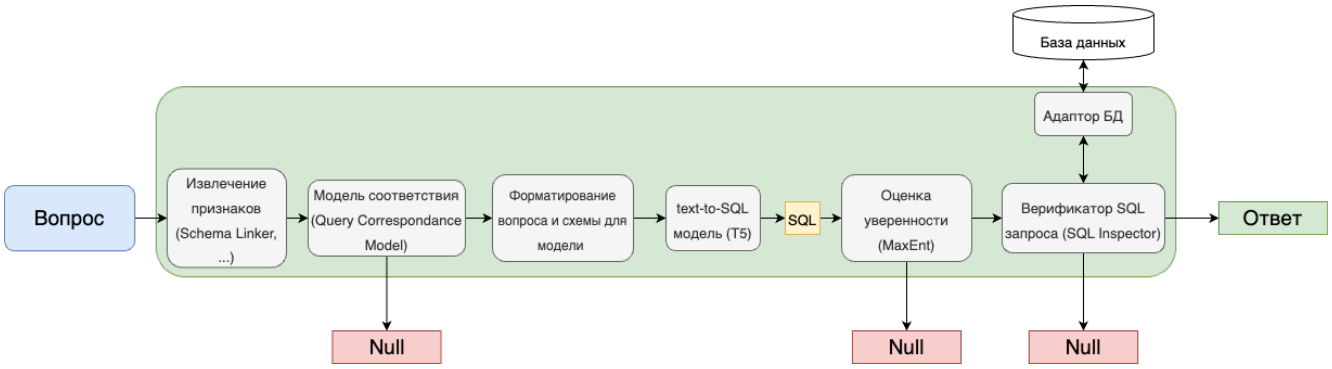

Рисунок 3.1 — Обзор системы. Пользовательский запрос вводится в систему text-to-SQL. Механизм извлечения признаков извлекает признаки для модели соответствия запроса. Модель соответствия запроса оценивает запрос по извлеченным признакам – подходит ли он системе. Если вопрос соответствует системе, он переходит в модель генерации text-to-SQL. Далее метод оценки неопределенности на основании метода максимальной энтропии определяет – является ли сгенерированный запрос корректным. Далее сгенерированный запрос передается инспектору результатов SQL, который проверяет возможность выполнения запроса и результат выполнения. Если результат выполнения запроса соответствует требованиям, результат исполнения возвращается пользователю.

[1]https://aclanthology.org/2024.clinicalnlp-1.62.pdf
[2]https://github.com/runnerup96/EHRSQL-text2sql-solution

Таблица 16 — Результаты экспериментов в соревновании EHRSQL. **1** - Финальное решения для соревнования, **2** - Результаты исследования после завершения соревнования.

	Название метода	RS_0	RS_5	RS_{10}	RS_N
1	Schema Linker + ChatGPT ICL 5-shot	0.550	-0.418	-1.386	-2254.5
2	Schema linker + T5-3B + SQL inspector + ChatGPT debugger	0.533	-0.272	-1.078	-18746.7
3	Schema linker + T5-3B + SQL Inspector	0.644	0.533	0.422	-2535.6
4	QCM + T5-3B + SQL Inspector	0.689	0.565	0.440	-2831.1
5	T5-3B +MaxEnt+ SQL Inspector	0.693	0.578	0.462	-2630.7
6	QCM + T5-3B + MaxEnt + SQL Inspector[1]	0.648	0.588	0.528	-1335.2
7	QCM + ChatGPT FT + MaxEnt + SQL Inspector[2]	**0.753**	**0.693**	**0.633**	**-1323.0**

3.2.1 Архитектура системы

text-to-SQL системы имеют две ключевые функции – генерация корректного SQL запроса и отвержение вопросов, на которые нельзя ответить. Таким образом, системы включают в себя компоненты, которые отвечают за реализацию этих функций.

Реализованная архитектура решения приведена на рисунке 3.1. Она состоит из последовательности синхронных компонент и либо возвращает сгенерированный SQL запрос, либо не дает ответа (возвращает **NULL**). Результаты системы, построенной из рассмотренных далее компонент, приведены в таблице 16.

3.2.2 Сопоставление вопроса и text-to-SQL системы

Первая компонента отвечает за определение релевантности входного запроса данной базе данных. Так как английский язык обладает низкой флективностью, данную проверку можно реализовать с помощью сопоставления вопроса и схемы/элементов базы данных с помощью строкового сопоставления (например, алгоритмом Левенштейна), как показано на рисунке 3.2. Данный алгоритм называется Schema Linker.

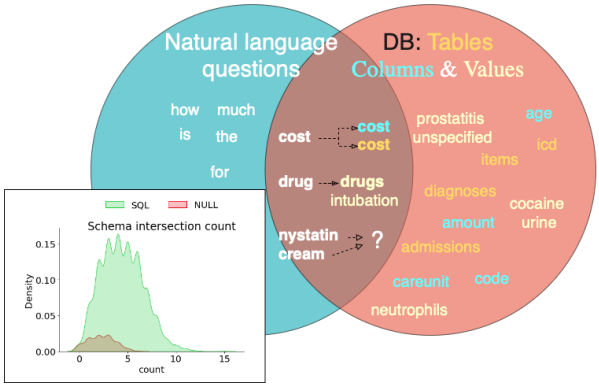

Рисунок 3.2 — Пример пересечения вопроса и контента базы данных – нормализированным нграммам вопроса “How much is the cost for the drug nystatin cream?” сопоставляются нормализованные элементы базы данных. NULL вопросы имеют гораздо меньшее количество таких пересечений по сравнению с SQL запросами.

Кроме факта пересечения Schema Linker, возможно извлечь другие признаки из обучающей выборки, которые помогут правильно сопоставить входной вопрос системе.

– Начало входных вопросов - в датасете EHRSQL была обнаружена закономерность, что пересечение начала обработанных NULL вопросов (первые два слова) и начала обработанных вопросов, для которых есть SQL составляет всего 8%. Таким образом, появился еще один булевый признак в дополнение к размеру пересечения – когда входной запрос имеет строковое пересечение с началом вопросов, у которых в обучающей выборке есть NULL.
– Длина запросов - также анализ данных показал, что средняя длина вопросов NULL составляет 11 ($\sigma = 3$) слов, а средняя длина вопросов

SQL — 15 ($\sigma = 6$). Таким образом длина вопроса – дополнительный численный признак модели.

Факт пересечения Schema Linker, бинарный признак начала вопроса и длина запроса – признаковое пространство модели **Query Correspondence Model**. Для взвешивания данных признаков и определения соответствия вопроса x_i системе была использована логистическая регрессия из формулы 3.2. Предсказание соответствия вопроса системе происходит при пороге 0.5.

$$p(y_i|x_i, \theta) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_i)}}, \hat{y}_i = [p(y_i|x_i, \theta) > 0.5]; \tag{3.2}$$

3.2.3 text-to-SQL модель

Для обучения text-to-SQL модели был проведен эксперимент с несколькими моделями в рамках соревнования. text-to-SQL модели (T5 и ChatGPT) обучались только на парах вопрос – SQL запрос.

Fine-tuning подход

В качестве модели для дообучения была выбрана T5-3B модель. Входные данные модели форматировалась в обычном text-to-SQL формате:

$$X = D : QS|t_1 : c_1^1, ...c_{n_1}^1|t_2 : c_1^2, ...c_{n_2}^2|t_N : c_1^N, ...c_{n_N}^N \tag{3.3}$$

Целевая переменная формировалась, как и в предыдущих экспериментах.

$$Y = D|QR \tag{3.4}$$

Параметры обучения приведены в Аппендиксе А.

In-context-learning подход

In-context-learning подход заключался в формировании затравки из инструкции, похожих пар вопрос – SQL запрос и вопроса Q, на которые требуется получить ответ, как описано в первой главе в секции описания методов. В качестве языковой модели использовался ChatGPT 4. В качестве примеров было добавлено 5 пар релевантных вопросов. Релевантность вопросов оценивается с помощью евклидового расстояния входного векторизованного вопроса $v_{q_i} \in \mathcal{R}^{1\times h}$, где h – размерность кодированного вектора, с индексом векторизованных вопросов (каждый из которых имеет SQL пару) обучающей выборки $[v_{q^1}, v_{q^2}, ..., v_{q^N}]$. Вопросы векторизуются с помощью SentenceBERT [3] [138] модели. Ближайшие вопросы добавляются в затравку как примеры генерации. В таблице результатов название компоненты **ChatGPT ICL 5-shot**.

Другой пример использования in-context-learning – использование ChatGPT для корректировки сгенерированного запроса. Данный подход позволил улучшить качество на бенчмарке BIRD [139]. Сгенерированный с помощью text-to-SQL модели SQL запрос подается на вход **ChatGPT Debugger** модуля, который инструктируется обратить внимание на сгенерированный SQL и исправить его в случае необходимости. Пример затравки для ChatGPT ICL и ChatGPT Debugger приведены в Аппендиксе Б.

3.2.4 Оценка уверенности запроса

Оценка неопределённости модели, посчитанная на основании выходов модели, может использоваться для определения уверенности модели в генерации.

Для решения EHRSQL Shared Task была выбрана оценка неопределенности методом максимальной энтропии (**MaxEnt**) [140]. Пусть авто-регрессионная модель задана уравнением:

[3] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

$$P(y|x,\theta) = \prod_{l=0}^{L} P(y_l|y_{<l},x,\theta)$$

где распределение P каждого следующего токена y_l зависит от предыдущих токенов последовательности $y_{<l} = y_0,...,y_{l-1}$ длины L. Идея оценки методом максимальной энтропии неопределённости модели заключается в том, что модель уверена в генерации всей последовательности настолько, насколько она уверена в самом неуверенном предсказании. С точки зрения энтропии, если распределение $P(y_l|y_{<l},x,\theta)$ – близко к равномерному, то вероятность предсказания корректного токена y_l невелика по сравнению с ярко-выраженной уверенностью в распределении. Формально метод максимальной энтропии выглядит так:

$$\begin{aligned} p_l &= P(y_l|y_{<l},x,\theta) \\ H(p_l) &= \sum_{v=0}^{|V|} p_v log(p_v) \\ u &= max(H_0,...,H_L) \end{aligned} \tag{3.5}$$

Для получения оценки уверенности c_i, максимальная оценка рассчитывается на валидационном датасете $(u_1,u_2,...,u_N)$ и каждая оценка u_i масштабируется в $[0;1]$:

$$\begin{aligned} u^{max} &= max(u_1,u_2,...,u_N) \\ c_i &= 1 - \frac{u_i}{u^{max}} \end{aligned} \tag{3.6}$$

На основании порога $t = 0.5$ сгенерированные запросы $c_i >= t$ – определяются как верные генерации, $c_i < t$ – как некорректные и отвергаются.

3.2.5 Верификация SQL запроса

В text-to-SQL задаче сгенерированный SQL запрос исполняется на целевой базе данных. В таком случае, факт исполнения запроса позволяет определить, корректен ли запрос с синтаксической точки зрения. Для этого был разработан

адаптер SQL базы данных (**SQL Inspector**), исполняющий сгенерированный запрос и фильтрующий результаты. Если запрос возвращает синтаксическую ошибку или ошибку, связанную со схемой базы данных – такой запрос не возвращается, а возвращается результат NULL. Также было проанализировано на валидационном датасете, что большинство NULL вопросов, на которые был сгенерирован запрос, возвращают пустое множество, None или 0 для агрегационных запросов – данное условие выполнения также определяет сгенерированные запросы как некорректные.

3.2.6 Результаты экспериментов

Финальные результаты экспериментов представлены в таблице 16. Как описано ранее, компоненты системы отвечают либо за генерацию запроса, либо за его отвержение. Для функции отвержения исследовался набор компонент в разных вариациях. Schema Linker, QCM, MaxEnt и SQL Inspector относятся к функционалу определения соответствия вопроса системе и проверке его корректности. ChatGPT и T5-3B – модели генерации SQL запроса. Первые **(1 и 2)** эксперименты (с ChatGPT компонентой) показывают не сильно высокое качество на специфичных медицинских данных. Замена ChatGPT на дообученную модель T5-3B показывает заметное улучшение в RS_{10} метрике в эксперименте **3**. Далее в трех последовательных экспериментах **(4,5,6)** были проверены разные комбинации функционала соответствия и проверки с помощью компонент QCM, MaxEnt и SQL Inspector – результаты привели к увеличению качества на **10%**. Стоит отметить, что просадка качества связана именно с генерацией некорректных SQL запросов, которые занижают итоговое качество. На валидационной выборке был проведен эксперимент с оценкой только NULL вопросов – построенная система отвергает неотвечаемые вопросы с качеством 98%. После соревнования было проведено исследование по замене дообучения T5-3B модели на дообучение ChatGPT модели с помощью API[4]. Использование ChatGPT fine-tune API (FT) позволило повысить итоговое качество еще на 10% по метрике RS_{10}. Решение победителя соревнования EHRSQL Shared Task также включало использование ChatGPT FT API.

[4]https://platform.openai.com/docs/guides/fine-tuning

3.3 Обсуждение результатов

В результате экспериментов разработана text-to-SQL система с возможностью отвержения неподходящих вопросов и некорректных генераций. Система построена из независимых компонент, каждая из которых решает отдельную задачу. Преимущества построенной системы:

- Комбинация независимых компонент позволяет независимо оптимизировать части и улучшать систему – также такой подход интерпретируем, так как позволяет объяснить результаты системы;
- Комбинация компонент проверки и верификации компонент на высоком уровне отвергает неотвечаемые вопросы, при этом на хорошем уровне возвращает пользователю корректные SQL запросы;
- Модель T5-3B может быть развернута на локальном сервере больницы, во избежание утечки конфиденциальных данных.

В тоже время у системы есть и недостатки:

- Каскадный эффект работы системы приводит к снижению итогового качества;
- Основной источник ошибок – недостаточная генерализации на новые шаблоны тестовой выборки. Решением этой проблемы является увеличение размера модели, внедрение индуктивного смещения алгоритма обучения и итеративное улучшение на основании обратной связи от пользователей;
- Для развертывания T5-3B модели требуются значительные вычислительные ресурсы, которые могут быть недоступны.

3.4 Выводы

В данной главе была разработана text-to-SQL система, которая генерирует ответ и отвергает вопросы, на которые она не может ответить. Система может быть развернута на локальных серверах, что очень важно для организаций с конфиденциальной информацией (таких как больницы, например). Ключевой источник ошибки данной системы – некорректные SQL запросы на отвечаемые

вопросы. Такие генерации ошибочны, более того при генерации таких запросов модели очень уверены в своей генерации, будто генерируют корректные запросы. Но это неправильное поведение надежных text-to-SQL систем, так как ожидается другое поведение:

1. Если вопрос относится к базе данных и модель уверена в его генерации, она, совместно с самим запросом, выдает высокую оценку уверенности;
2. Если модель не уверена в генерации запроса или вопрос не относится к используемой базе данных, она, совместно с самим запросом, выдает низкую оценку уверенности.

В данной главе представлен механизм поиска ошибок на основании оценки неопределенности – метод максимальной энтропии. В следующей главе будет проведено исследование – **возможно ли с помощью оценки неопределенности находить ошибочные генерации в условиях сдвига обучающей выборки?**

Глава 4. Поиск ошибок text-to-SQL моделей с помощью оценки неопределенности в условиях сдвига обучающей выборки

Text-to-SQL позволяет пользователям взаимодействовать с базами данных с помощью естественного языка, упрощая поиск и синтез информации. Несмотря на успех больших языковых моделей (LLM) в преобразовании вопросов на естественном языке в запросы SQL, их широкое внедрение ограничено двумя основными проблемами: достижением надежного обобщения на новые запросы и обеспечением высокого уровня интерпретации уверенности в прогнозах. Для эффективного использования моделей преобразования текста в SQL пользователи должны четко понимать возможности модели, в частности, диапазон вопросов, на которые она может точно ответить. В этом контексте способность к обобщению имеет решающее значение для обеспечения точной генерации SQL-запросов, в то время как интерпретация имеет важное значение для минимизации ложных срабатываний — случаев, когда модель генерирует неверные SQL-запросы, которые могут быть ошибочно восприняты как правильные.

В данной главе диссертации исследуется, как оценка неопределенности модели может быть использована для обнаружения ошибочных предсказаний text-to-SQL модели. Оценка неопределенности – оценка уверенности модели в предсказании (чем выше неопределенность, тем ниже уверенность). Эта оценка может быть рассчитана с помощью выходного softmax распределения целевой переменной или скрытых представлений нейронной сети. Text-to-SQL системы обычно состоят из двух ключевых компонент — компоненты генерации запроса и внешнего классификатора, определяющего, нужно ли возвращать сгенерированный запрос пользователю на основании оценки неопределенности. Изначально постановка задачи внешнего классификатора с возможностью отклонения принадлежит работе [141], которая исследует системы, где ошибка крайне нежелательна, а отсутствие ответа допустимо.

В качестве предмета исследования в главе рассмотрены сценарии сдвига обучающей выборки text-to-SQL моделей, так как именно при таких сценариях модели испытывают трудности с генерализацией. Примером таких трудностей является низкий уровень композиционной генерализации, когда модель не может предсказывать новые структуры SQL-запросов, невстречающиеся в

обучающем наборе, или низкое доменное обобщение, когда модель не может адаптироваться к новым элементам схемы или новым базам данных.

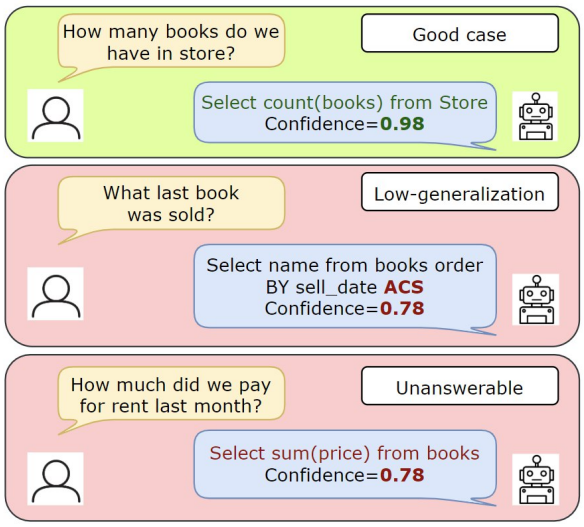

Рисунок 4.1 — Сценарии взаимодействия пользователя с text-to-SQL системой в контексте оценки неопределенности. Существует три ключевых сценария – детекция верных генераций (Good Case), определение неверной генерации (Low-generalization) и детекция неотвечамого запроса (Unanswerable).

Когда вопросы не могут быть отвечены с помощью базы данных или требуют внешних знаний, они называются неотвечаемыми. Поскольку text-to-SQL модели обучаются на парах вопрос-SQL, появление таких примеров на этапе предсказания является примером ковариационного сдвига распределения, и системы должны избегать генерации запроса на такой вопрос. Оба типа ошибок приводят к неправильным ответам, будь то невыполнимые запросы SQL или исполняемые запросы, которые возвращают ложные ответы. В использовании text-to-SQL систем, оценка неопределенности проявляется в трех сценариях взаимодействия, как показано на рисунке 4.1 – детекция верных генераций (Good Case), определение неверной генерации (Low-generalization) и детекция неотвечамого запроса (Unanswerable). Роль оценки неопределенности в этих сценариях следующая:

– Если вопрос относится к базе данных и модель уверена в его генерации, она, совместно с самим запросом, выдает низкую оценку неопределенности;
– Если модель не уверена в генерации запроса или вопрос не относится к используемой базе данных, она, совместно с самим запросом, выдает высокую оценку неопределенности.

Ключевой вопрос 4 главы — **возможно ли идентифицировать ошибочные предсказания языковых моделей в задаче text-to-SQL при сдвиге распределения с использованием оценки неопределенности?** В данной главе проведено исследование с точки зрения обнаружения ошибок и калиброванности text-to-SQL моделей.

4.1 Выборочный text-to-SQL

В данной главе рассматривается построение text-to-SQL систем с возможностью отклонения (**система выборочного text-to-SQL**) в условиях сдвига обучающей выборки. Допустим существует text-to-SQL модель $\mathcal{Y}$, которой на вход подается вопрос на естественном языке x совместно со схемой S. Модель $\mathcal{Y}$ генерирует SQL запрос $\tilde{y}$ и внешний классификатор $\mathcal{C}$ определяет, возвращать ли данный запрос пользователю или отправлять данный запрос в базу данных.

Оценка неопределенности text-to-SQL модель $\mathcal{Y}$ также возвращает скрытые представления нейронной сети (логиты), использованные для генерации SQL примера $\tilde{y}$. На основании логитов возможно рассчитать вероятностную оценку неопределенности модели. Пусть авторегрессионная sequence-to-sequence модель, параметризованная θ, с аргументами x генерирует последовательность $\tilde{y}$:

$$P(\tilde{y}_l|x,\theta) = \prod_{l=0}^{L} P(\tilde{y}_l|\tilde{y}_{<l},x,\theta)$$

где $P(\tilde{y}_l|x,\theta)$ — распределение вероятности по словарю для токена $\tilde{y}_l$ на основании x и предыдущих токенов последовательности $\tilde{y}_{<l} = \{\tilde{y}_0,...,\tilde{y}_{l-1}\}$. Одним из известных методов оценки неопределенности u для авторегрессионных моделей является среднее значение энтропии $\mathcal{H}$ по всей последовательности длины L [142]:

$$u = \frac{1}{L}\sum_{l=1}^{L} \mathcal{H}(\tilde{y}_l|\tilde{y}_{<l},x,\theta)$$

Однако в задаче семантического парсинга некоторые токены всегда появляются в определённом месте, и модель очень уверена в предсказании таких элементов (например, `SELECT` в начале каждого SQL выражения), таким образом, среднее значение может сделать итоговое значение слишком оптимистичным относительно истинной уверенности модели. Поэтому в данной работе используется модификация такой оценки неопределенности — **метод максимальной энтропии**:

$$\begin{aligned} p_l &= P(\tilde{y}_l | \tilde{y}_{<l}, x, \theta), \\ H(p_l) &= \sum_{v=0}^{|V|} p_v \log(p_v), \\ u &= \max(H_0, ..., H_L) \end{aligned} \tag{4.1}$$

Интуиция в такой модификации заключается в том, что модель настолько уверена в своем предсказании, насколько она уверена в самом неуверенном токене последовательности.

Для моделей, которые используются через API (например, ChatGPT), нет доступа ко всему распределению по словарю, поэтому используется метод нормализованной оценки вероятности [143]:

$$u = \frac{1}{L} \sum_{l=0}^{L} \log(p_l). \tag{4.2}$$

Выборочное предсказание text-to-SQL модель возвращает пару $(\tilde{y}_i, u_i)$, где $\tilde{y}_i$ — сгенерированный SQL-запрос, а u_i — оценка неопределенности модели. С помощью порога $\gamma \in \mathbb{R}$ внешний классификатор $\mathcal{C}(u_i)$ пропускает сгенерированный SQL-запрос далее (возвращает пользователю или исполняет на базе данных), либо отвергает его. Для обучения и тестирования внешнего классификатора используется выборка D_{tst}, которая включает в себя out-of-distribution данные относительно обучающей выборки D_{tr}. D_{tst} разделяется по схеме **i.i.d** на 2 части: выборка D_{known}, которая включает в себя известные o.o.d. примеры, и выборка D_{unk}, которая включает неизвестные o.o.d. примеры. Выборка D_{known} используется для обучения внешнего классификатора и метода калибровки, а выборка D_{unk} для оценки [144].

Ключевыми характеристиками качества выборочного классификатора $\mathcal{C}$ являются покрытие и риск. Покрытие — это доля выборки D_{unk}, на

которой классификатор делает корректные предсказания, а риск — это доля D_{unk}, на которой классификатор ошибается. Для разработки системы выборочного text-to-SQL, в частности выборочного классификатора или метода калибровки, целевая переменная формируется методом инвертированного Execution Match, как в формуле 4.3: если результат истинного запроса g_i не равен результату предсказанного $\tilde{y}_i$, такой результат считается ошибочным и является положительным классом для обучения выборочного классификатора:

$$\hat{y}_i = \begin{cases} 0 & \text{if EX}(g_i) == \text{EX}(\tilde{y}_i) \\ 1 & \text{if EX}(g_i) \neq \text{EX}(\tilde{y}_i) \end{cases} \tag{4.3}$$

С увеличением порога $\boldsymbol{\gamma}$ внешнего классификатора $\mathcal{C}$ значения покрытия и риска также изменяются. Оценить качество покрытия и риска внешнего классификатора для выбранного значения $\boldsymbol{\gamma}$ можно с помощью метрик полноты и точности. При оценке на D_{unk}, TP — количество верных определений некорректной генерации, TN — количество верных определений корректной генерации, FP — количество неверных определений корректных генераций, помеченных как некорректные, FN — количество неверных определений некорректных генераций, помеченных как корректные.

$$\text{Точность} = \frac{TP}{TP + FP}, \quad \text{Полнота} = \frac{TP}{TP + FN} \tag{4.4}$$

В контексте выборочного text-to-SQL покрытие соответствует способности внешнего классификатора находить некорректные генерации, а риск — это пропорция случаев, когда корректные запросы классифицируются как некорректные. Таким образом, покрытие можно оценить с помощью метрики полноты, а риск — с помощью False Discovery Rate (1 - Точность) — пропорции ложных срабатываний внешнего классификатора. Для оценки качества внешнего классификатора по всем значениям порога $\boldsymbol{\gamma}$ используется интегральная оценка Area Under Curve (AUC).

4.2 Исследования

Для исследования выборочного text-to-SQL будут использованы 4 модели. Это модели архитектуры кодировщик-декодировщик T5-large и T5-3B, декодировщик Llama3-8B (обученная в формате SFT и LoRA), а также in-context-learning модель DAIL-SQL на базе ChatGPT. T5 модели и модель Llama3-8B были дообучены на парах вопрос x и схема базы данных S – запрос y. В решении DAIL-SQL дополнительно добавляются релевантные примеры, как описано в первой главе в части методов. Каждое обучение модели проводилось от 3 до 5 раз. Параметры обучения и предсказания моделей описаны в Аппендиксе А. Вид затравки для решения DAIL-SQL на базе ChatGPT описан в Аппендиксе Б. Модели в связке с выборочным классификатором оценивались на 4 разбиениях данных, рассмотренных в предыдущих главах:

- **PAUQ XSP** — разбиение по базам данных таким образом, что в обучающей выборке датасета присутствует один набор баз данных D_{tr}, а в тестовой — другой D_{ts}, так что $D_{tr} \cap D_{ts} = \varnothing$.
- **Template SSP** — композиционное разбиение для датасета PAUQ, где обучающие и тестовые выборки нацелены на один и тот же набор баз данных, но SQL-шаблоны запросов не пересекаются между обучением и тестированием.
- **TSL SSP** — композиционное разбиение, где обучающие и тестовые выборки нацелены на один и тот же набор баз данных, но шаблоны запросов в обучающем наборе данных в среднем короче, чем в тестовой выборке.
- **EHRSQL** — оригинальное разбиение данных SSP с EHRSQL Shared Task, рассмотренное в третьей главе. В дополнение к новым шаблонам в тестовой выборке и новым контекстам использования элементов схемы в вопросах, в тестовой выборке также появляются неотвечаемые вопросы.

Статистики для PAUQ XSP представлены в таблице 6, статистики для Template SSP и TSL SSP — в таблице 7, а статистики для EHRSQL — в таблице 15. На основании данных моделей и датасетов далее в главе будет произведено три исследования:

– Оценка качества выборочных text-to-SQL систем в условиях сдвига обучающей выборки;
– Оценка калиброванности text-to-SQL моделей в условиях сдвига обучающей выборки;
– Интерпретация уверенности внешнего классификатора через характеристики сгенерированного запроса.

4.2.1 Оценка качества выборочных text-to-SQL систем

Первое исследование состоит в построении выборочной text-to-SQL системы на основании text-to-SQL модели и выборочного классификатора, и оценки системы в условиях сдвига обучающей выборки. В исследовании оцениваются 3 внешних классификатора для предсказания на основании оценки u_i – логистическая регрессия, смесь гауссиан и механический подбор порога.

Для разностороннего исследования будут даны ответы на 4 вопроса:

1. **RQ-1:** Какой из внешних классификаторов обладает лучшим компромиссом между покрытием и риском?
2. **RQ-2:** Как влияет внешний классификатор на качество работы text-to-SQL модели?
3. **RQ-3:** Какое из исследуемых разбиений является наиболее сложным для выборочной text-to-SQL системы?
4. **RQ-4:** При наличии в тестовой выборке неотвечаемых вопросов, какой вид ошибочных генераций внешний классификатор (случай слабой генерализации или генерация на неотвечаемый вопрос) лучше распознает?

Внешний классификатор Внешний классификатор используется для классификации сгенерированного запроса на основании оценки неопределенности u_i модели. Выборка D_{known} используется для обучения, а выборка D_{unk} для оценки. Для этого в работе используются 3 подхода:

– **Логистическая регрессия.** Обучение логистической регрессии, параметризованной θ, методом максимального правдоподобия и

определение сгенерированного SQL запроса как ошибочного, при вероятности более 0.5:

$$p(y_i|u_i,\theta) = \frac{1}{1+e^{-(\theta_0+\theta_1 u_i)}}, \hat{y}_i = [p(y_i|u_i,\theta) > 0.5]; \tag{4.5}$$

– **Смесь гауссиан.** Набор оценок u_i рассматривается как смесь двух нормальных распределений. Пусть оценки неопределенности некорректно сгенерированных запросов относятся к одному распределению $z_0 \sim N(u_i|\mu_0,\sigma_0)$, а оценки неопределенности корректно сгенерированных запросов относятся к распределению $z_1 \sim N(u_i|\mu_1,\sigma_1)$. Также заданы априорные вероятности π_0 и π_1 выбрать то или иное распределение (изначально распределения равновероятны). Параметры $\mu_0,\sigma_0,\mu_1,\sigma_1,\pi_0,\pi_1$ подбираются на выборке D_{known} EM (Expectation-Maximization) алгоритмом. На этапе предсказания для точки u_i выбирается наиболее вероятное распределение:

$$\hat{y}_i = \arg\max_z (\pi_z \mathcal{N}(u_i \mid \mu_k, \sigma_k)) \tag{4.6}$$

– **Механический подбор порога.** Подбор заключается в кумулятивном накоплении корректных генераций до прекращения накопления [140]. Алгоритм выглядит так:

1. Отсортировать примеры на основании уверенности (так как уверенность основана на максимальной энтропии, то меньшая энтропия соответствует большей уверенности);
2. Корректной генерации присваивается $+1$ балл, некорректной – -1 балл;
3. Двигаясь от наибольшей уверенности к наименьшей, кумулятивно увеличивать на балл, если встречена корректная генерация SQL, и уменьшать, если встречена некорректная генерация;
4. Выбрать ту оценку неопределенности как порог γ, где кумулятивная оценка перестает увеличиваться;
5. Применить γ к D_{unk} и после инвертировать предсказания $\hat{y}_i$ (в работе положительный класс — это ошибочная генерация, а не корректная).

Для сравнения работы методов и ответа на **RQ-1** была выбрана оценка F_β:

$$F_\beta = \frac{(1+\beta^2)\cdot \mathrm{TP}}{(1+\beta^2)\cdot \mathrm{TP} + \mathrm{FP} + \beta^2 \cdot \mathrm{FN}}, \tag{4.7}$$

В работе $\beta = 3$, так как детекция некорректных генераций в три раза важнее, нежели ошибочная классификация верных генераций (выбрано эмпирически). На рисунке 4.2 были построены тепловые карты оценок $F_{\beta=3}$ для каждой модели и каждого датасета. На рисунке видно, что смесь гауссиан в роли внешнего классификатора в целом даёт лучший компромисс между покрытием и риском. В отдельных моментах заметно ухудшение по сравнению с логистической регрессией и подбором порога. Ухудшение качества для моделей T5-large, T5-3B и DAIL-SQL наблюдается по сравнению с методом подбора порога на разбиении Template SSP, а модели Llama лучше работают с логистической регрессией на EHRSQL датасете. **Далее для экспериментов внешнего классификатора будет использоваться внешний классификатор на основе смеси гауссиан.**

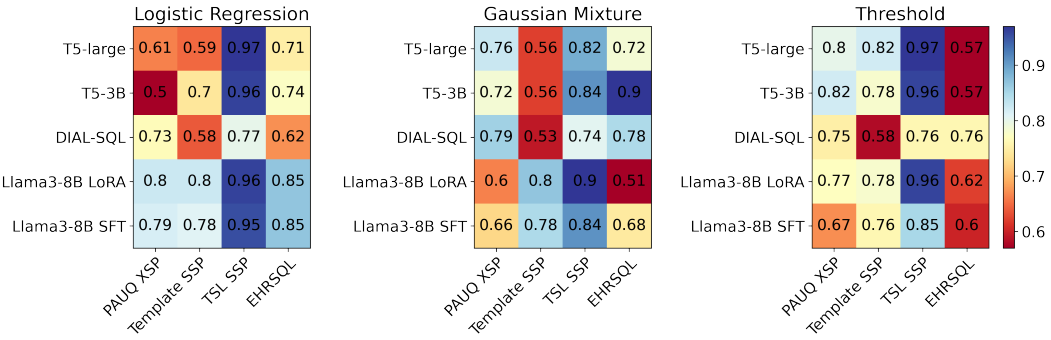

Рисунок 4.2 — Тепловая карта оценок $\mathrm{F}_{\beta=3}$ для сравнения методов – логистическая регрессия (Logistic Regression), смесь гауссиан (Gaussian Mixture) и механический подбор порога (Threshold).

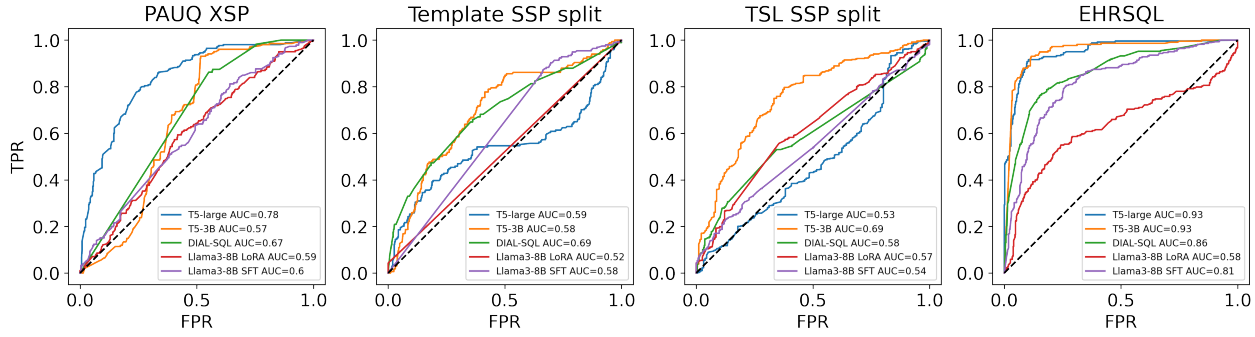

Рисунок 4.3 — ROC-кривые для выбранных text-to-SQL моделей с внешним классификатором смеси гауссиан.

Таблица 17 — Классификация предсказаний text-to-SQL моделей методом Gaussian Mixture на различных датасетах. EX - Execution Match. Полнота - пропорция найденных неверных предсказаний по оценке неопределённости. FDR - пропорция ложноположительных срабатываний алгоритма поиска ошибки на верных предсказаниях. Финальный EX - Execution Match с учетом ложноположительных срабатываний алгоритма поиска ошибки.

		EX	Полнота	FDR	Финальный EX	σ Result EX
T5-large	PAUQ XSP	0.634	0.795	0.218	0.415	0.097
	Template SSP	0.69	0.624	0.461	0.229	0.24
	TSL SSP	0.243	0.828	0.201	0.043	0.016
	EHRSQL	0.687	0.705	0.034	0.654	0.007
T5-3B	PAUQ XSP	0.709	0.816	0.444	0.265	0.157
	Template SSP	0.738	0.608	0.335	0.404	0.228
	TSL SSP	0.282	0.852	0.17	0.112	0.021
	EHRSQL	0.712	0.919	0.086	0.626	0.027
DIAL-SQL	PAUQ XSP	0.765	0.982	0.577	0.188	$\leqslant 0.01$
	Template SSP	0.86	0.868	0.846	0.014	$\leqslant 0.01$
	TSL SSP	0.664	0.854	0.636	0.028	$\leqslant 0.01$
	EHRSQL	0.542	0.793	0.105	0.437	$\leqslant 0.01$
Llama3-8B LoRA	PAUQ XSP	0.717	0.66	0.373	0.344	0.068
	Template SSP	0.673	0.958	0.673	0	$\leqslant 0.01$
	TSL SSP	0.296	0.929	0.265	0.032	0.026
	EHRSQL	0.643	0.546	0.403	0.239	0.202
Llama3-8B SFT	PAUQ XSP	0.731	0.756	0.449	0.282	0.002
	Template SSP	0.697	0.907	0.524	0.173	0.009
	TSL SSP	0.361	0.88	0.342	0.019	0.016
	EHRSQL	0.64	0.688	0.131	0.509	0.042

Для ответа на **RQ-2** и **RQ-3** была построена таблица 17. Таблица показывает, как работает система выборочного text-to-SQL со смесью гауссиан в роли внешнего классификатора. Характеристики покрытия и риска в колонках **Полнота** и **FDR**. Лучший результат показывают модели на EHRSQL датасете. Модель T5-3B показывает покрытие более 90%, а риск – менее 10%. Таким образом, итоговое качество системы **Финальный EX** для этого разбиения для большинства моделей самое лучшее. С точки зрения покрытия, наибольшую проблему представляет **Template SSP** разбиение – внешний классификатор на этом разбиении имеет очень высокую FDR оценку, что ведет к низкому итоговому качеству генерации SQL всей системы. Модели, обученные на TSL SSP разбиении изначально имеют низкое качество Execution Match, хотя покрытие и риск внешнего классификатора у них лучше, чем у Template SSP. Также заметно, что внешние классификаторы для T5 моделей проявляют более низкий риск по сравнению с ChatGPT и Llama моделями. **Такое поведение тесно связано с калиброванностью модели, что будет исследовано далее.**

Для оценки качества моделей на разбиениях при разных значениях порога внешнего классификатора смеси гауссиан использовалась ROC-кривая. Как показано на рисунке 4.3, T5 модели показывают лучшее качество по сравнению с другими моделями, особенно на EHRSQL датасете, где они достигают наибольшего значения AUC = 0.93. Исследования показывают, что T5 модели более приспособлены для построения систем выборочных text-to-SQL систем, нежели Llama и ChatGPT модели из-за более интерпретируемой оценки неопределенности.

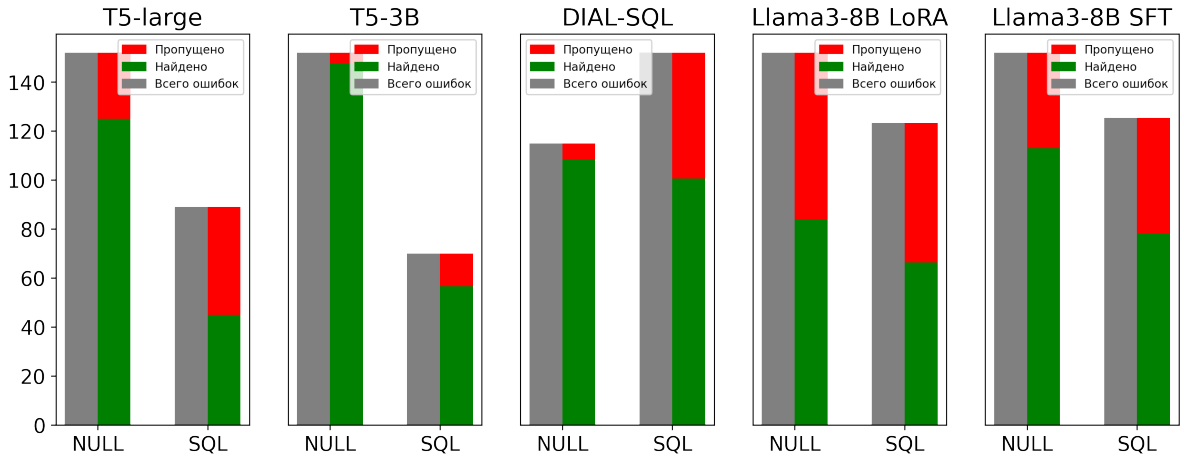

Рисунок 4.4 — Разбиение по типам ошибок EHRSQL набора данных. NULL – SQL, сгенерированные на неотвечаемые вопросы, SQL – ошибочные SQL запросы. Пропущено – количество пропущенных некорректных SQL запросов, Найдено – количество найденных некорректных SQL запросов.

Для ответа на **RQ-4** был построен график 4.4 – распределение двух типов найденных ошибочных генераций для каждой модели на EHRSQL с помощью внешнего классификатора смеси гауссиан. Каждая модель с большей вероятностью находит некорректные генерации на неотвечаемые вопросы. Можно предположить, что модель более уверена в генерации некорректного запроса на отвечаемый вопрос, нежели в генерации запроса на неотвечаемый вопрос. В подтверждение прошлых выводов, модель T5-3B показывает наилучшее качество по определению ошибочных генераций.

Выводы из исследования выборочного text-to-SQL следующие:

– **Вывод 1 (RQ-1, RQ-2, RQ-3)**: среди рассмотренных внешних классификаторов лучший компромисс между риском и покрытием на всех разбиениях показала смесь гауссиан. Смесь гауссиан улучшает качество text-to-SQL системы, находя ошибки text-to-SQL модели и не вводя пользователя в заблуждение. Лучшим примером работы системы text-to-SQL с выборочным классификатором является EHRSQL датасет. Например, выборочный классификатор модели T5-3B определяет 91% некорректных генераций с низким ложно-положительным процентом срабатываний, менее 10% – таким образом сохраняя итоговый Execution Match на высоком уровне. Но на Template SSP внешний классификатор критично ухудшает работу

text-to-SQL модели, обладая высоким уровнем ложноположительных срабатываний.

– **Вывод 2 (RQ-4)**: выборочный классификатор с большей вероятностью определяет генерации на неотвечаемые вопросы, нежели некорректные генерации SQL запросов. Таким образом можно предположить, что модель более уверена в генерации некорректного запроса на отвечаемый вопрос, нежели в генерации запроса на неотвечаемый вопрос.

4.2.2 Оценка калиброванности text-to-SQL моделей

В первом исследовании было эмпирически установлено, что связка модели T5-3B и внешнего классификатора позволяет на высоком уровне выявлять ошибки генерации и при этом иметь низкий уровень ложно-положительных срабатываний. Одна из причин – изначальная хорошая калиброванность оценки неопределенности этой модели. В этой части будет проведено исследование оценки неопределённости моделей, обученных на разных сдвигах обучающей выборки. Ключевые вопросы **RQ-5** исследования:

– **RQ-5**: Как влияют методы калибровки на итоговую калиброванность модели?

– **RQ-6**: Каков компромисс между калиброванностью модели и её качеством (Execution Match)?

Калибровка уверенности моделей машинного обучения Хорошая калиброванность модели машинного обучения определяется близостью вероятности классификации к пропорции верных классификаций. На примере если предсказывается корректная вероятность класса 1, то прогноз *объект* x_i *принадлежит классу 1 с вероятностью* 0.75 *для модели* a будет сбываться в 0.75 случаях:

$$P(\hat{y}_i = 1 | a(x_i) = p) = p$$

Обычно глубокие нейронные сети возвращают некалиброванные оценки неопределенности. Для калибровки оценки неопределенности существуют методы калибровки моделей машинного обучения.

– **Калибровка Платта** [145] – обучение логистической регрессии, как в формуле 4.5, где независимая переменная оценка неопределенности u_i, а $\hat{y}_i$ – метка бинарной классификации 4.3.

– **Изотоническая регрессия** [146] – построение кусочно-постоянной константной функции вида $g_m(u_i) = \theta_m$ для трансформации оценок неопределённости методом минимизации квадратичной функции разности. Так как g_m - кусочно-постоянная константная функция, в процессе обучения данного метода калибровки решается следующая задача оптимизации:

$$\begin{aligned} \min_{M,\theta,a} \quad & \sum_{m=1}^{M}\sum_{i=1}^{N} \mathbb{1}(a_m \leqslant u_i < a_{m+1})(\hat{y}_i - \theta_m)^2 \\ \text{s.t.} \quad & 0 \leqslant a_1 \leqslant a_2 \leqslant ... \leqslant a_{M+1} = 1, \\ & \theta_1 \leqslant \theta_2 \leqslant ... \leqslant \theta_M \end{aligned} \tag{4.8}$$

где M - количество интервалов; $a_1, ..., a_{M+1}$ - границы интервалов; $\theta_0, ..., \theta_M$ значения функции g_m. В процессе обучения изотоническая регрессия оптимизирует размеры столбцов гистограммы для калибровки вероятностей модели.

– **MinMax нормализация** – нормализация на максимальное значение D_{known} каждой оценки неопределенности u_i из интервала $[0; +\inf]$ в интервал $[0; 1]$ методом MinMax нормализации:

$$u_i^c = \frac{(u_i - min(u_i))}{max(u_i) - min(u_i)} \tag{4.9}$$

Для оценки калибровки используется две метрики – Expected Calibration Error и Brier Score. Метрика Expected Calibration Error (ECE) представлена в формуле 4.10:

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\text{acc}(B_m) - \text{conf}(B_m)\right| \tag{4.10}$$

где M — это количество интервалов (или "бинов") вероятностей, на которые разбивается диапазон предсказанных вероятностей, B_m — это бин m, содержащий все примеры, чьи предсказанные вероятности попадают в этот интервал, $|B_m|$ — количество примеров в бине B_m, n — общее количество примеров, $\text{acc}(B_m)$ — это точность в бине B_m, то есть доля правильно

предсказанных классов среди всех примеров в этом бине, $\mathrm{conf}(B_m)$ — это средняя предсказанная вероятность (уверенность) для всех примеров в бине B_m. Expected Calibration Error (ECE) измеряет расхождение между точностью и уверенностью модели. Модель хорошо откалибрована, если её уверенность соответствует фактической точности. Чем меньше ECE, тем лучше.

Метрика Brier Score представлена в формуле 4.11:

$$\mathrm{BS} = \frac{1}{n}\sum_{i=1}^{n}(p_i - \hat{y}_i)^2 \tag{4.11}$$

где n — общее количество примеров, p_i — предсказанная вероятность для примера i, $\hat{y}_i$ — истинная метка класса для примера i (1 для положительного класса и 0 для отрицательного класса). Brier Score (BS) измеряет среднеквадратичную ошибку между предсказанной вероятностью и фактическим результатом. Если модель предсказывает вероятность 1 для положительного класса и вероятность 0 для отрицательного, и эти предсказания верны, Brier Score будет 0 (идеальное предсказание). Если предсказания вероятности модели сильно расходятся с истинными значениями, значение Brier Score увеличивается. Согласно работе [147], Brier Score более интерпретируем для сравнения нескольких моделей, поэтому далее методы калибровки будут сравниваться по Brier Score.

	MinMax	Платта	Изотоническая регрессия
T5-large	0.2	0.121	**0.117**
T5-3B	0.17	0.108	**0.106**
Llama3-8B LoRA	0.22	0.216	**0.199**
Llama3-8B SFT	0.21	0.19	**0.175**
DIAL-SQL	0.239	0.16	**0.152**

Таблица 18 — Сравнение трех методов калибровки оценкой Брайера, усредненное по каждому разбиению данных.

Таблица 18 представляет сравнение методов калибровки с использованием Brier Score, усредненной по всем разделениям данных. Изотоническая регрессия последовательно превосходит как нормализацию MinMax, так и калибровку Платта для всех моделей. Примечательно, что по мере увеличения размера

моделей T5 производительность изотонической регрессии также улучшается. Для T5-large, T5-3B и DIAL-SQL преимущество изотонической калибровки над нормализацией MinMax является значительным. Напротив, как Llama3-8B LoRA, так и SFT демонстрируют меньшую разницу между изотонической калибровкой и MinMax, что позволяет предположить, что выбор метода калибровки для этих моделей при выбранной оценке неопределенности не оказывает сильного влияния на калиброванность.

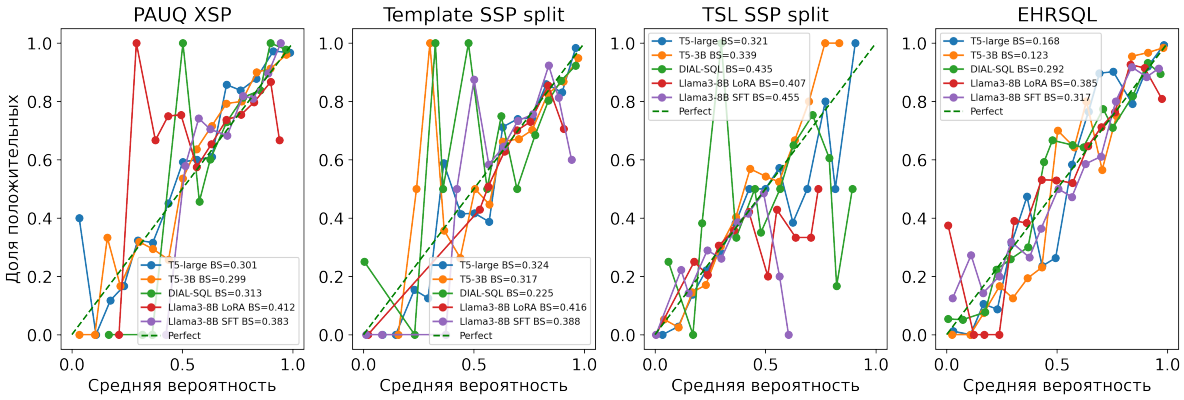

Рисунок 4.5 — Калибровка модели изотонической регрессией на каждом разбиении данных.

На рисунке 4.5 представлен результат калибровки методом изотонической регрессии по всем моделям и по всем разделениям (усредненным по 3 независимым запускам). Очевидно, что датасеты со сдвигом обучающей выборки (PAUQ XSP, Template SSP и TSL SSP) не приводят к калиброванным моделям. В наборе данных EHRSQL, где нет сильного сдвига, модели демонстрируют хороший уровень калиброванности. В то же время модели Llama не очень хорошо калибруются даже при применении методов калибровки. Однако при сравнении Llama LoRA и Llama SFT, версия SFT демонстрирует лучшую калиброванность.

Рисунок 4.6 показывает сравнение калибровочных кривых разных методов калибровки (Платта и Изотоническая регрессия) и нормализованной оценки неопределённости (MinMax) на модели T5-3B на PAUQ XPS и EHRSQL. Как показано на рисунке, исходная нормализованная оценка неопределенности не является изначально калиброванной, но методы калибровки позволяют улучшить эту оценку.

В заключении исследования, на рисунке 4.7 представлен компромисс между Execution Match моделей и их калиброванностью. Видно, что модели

архитектуры кодировщик-декодировщик более калиброваны, чем модели Llama. Но модели Llama обладают более высоким качеством на некоторых разбиениях, например PAUQ XSP. С увеличением размера модели (T5-large → T5-3B) модель не только лучше генерализирует, но и обладает лучшим уровнем калиброванности.

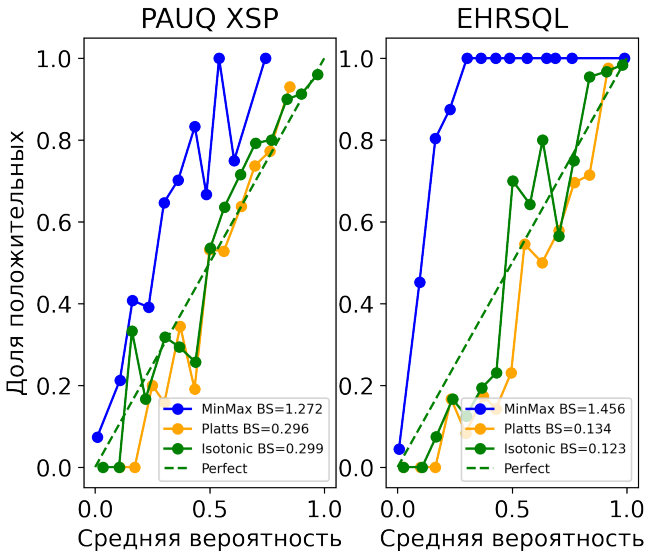

Рисунок 4.6 — Сравнение калибровочных кривых разных методов калибровки (Платта и Изотоническая регрессия) и нормализованной оценки неопределённости (MinMax) на модели T5-3B на PAUQ XPS и EHRSQL.

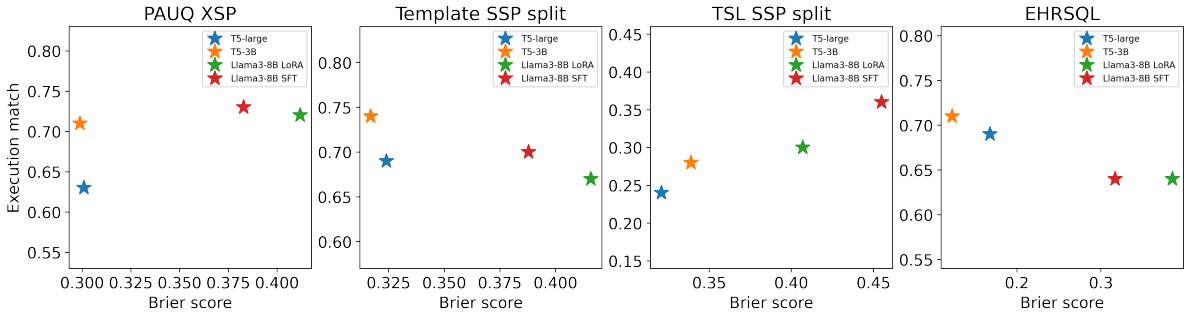

Рисунок 4.7 — Компромисс между калиброванностью (Brier score) и Execution Match text-to-SQL моделей после калибровки методом изотонической регрессии.

– **Вывод 3 (RQ-5):** Результаты показывают, что изначальная оценка неопределенности MinMax не калибрована; тем не менее, изотоническая регрессия позволяет значительно улучшить калибровку моделей.
– **Вывод 4 (RQ-6):** Модели архитектуры кодировщик-декодировщик лучше калиброваны, чем модели Llama. Более того, с увеличением

размера модели (T5-large → T5-3B) модель не только лучше генерализирует, но и обладает лучшим уровнем калиброванности.

4.2.3 Интерпретация уверенности внешнего классификатора

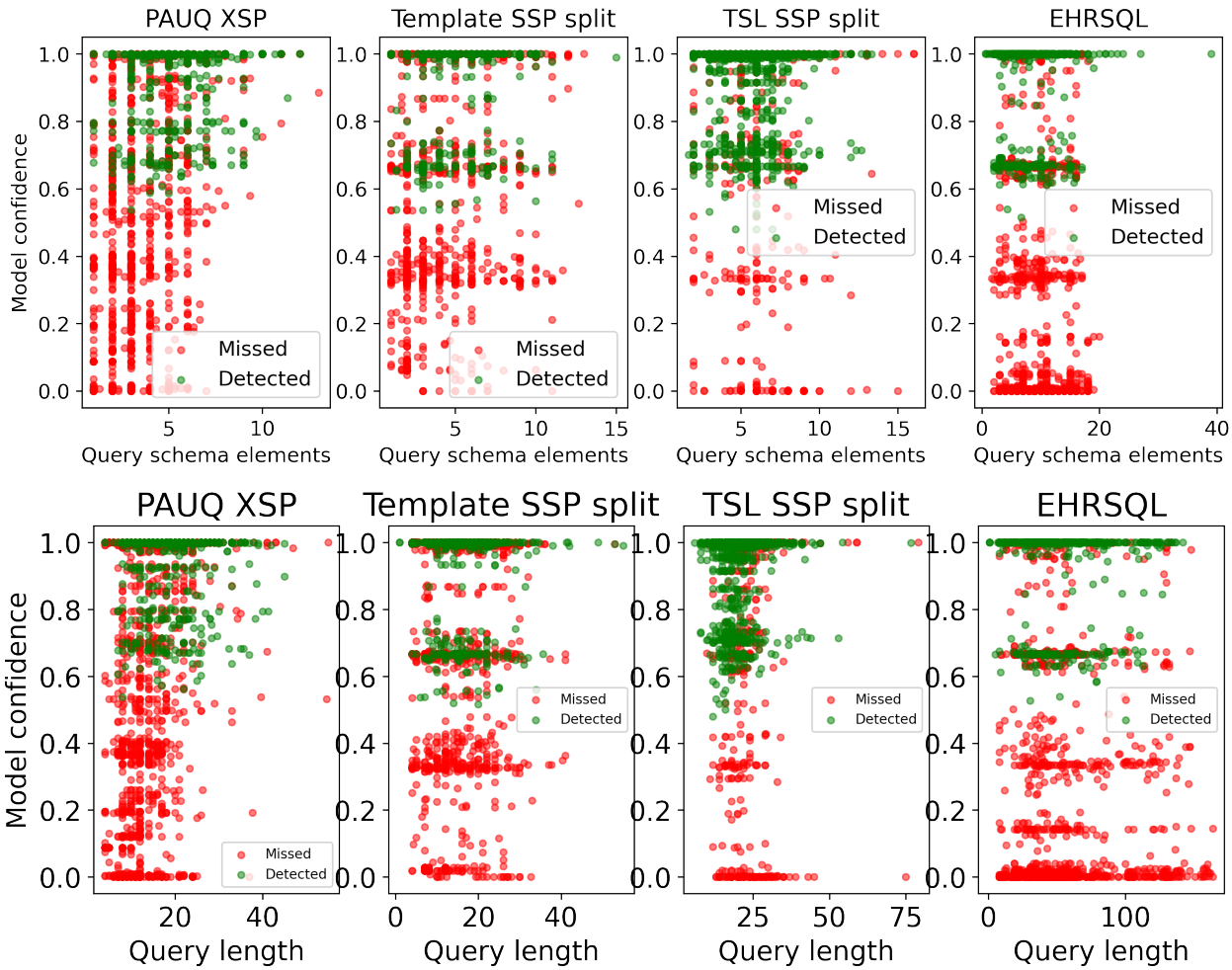

Рисунок 4.8 — Графики сравнения характеристик запроса (кол-во элементов схемы (Query Schema elements) и длина запроса (Query length)) с уверенностью внешнего классификатора смеси гауссиан на различных разбиениях: PAUQ XSP, Template SSP, TSL SSP, и EHRSQL семплированных из уверенности выборочных text-to-SQL систем с моделями T5-large и T5-3B.

Модели формата кодировщик–декодировщик показали себя лучше остальных в связке с внешним классификатором смеси гауссиан. Вопрос данного исследования **RQ-7** – Связана ли вероятность отклонения внешним классификатором с характеристикой сложности сгенерированного запроса?

В качестве характеристик выбраны те, которые соответствуют сложности запроса – длина запроса и количество элементов схемы в запросе. Для ответа на данный вопрос использована уверенность модели смеси гауссиан для неправильно сгенерированных примеров модели T5. Были выбраны именно T5 модели, так как предыдущие эксперименты показали превосходство этих моделей с точки зрения калиброванности над другими.

Графики зависимости, которые представлены на рисунке 4.8, были построены для каждого разделения данных, чтобы проанализировать взаимосвязь между этими характеристиками запроса и уверенностью внешнего классификатора. Изначальная гипотеза исследования заключалась в том, что более сложные неправильные запросы, характеризованные большей длиной или увеличенным числом элементов схемы, будут соответствовать более низкой уверенности, что приводит к тому, что внешний классификатор неверно определяет их как неправильные генерации. Однако, как показывают графики, уверенность выборочного классификатора не имеет отношения к сложности запроса.

Вопреки поставленой гипотезе не было определено пропорциональное снижение уверенности модели для некорректных запросов по мере увеличения сложности запросов. Во всех разделениях, даже для наиболее калиброванных моделей, нет четкой связи между уверенностью классификатора и сложностью запроса по выбранным характеристикам.

4.3 Выводы

В четвертой главе диссертации проведено исследование оценки неопределенности больших языковых моделей в задаче text-to-SQL. Для этого были разработаны системы выборочного text-to-SQL – text-to-SQL модели и внешнего классификатора, который на основании оценки неопределенности детектирует некорректные генерации. Главными выводами исследования четвертой главы являются:

– При полном сдвиге, будь то доменный или композиционный, классификатор сгенерированных SQL запросов на основании

энтропийной оценки неопределённости ошибочно определяет много верных генераций, как ошибочные;

- В то же время модели, обученные на датасете EHRSQL, где доменный и композиционный сдвиг не так сильно выражен, классификатор обнаружения ошибок работает более эффективно чем на датасете, где ярко выражен композиционный (Template SSP) или доменный сдвиг (PAUQ XSP);
- Но б**о**льшее количество обнаруженных ошибок в EHRSQL связано с нахождением SQL, сгенерированных на неотвечаемые вопросы. Поиск ошибочно сгенерированных запросов остается на неудовлетворительном уровне для всех моделей, кроме T5-3B - самой калиброванной модели на ERHSQL датасете среди всех остальных;
- Изначальная энтропийная оценка неопределенности моделей не калибрована, но методы калибровки машинного обучения могут сделать оценку более интерпретируемой;
- Модели архитектуры кодировщик-декодировщик более калиброваны, нежели модели архитекутры декодировщик в задаче text-to-SQL.

Заключение

Разработка text-to-SQL систем — важный этап в создании систем принятия решений. В данной работе было проведено исследование на тему определения закономерностей поведения предобученных больших языковых моделей в условиях сдвига обучающей выборки. Результаты и выводы данной работы позволяют определить область применимости text-to-SQL решений в условиях производства и оптимально встроить данное решение в рабочие процессы.

Основными результатами и выводами диссертации являются:

1. Создан первый русский text-to-SQL датасет PAUQ, который является переработанной и улучшенной версией англоязычного датасета Spider. В рамках разработки этого датасета были определены ключевые правила адаптации text-to-SQL датасетов, предложена оптимизация сбора такого датасета и методика, позволяющая адаптировать англоязычные наборы данных быстрее и дешевле.
2. Проведено исследование способности современных больших языковых моделей к генерализации. Несмотря на то, что большие языковые модели обладают хорошим качеством в условиях равномерного распределения данных между обучением и тестированием, при использовании этих моделей в условиях сдвига обучающей выборки, качество таких моделей сильно снижается, так как доменная и композиционная генерализация предобученных языковых моделей на самом деле не находится на высоком уровне. Поэтому очень важно осторожно создавать обучающую выборку без значимых видимых сдвигов, чтобы она покрывала большинство используемых сценариев, и не ожидать, что модель сможет эффективно генерализировать за пределами данных сценариев.
3. Проведено исследование калиброванности современных больших языковых моделей на основании энтропийной оценки неопределенности. В условиях сдвига обучающей выборки, оценки неопределенности языковых моделей сложно интерпретировать – на основании данной оценки сложно сделать вывод о корректности генерации. Лучше всего

модели поддаются интерпретации в условиях отсутствия сильного сдвига обучающей выборки.

4. Разработана качественная и надежная text-to-SQL система для узкоспециализированного медицинского домена, упрощающая взаимодействие с данными пациентов для работников госпиталя, вошедшая в топ-5 решений на международном соревновании EHRSQL Shared Task.

Перспективными направлениями дальнейших исследований являются работы по генерализации и интерпретации больших языковых моделей. С точки зрения генерализации, необходимо определить те случаи, когда модели фундаментально не способны обобщать на новые данные на основании данных для обучения. С точки зрения интерпретации надо определить факторы, отвечающие за те или иные решения, принимаемые нейронной сетью или влияющие на такие решения. Дальнейшее исследование этих аспектов нейронных сетей позволит построить качественные и надежные системы искусственного интеллекта.

В заключении работы я хочу выразить благодарность моему научному руководителю, **Тутубалиной Елене**, за мудрое и отзывчивое научное руководство; моей жене, **Сомовой Анне**, за постоянную поддержку и понимание на протяжении всего обучения в аспирантуре; моим родителям, **Сомовой Татьяне** и **Нехаеву Дмитрию**, за то, что вдохновили меня поступить на физтех и поддерживали на протяжении всей учёбы. Также хочу поблагодарить **компанию SberDevices** за возможность участвовать в разработке продукта для миллионов пользователей, и **институт AIRI** за поддержку моих исследований.

Список сокращений и условных обозначений

AUC Area Under Curve, интегральная оценка классификации, позволяющая оценить качество для каждого порога классификатора.

BERT Bidirectional Encoder Representations from Transformers, кодировщик на основе архитектуры нейронной сети трансформер.

bi-LSTM Bidirectional Long Short Term Memory, двунаправленная LSTM.

ChatGPT большая предобученная модель архитектуры GPT от компании OpenAI.

GPT Generative Pretrained Decoder, декодировщик на основе архитектуры нейронной сети трансформер.

LSTM Long Short Term Memory, нейронная сеть с долгой краткосрочной памятью.

T5 Text-To-Text Transfer Transformer, модель архитектуры кодировщик-декодировщик на основе архитектуры нейронной сети трансформер.

Словарь терминов

бенчмарк: набор данных, используемый для оценки качества модели машинного обучения.

датасет: набор данных состоящий из пар (x, y), где x – независимая переменная, объект для обучения, а y – целевая метка объекта для обучения .

дообучение: второстепенный этап обучения языковых моделей на специфичную задачу.

эмбеддинг: векторное представление объекта в пространстве размерности $\mathbb{R}^n$.

F1-мера: метрика качества классификации, вычисляется как среднее гармоническое между точностью и полнотой.

multi-head attention: компонента глубоких нейронных сетей, отвечающая за обогощения эмбеддинга информацией эмбеддингов из контекста последовательности.

Полнота: вычисляется как отношение истинно-положительных объектов к общему количеству известных положительных объектов.

предобучение: первоначальный этап обучения больших языковых моделей методом языкового моделирования.

семплирование: стратегия выбора примеров из выборки.

Софтмакс (softmax): обобщение логистической функции для многомерного случая, вычисляется по формуле:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}},$$

где $[x_1, ...x_n]$ - эмбеддинг нейронной сети размерностью $\mathbb{R}^n$.

Точность: вычисляется как отношение истинно-положительных релевантных объектов к общему количеству определенных системой положительных объектов.

токен: текстовая единица, представляющая собой слово целиком или символьную N-грамму.

токенизация: процесс разбиения текста на токены.

трансформер: архитектура нейронной сети, состоящая из компонент multi-head attention и полносвязных слоев.

Список литературы

[1] Jonathan Berant, Andrew Chou, Roy Frostig и Percy Liang. — «Semantic parsing on freebase from question-answer pairs». — В: *Proceedings of the 2013 conference on empirical methods in natural language processing*. — 2013, — С. 1533—1544.

[2] Qingqing Cai и Alexander Yates. — «Large-scale semantic parsing via schema matching and lexicon extension». — В: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. — 2013, — С. 423—433.

[3] Tom Kwiatkowski, Eunsol Choi, Yoav Artzi и Luke Zettlemoyer. — «Scaling semantic parsers with on-the-fly ontology matching». — В: *Proceedings of the 2013 conference on empirical methods in natural language processing*. — 2013, — С. 1545—1556.

[4] Ngonga Ngomo. — «9th challenge on question answering over linked data (QALD-9)». — В: *language* 7.1 (2018), с. 58—64.

[5] Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, Fei Huang и Yongbin Li. — *A Survey on Text-to-SQL Parsing: Concepts, Methods, and Future Directions*. — 2022. — arXiv: 2208.13629 [cs.CL]. — (Дата обр. 17.02.2024).

[6] Pengcheng Yin и Graham Neubig. — «A Syntactic Neural Model for General-Purpose Code Generation». — В: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. — Association for Computational Linguistics. 2017.

[7] Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan и Yu Su. — «Beyond IID: three levels of generalization for question answering on knowledge bases». — В: *Proceedings of the Web Conference 2021*. — 2021, — С. 3477—3488.

[8] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman и др. — «Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task». — В: *Proceedings of the 2018*

Conference on Empirical Methods in Natural Language Processing. — Association for Computational Linguistics. 2018.

[9] Yushi Wang, Jonathan Berant и Percy Liang. — «Building a Semantic Parser Overnight». — В: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers).* — Под ред. Chengqing Zong и Michael Strube. — Beijing, China: Association for Computational Linguistics, июль 2015, — С. 1332—1342. — URL: https://aclanthology.org/P15-1129.

[10] Christopher M Bishop и Nasser M Nasrabadi. — *Pattern recognition and machine learning.* — Т. 4. — 4. — Springer, 2006, — С. 1—58.

[11] Catherine Finegan-Dollak, Jonathan K Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang и Dragomir Radev. — «Improving Text-to-SQL Evaluation Methodology». — В: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* — 2018, — С. 351—360.

[12] Brenden Lake и Marco Baroni. — «Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks». — В: *35th International Conference on Machine Learning, ICML 2018.* — International Machine Learning Society (IMLS). 2018, — С. 4487—4499.

[13] Dieuwke Hupkes, Verna Dankers, Mathijs Mul и Elia Bruni. — «Compositionality Decomposed: How do Neural Networks Generalise? (Extended Abstract)». — В: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20.* — Под ред. Christian Bessiere. — Journal track. — International Joint Conferences on Artificial Intelligence Organization, июль 2020, — С. 5065—5069. — URL: https://doi.org/10.24963/ijcai.2020/708 (дата обр. 17.02.2024).

[14] Terence Parsons. — *Formal Philosophy: Selected Papers of Richard Montague.* — 1975.

[15] Theo MV Janssen и Barbara H Partee. — «Compositionality». — В: *Handbook of logic and language.* — Elsevier, 1997, — С. 417—473.

[16] Peter Shaw, Ming-Wei Chang, Panupong Pasupat и Kristina Toutanova. — «Compositional Generalization and Natural Language Variation: Can a Semantic Parsing Approach Handle Both?» — В: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. — Под ред. Chengqing Zong, Fei Xia, Wenjie Li и Roberto Navigli. — Online: Association for Computational Linguistics, авг. 2021, — С. 922—938. — URL: https://aclanthology.org/2021.acl-long.75 (дата обр. 12.01.2024).

[17] Yongjin Yang, Sihyeon Kim, SangMook Kim, Gyubok Lee, Se-Young Yun и Edward Choi. — «Towards Unbiased Evaluation of Detecting Unanswerable Questions in EHRSQL». — В: *ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

[18] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher и др. — «A survey of uncertainty in deep neural networks». — В: *Artificial Intelligence Review* 56.Suppl 1 (2023), с. 1513—1589.

[19] Oleg Somov, Daria Bakshandaeva, Ekaterina Dmitrieva, Vera Davydova и Elena Tutubalina. — «PAUQ: Text-to-SQL in Russian». — В: *Findings of the Association for Computational Linguistics: EMNLP 2022*. — Под ред. Yoav Goldberg, Zornitsa Kozareva и Yue Zhang. — Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, дек. 2022, — С. 2355—2376. — URL: https://aclanthology.org/2022.findings-emnlp.175.

[20] Oleg Somov и Elena Tutubalina. — «Shifted PAUQ: Distribution shift in text-to-SQL». — В: *Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP*. — Под ред. Dieuwke Hupkes, Verna Dankers, Khuyagbaatar Batsuren и Koustuv Sinha. — Singapore: Association for Computational Linguistics, дек. 2023, — С. 214—220. — URL: https://aclanthology.org/2023.genbench-1.18.

[21] Oleg Somov, Alexey Dontsov и Elena Tutubalina. — «AIRI NLP Team at EHRSQL 2024 Shared Task: T5 and Logistic Regression to the Rescue». — В: *Proceedings of the 6th Clinical Natural Language Processing Workshop*. — Под ред. Tristan Naumann, Asma Ben Abacha,

Steven Bethard, Kirk Roberts и Danielle Bitterman. — Mexico City, Mexico: Association for Computational Linguistics, июнь 2024, — С. 431—438. — URL: https://aclanthology.org/2024.clinicalnlp-1.43.

[22] Олег Сомов. — «Многозадачное обучение для улучшения генерализации в задаче генерации структурированных запросов». — В: *Труды МФТИ* 16.2(62) (2024), с. 25—31. — URL: https://mipt.ru/science/trudy/62.

[23] Олег Сомов. — «Data-driven question answering system based on knowledge graphs». — В: *Dialogue 2020* (2020). — URL: https://www.dialog-21.ru/media/4912/somovod.pdf.

[24] Aleksei S. Krylov и Oleg D. Somov. — «Table-to-Text Generation With Pretrained Diffusion Models». — В: *IEEE Access* 12 (2024), с. 110517—110525.

[25] Chandra Khatri, Anu Venkatesh, Behnam Hedayatnia, Raefer Gabriel, Ashwin Ram и Rohit Prasad. — «Alexa Prize — State of the Art in Conversational AI». — В: *AI Magazine* 39.3 (сент. 2018), с. 40—55. — URL: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2810.

[26] JH Rai и PO Bagde. — «Building chatbots: A guide to frameworks and platforms». — В: *AIP Conference Proceedings*. — Т. 3180. — 1. — AIP Publishing. 2024.

[27] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev и Percy Liang. — «SQuAD: 100,000+ Questions for Machine Comprehension of Text». — В: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. — Под ред. Jian Su, Kevin Duh и Xavier Carreras. — Austin, Texas: Association for Computational Linguistics, нояб. 2016, — С. 2383—2392. — URL: https://aclanthology.org/D16-1264.

[28] Richard D Hipp. — *SQLite*. — Вер. 3.31.1. — 2020. — URL: https://www.sqlite.org/index.html (дата обр. 18.03.2024).

[29] Victor Zhong, Caiming Xiong и Richard Socher. — «Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning». — В: *CoRR* abs/1709.00103 (2017).

[30] Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo и др. — «Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls». — В: *Advances in Neural Information Processing Systems* 36 (2024).

[31] Ruoxi Sun, Sercan Ö Arik, Alex Muzio, Lesly Miculicich, Satya Gundabathula, Pengcheng Yin, Hanjun Dai, Hootan Nakhost, Rajarishi Sinha, Zifeng Wang и др. — «SQL-PaLM: Improved Large Language Model Adaptation for Text-to-SQL (extended)». — В: *arXiv preprint arXiv:2306.00739* (2023). — (Дата обр. 12.05.2024).

[32] Niklas Wretblad, Fredrik Gordh Riseby, Rahul Biswas, Amin Ahmadi и Oskar Holmström. — «Understanding the Effects of Noise in Text-to-SQL: An Examination of the BIRD-Bench Benchmark». — В: *arXiv preprint arXiv:2402.12243* (2024). — (Дата обр. 07.04.2024).

[33] Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim и Edward Choi. — «Ehrsql: A practical text-to-sql benchmark for electronic health records». — В: *Advances in Neural Information Processing Systems* 35 (2022), с. 15589—15601.

[34] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi и Roger G Mark. — «MIMIC-III, a freely accessible critical care database». — В: *Scientific data* 3.1 (2016), с. 1—9.

[35] Tom J Pollard, Alistair EW Johnson, Jesse D Raffa, Leo A Celi, Roger G Mark и Omar Badawi. — «The eICU Collaborative Research Database, a freely available multi-center database for critical care research». — В: *Scientific data* 5.1 (2018), с. 1—13.

[36] Charles T. Hemphill, John J. Godfrey и George R. Doddington. — «The ATIS Spoken Language Systems Pilot Corpus». — В: *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*. — 1990. — URL: https://aclanthology.org/H90-1021.

[37] Lappoon R. Tang и Raymond J. Mooney. — «Automated construction of database interfaces: integrating statistical and relational learning for semantic parsing». — В: *Proceedings of the 2000 Joint SIGDAT Conference on*

Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13. — EMNLP '00. — Hong Kong: Association for Computational Linguistics, 2000, — С. 133—141. — URL: https://doi.org/10.3115/1117794.1117811.

[38] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy и Luke Zettlemoyer. — «Learning a Neural Semantic Parser from User Feedback». — В: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* — Под ред. Regina Barzilay и Min-Yen Kan. — Vancouver, Canada: Association for Computational Linguistics, июль 2017, — С. 963—973. — URL: https://aclanthology.org/P17-1089.

[39] Fei Li и H. V. Jagadish. — «Constructing an interactive natural language interface for relational databases». — В: *Proc. VLDB Endow.* 8.1 (сент. 2014), с. 73—84. — URL: https://doi.org/10.14778/2735461.2735468.

[40] Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig и Thomas Dillig. — «SQLizer: query synthesis from natural language». — В: *Proc. ACM Program. Lang.* 1.OOPSLA (окт. 2017). — URL: https://doi.org/10.1145/3133887.

[41] Haoyang Li, Jing Zhang, Cuiping Li и Hong Chen. — «RESDSQL: Decoupling Schema Linking and Skeleton Parsing for Text-to-SQL». — В: *AAAI.* — 2023.

[42] Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov и Matthew Richardson. — «RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers». — В: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.* — Под ред. Dan Jurafsky, Joyce Chai, Natalie Schluter и Joel Tetreault. — Online: Association for Computational Linguistics, июль 2020, — С. 7567—7578. — URL: https://aclanthology.org/2020.acl-main.677.

[43] Xi Victoria Lin, Richard Socher и Caiming Xiong. — «Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing». — В: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, November 16-20, 2020.* — 2020.

[44] Haoyang Li, Jing Zhang, Cuiping Li и Hong Chen. — «Resdsql: Decoupling schema linking and skeleton parsing for text-to-sql». — В: *Proceedings of the AAAI Conference on Artificial Intelligence*. — Т. 37. — 11. — 2023, — С. 13067—13075.

[45] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever и Dario Amodei. — «Language Models are Few-Shot Learners». — В: *Advances in Neural Information Processing Systems*. — Под ред. H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan и H. Lin. — Т. 33. — Curran Associates, Inc., 2020, — С. 1877—1901. — URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[46] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li и Peter J Liu. — «Exploring the limits of transfer learning with a unified text-to-text transformer». — В: *Journal of machine learning research* 21.140 (2020), с. 1—67.

[47] Wonseok Hwang, Jinyeong Yim, Seunghyun Park и Minjoon Seo. — «A comprehensive exploration on wikisql with table-aware word contextualization». — В: *arXiv preprint arXiv:1902.01069* (2019). — (Дата обр. 11.04.2024).

[48] Xiaojun Xu, Chang Liu и Dawn Song. — «SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning». — В: *arXiv preprint arXiv:1711.04436* (2017). — (Дата обр. 11.04.2024).

[49] Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang и Dragomir Radev. — «TypeSQL: Knowledge-Based Type-Aware Neural Text-to-SQL Generation». — В: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. — 2018, — С. 588—594.

[50] Kevin Lin, Ben Bogin, Mark Neumann, Jonathan Berant и Matt Gardner. — «Grammar-based neural text-to-sql generation». — В: *arXiv preprint arXiv:1905.13326* (2019). — (Дата обр. 18.01.2024).

[51] Ohad Rubin и Jonathan Berant. — «SmBoP: Semi-autoregressive Bottom-up Semantic Parsing». — В: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. — Под ред. Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty и Yichao Zhou. — Online: Association for Computational Linguistics, июнь 2021, — С. 311—324. — URL: https://aclanthology.org/2021.naacl-main.29.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser и Illia Polosukhin. — «Attention is All you Need». — В: *Advances in Neural Information Processing Systems*. — Под ред. I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan и R. Garnett. — Т. 30. — Curran Associates, Inc., 2017. — URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[53] Torsten Scholak, Nathan Schucher и Dzmitry Bahdanau. — «PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models». — В: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. — Под ред. Marie-Francine Moens, Xuanjing Huang, Lucia Specia и Scott Wen-tau Yih. — Online и Punta Cana, Dominican Republic: Association for Computational Linguistics, нояб. 2021, — С. 9895—9901. — URL: https://aclanthology.org/2021.emnlp-main.779.

[54] Jacob Devlin, Ming-Wei Chang, Kenton Lee и Kristina Toutanova. — «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». — В: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. — Под ред. Jill Burstein, Christy Doran и Thamar Solorio. — Minneapolis, Minnesota: Association for Computational Linguistics, июнь 2019, — С. 4171—4186. — URL: https://aclanthology.org/N19-1423.

[55] Sepp Hochreiter и Jürgen Schmidhuber. — «Long Short-term Memory». — В: *Neural computation* 9 (дек. 1997), с. 1735—80.

[56] Abigail See, Peter J Liu и Christopher D Manning. — «Get To The Point: Summarization with Pointer-Generator Networks». — В: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. — Association for Computational Linguistics. 2017.

[57] Peter Shaw, Jakob Uszkoreit и Ashish Vaswani. — «Self-Attention with Relative Position Representations». — В: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. — Под ред. Marilyn Walker, Heng Ji и Amanda Stent. — New Orleans, Louisiana: Association for Computational Linguistics, июнь 2018, — С. 464—468. — URL: https://aclanthology.org/N18-2074.

[58] Pengcheng Yin и Graham Neubig. — «A Syntactic Neural Model for General-Purpose Code Generation». — В: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. — Под ред. Regina Barzilay и Min-Yen Kan. — Vancouver, Canada: Association for Computational Linguistics, июль 2017, — С. 440—450. — URL: https://aclanthology.org/P17-1041.

[59] Philip Gage. — «A new algorithm for data compression». — В: *The C Users Journal archive* 12 (1994), с. 23—38. — URL: https://api.semanticscholar.org/CorpusID:59804030.

[60] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He и Piotr Dollár. — «Focal Loss for Dense Object Detection». — В: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.2 (2020), с. 318—327.

[61] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding и Jingren Zhou. — «Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation». — В: *CoRR* abs/2308.15363 (2023).

[62] Qingkai Min, Yuefeng Shi и Yue Zhang. — «A Pilot Study for Chinese SQL Semantic Parsing». — В: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. — 2019, — С. 3643—3649.

[63] Marcelo Archanjo José и Fabio Gagliardi Cozman. — «mRAT-SQL+GAP: Portuguese Text-to-SQL Transformer». — В: *Intelligent Systems: 10th Brazilian Conference, BRACIS 2021, Virtual Event, November 29 – December 3, 2021, Proceedings, Part II.* — Berlin, Heidelberg: Springer-Verlag, 2021, — С. 511—525. — URL: https://doi.org/10.1007/978-3-030-91699-2_35.

[64] Hyeonji Kim, Byeong-Hoon So, Wook-Shin Han и Hongrae Lee. — «Natural language to SQL: where are we today?» — В: *Proc. VLDB Endow.* 13.10 (июнь 2020), с. 1737—1750. — URL: https://doi.org/10.14778/3401960.3401970.

[65] Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan и Tat-Seng Chua. — «Re-examining the Role of Schema Linking in Text-to-SQL». — В: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).* — Под ред. Bonnie Webber, Trevor Cohn, Yulan He и Yang Liu. — Online: Association for Computational Linguistics, нояб. 2020, — С. 6943—6954. — URL: https://aclanthology.org/2020.emnlp-main.564.

[66] Bailin Wang, Mirella Lapata и Ivan Titov. — «Meta-Learning for Domain Generalization in Semantic Parsing». — В: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* — Под ред. Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty и Yichao Zhou. — Online: Association for Computational Linguistics, июнь 2021, — С. 366—379. — URL: https://aclanthology.org/2021.naacl-main.33.

[67] Aparna Elangovan, Jiayuan He и Karin Verspoor. — «Memorization vs. Generalization : Quantifying Data Leakage in NLP Performance Evaluation». — В: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume.* — Под ред. Paola Merlo, Jorg Tiedemann и Reut Tsarfaty. — Online: Association for Computational Linguistics, апр. 2021, — С. 1325—1335. — URL: https://aclanthology.org/2021.eacl-main.113.

[68] Robert Kirk, Amy Zhang, Edward Grefenstette и Tim Rocktäschel. — «A Survey of Zero-shot Generalisation in Deep Reinforcement Learning». — В:

Journal of Artificial Intelligence Research 76 (янв. 2023), с. 201—264. — URL: http://dx.doi.org/10.1613/jair.1.14174.

[69] Jürgen Schmidhuber. — *Towards compositional learning with dynamic neural networks*. — Technical Report FKI-129-90. — Institut für Informatik, Technische Universität München, 1990.

[70] Francis C. K. Wong и William S-Y Wang. — «Generalisation towards Combinatorial Productivity in Language Acquisition by Simple Recurrent Networks». — В: *2007 International Conference on Integration of Knowledge Intensive Multi-Agent Systems*. — 2007, — С. 139—144.

[71] Dieuwke Hupkes, Mario Giulianelli, Verna Dankers, Mikel Artetxe, Yanai Elazar, Tiago Pimentel, Christos Christodoulopoulos, Karim Lasri, Naomi Saphra, Arabella Sinclair, Dennis Ulmer, Florian Schottmann, Khuyagbaatar Batsuren, Kaiser Sun, Koustuv Sinha, Leila Khalatbari, Maria Ryskina, Rita Frieske, Ryan Cotterell и Zhijing Jin. — «A taxonomy and review of generalization research in NLP». — В: *Nature Machine Intelligence* 5.10 (окт. 2023), с. 1161—1174. — URL: https://doi.org/10.1038/s42256-023-00729-y.

[72] Mitchell P. Marcus, Beatrice Santorini и Mary Ann Marcinkiewicz. — «Building a Large Annotated Corpus of English: The Penn Treebank». — В: *Computational Linguistics* 19.2 (1993). Под ред. Julia Hirschberg, с. 313—330. — URL: https://aclanthology.org/J93-2004.

[73] Michael John Collins. — «A new statistical parser based on bigram lexical dependencies». — В: *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*. — ACL '96. — Santa Cruz, California: Association for Computational Linguistics, 1996, — С. 184—191. — URL: https://doi.org/10.3115/981863.981888.

[74] Slav Petrov и Dan Klein. — «Improved Inference for Unlexicalized Parsing». — В: *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. — Под ред. Candace Sidner, Tanja Schultz, Matthew Stone и ChengXiang Zhai. — Rochester, New York: Association for Computational Linguistics, апр. 2007, — С. 404—411. — URL: https://aclanthology.org/N07-1051.

[75] Khalil Mrini, Franck Dernoncourt, Quan Hung Tran, Trung Bui, Walter Chang и Ndapa Nakashole. — «Rethinking Self-Attention: Towards Interpretability in Neural Parsing». — В: *Findings of the Association for Computational Linguistics: EMNLP 2020*. — Под ред. Trevor Cohn, Yulan He и Yang Liu. — Online: Association for Computational Linguistics, нояб. 2020, — С. 731—742. — URL: https://aclanthology.org/2020.findings-emnlp.65.

[76] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy и Samuel Bowman. — «GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding». — В: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. — Под ред. Tal Linzen, Grzegorz Chrupała и Afra Alishahi. — Brussels, Belgium: Association for Computational Linguistics, нояб. 2018, — С. 353—355. — URL: https://aclanthology.org/W18-5446.

[77] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer и Veselin Stoyanov. — *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. — 2019. — arXiv: 1907.11692 [cs.CL]. — URL: https://arxiv.org/abs/1907.11692 (дата обр. 22.06.2024).

[78] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann и др. — «Palm: Scaling language modeling with pathways». — В: *Journal of Machine Learning Research* 24.240 (2023), с. 1—113.

[79] Su Lin Blodgett, Lisa Green и Brendan O'Connor. — «Demographic Dialectal Variation in Social Media: A Case Study of African-American English». — В: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. — Под ред. Jian Su, Kevin Duh и Xavier Carreras. — Austin, Texas: Association for Computational Linguistics, нояб. 2016, — С. 1119—1130. — URL: https://aclanthology.org/D16-1120.

[80] Temuulen Khishigsuren, Gabor Bella, Khuyagbaatar Batsuren, Abed Alhakim Ali
Nandu Chandran Nair, Amarsanaa Ganbold, Hadi Khalilia, Yamini Chandrashekar
Fausto Giunchiglia и др. — «Using Linguistic Typology to Enrich

Multilingual Lexicons: the Case of Lexical Gaps in Kinship». — В: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. — European Language Resources Association. 2022.

[81] Najoung Kim и Tal Linzen. — «COGS: A Compositional Generalization Challenge Based on Semantic Interpretation». — В: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. — Под ред. Bonnie Webber, Trevor Cohn, Yulan He и Yang Liu. — Online: Association for Computational Linguistics, нояб. 2020, — С. 9087—9105. — URL: https://aclanthology.org/2020.emnlp-main.731.

[82] Brenden Lake и Marco Baroni. — «Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks». — В: *35th International Conference on Machine Learning, ICML 2018*. — Под ред. Jennifer Dy и Andreas Krause. — 35th International Conference on Machine Learning, ICML 2018. — Publisher Copyright: © Copyright 2018 by the author(s).; 35th International Conference on Machine Learning, ICML 2018 ; Conference date: 10-07-2018 Through 15-07-2018. — International Machine Learning Society (IMLS), 2018, — С. 4487—4499.

[83] Tom McCoy, Ellie Pavlick и Tal Linzen. — «Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference». — В: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. — Под ред. Anna Korhonen, David Traum и Lluis Màrquez. — Florence, Italy: Association for Computational Linguistics, июль 2019, — С. 3428—3448. — URL: https://aclanthology.org/P19-1334.

[84] Barbara Plank. — *What to do about non-standard (or non-canonical) language in NLP*. — 2016. — arXiv: 1608.07836 [cs.CL]. — URL: https://arxiv.org/abs/1608.07836 (дата обр. 18.06.2024).

[85] Yasaman Razeghi, Robert L Logan IV, Matt Gardner и Sameer Singh. — «Impact of Pretraining Term Frequencies on Few-Shot Numerical Reasoning». — В: *Findings of the Association for Computational Linguistics: EMNLP 2022*. — 2022, — С. 840—854.

[86] Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar,

Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang и Ben Zhou. — «Evaluating Models' Local Decision Boundaries via Contrast Sets». — В: *Findings of the Association for Computational Linguistics: EMNLP 2020*. — Под ред. Trevor Cohn, Yulan He и Yang Liu. — Online: Association for Computational Linguistics, нояб. 2020, — С. 1307—1323. — URL: https://aclanthology.org/2020.findings-emnlp.117.

[87] Divyansh Kaushik, Eduard Hovy и Zachary Lipton. — «Learning The Difference That Makes A Difference With Counterfactually-Augmented Data». — В: *International Conference on Learning Representations*. — 2020. — URL: https://openreview.net/forum?id=Sklgs0NFvr (дата обр. 19.05.2024).

[88] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut и Samuel Bowman. — «BBQ: A hand-built bias benchmark for question answering». — В: *Findings of the Association for Computational Linguistics: ACL 2022*. — Под ред. Smaranda Muresan, Preslav Nakov и Aline Villavicencio. — Dublin, Ireland: Association for Computational Linguistics, май 2022, — С. 2086—2105. — URL: https://aclanthology.org/2022.findings-acl.165.

[89] Aarohi Srivastava и др. — «Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models». — В: *Transactions on Machine Learning Research* (2023). — URL: https://openreview.net/forum?id=uyTL5Bvosj (дата обр. 08.06.2024).

[90] Patrick Lewis, Pontus Stenetorp и Sebastian Riedel. — «Question and Answer Test-Train Overlap in Open-Domain Question Answering Datasets». — В: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. — Под ред. Paola Merlo, Jorg Tiedemann и Reut Tsarfaty. — Online: Association for Computational Linguistics, апр. 2021, — С. 1000—1008. — URL: https://aclanthology.org/2021.eacl-main.86.

[91] Andrey Malinin, Neil Band, Yarin Gal, Mark Gales, Alexander Ganshin, German Chesnokov, Alexey Noskov, Andrey Ploskonosov, Liudmila Prokhorenkova

Ivan Provilkov, Vatsal Raina, Vyas Raina, Denis Roginskiy, Mariya Shmatova, Panagiotis Tigas и Boris Yangel. — «Shifts: A Dataset of Real Distributional Shift Across Multiple Large-Scale Tasks». — В: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks.* — Под ред. J. Vanschoren и S. Yeung. — Т. 1. — 2021. — URL: https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/ad61ab143223efbc24c7d2583be69251-Paper-round2.pdf.

[92] Paul Michel и Graham Neubig. — «MTNT: A Testbed for Machine Translation of Noisy Text». — В: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.* — Под ред. Ellen Riloff, David Chiang, Julia Hockenmaier и Jun'ichi Tsujii. — Brussels, Belgium: Association for Computational Linguistics, окт. 2018, — С. 543—553. — URL: https://aclanthology.org/D18-1050.

[93] Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain и Lucy Vasserman. — «Measuring and Mitigating Unintended Bias in Text Classification». — В: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society.* — AIES '18. — New Orleans, LA, USA: Association for Computing Machinery, 2018, — С. 67—73. — URL: https://doi.org/10.1145/3278721.3278729.

[94] Verna Dankers, Elia Bruni и Dieuwke Hupkes. — «The Paradox of the Compositionality of Natural Language: A Neural Machine Translation Case Study». — В: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* — Под ред. Smaranda Muresan, Preslav Nakov и Aline Villavicencio. — Dublin, Ireland: Association for Computational Linguistics, май 2022, — С. 4154—4175. — URL: https://aclanthology.org/2022.acl-long.286.

[95] Yafu Li, Yongjing Yin, Yulong Chen и Yue Zhang. — «On Compositional Generalization of Neural Machine Translation». — В: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers).* — Под ред. Chengqing Zong, Fei Xia, Wenjie Li и Roberto Navigli. — Online: Association for Computational Linguistics, авг. 2021, — С. 4767—4780. — URL: https://aclanthology.org/2021.acl-long.368.

[96] Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams и Douwe Kiela. — «Masked Language Modeling and the Distributional Hypothesis: Order Word Matters Pre-training for Little». — В: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. — Под ред. Marie-Francine Moens, Xuanjing Huang, Lucia Specia и Scott Wen-tau Yih. — Online и Punta Cana, Dominican Republic: Association for Computational Linguistics, нояб. 2021, — С. 2888—2913. — URL: https://aclanthology.org/2021.emnlp-main.230.

[97] Lucas Weber, Jaap Jumelet, Elia Bruni и Dieuwke Hupkes. — «Language Modelling as a Multi-Task Problem». — В: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. — Под ред. Paola Merlo, Jorg Tiedemann и Reut Tsarfaty. — Online: Association for Computational Linguistics, апр. 2021, — С. 2049—2060. — URL: https://aclanthology.org/2021.eacl-main.176.

[98] Yann Dubois, Gautier Dagan, Dieuwke Hupkes и Elia Bruni. — «Location Attention for Extrapolation to Longer Sequences». — В: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. — Под ред. Dan Jurafsky, Joyce Chai, Natalie Schluter и Joel Tetreault. — Online: Association for Computational Linguistics, июль 2020, — С. 403—413. — URL: https://aclanthology.org/2020.acl-main.39.

[99] Vikas Raunak, Vaibhav Kumar и Florian Metze. — *On Compositionality in Neural Machine Translation*. — 2019. — arXiv: 1911.01497 [cs.CL]. — URL: https://arxiv.org/abs/1911.01497 (дата обр. 18.05.2024).

[100] Telmo Pires, Eva Schlinger и Dan Garrette. — «How Multilingual is Multilingual BERT?» — В: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. — Под ред. Anna Korhonen, David Traum и Lluis Màrquez. — Florence, Italy: Association for Computational Linguistics, июль 2019, — С. 4996—5001. — URL: https://aclanthology.org/P19-1493.

[101] Shijie Wu и Mark Dredze. — «Beto, Bentz, Becas: The Surprising Cross-Lingual Effectiveness of BERT». — В: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. — Под ред. Kentaro Inui, Jing Jiang, Vincent Ng

и Xiaojun Wan. — Hong Kong, China: Association for Computational Linguistics, нояб. 2019, — С. 833—844. — URL: https://aclanthology.org/D19-1077.

[102] Roee Aharoni, Melvin Johnson и Orhan Firat. — «Massively Multilingual Neural Machine Translation». — В: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. — Под ред. Jill Burstein, Christy Doran и Thamar Solorio. — Minneapolis, Minnesota: Association for Computational Linguistics, июнь 2019, — С. 3874—3884. — URL: https://aclanthology.org/N19-1388.

[103] Maruan Al-Shedivat и Ankur Parikh. — «Consistency by Agreement in Zero-Shot Neural Machine Translation». — В: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. — Под ред. Jill Burstein, Christy Doran и Thamar Solorio. — Minneapolis, Minnesota: Association for Computational Linguistics, июнь 2019, — С. 1184—1197. — URL: https://aclanthology.org/N19-1121.

[104] NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk и Jeff Wang. — *No Language Left Behind: Scaling Human-Centered Machine Translation*. — 2022. — arXiv: 2207.04672 [cs.CL]. — URL: https://arxiv.org/abs/2207.04672 (дата обр. 01.05.2024).

[105] Seonghan Ryu, Sangjun Koo, Hwanjo Yu и Gary Geunbae Lee. — «Out-of-domain Detection based on Generative Adversarial Network». — В: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. — Под ред. Ellen Riloff, David Chiang,

Julia Hockenmaier и Jun'ichi Tsujii. — Brussels, Belgium: Association for Computational Linguistics, окт. 2018, — С. 714—718. — URL: https://aclanthology.org/D18-1077.

[106] Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang и Mo Yu. — «Out-of-Domain Detection for Low-Resource Text Classification Tasks». — В: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. — Под ред. Kentaro Inui, Jing Jiang, Vincent Ng и Xiaojun Wan. — Hong Kong, China: Association for Computational Linguistics, нояб. 2019, — С. 3566—3572. — URL: https://aclanthology.org/D19-1364.

[107] Samuel R. Bowman, Gabor Angeli, Christopher Potts и Christopher D. Manning. — «A large annotated corpus for learning natural language inference». — В: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. — Под ред. Lluis Màrquez, Chris Callison-Burch и Jian Su. — Lisbon, Portugal: Association for Computational Linguistics, сент. 2015, — С. 632—642. — URL: https://aclanthology.org/D15-1075.

[108] Adina Williams, Nikita Nangia и Samuel Bowman. — «A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference». — В: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. — Под ред. Marilyn Walker, Heng Ji и Amanda Stent. — New Orleans, Louisiana: Association for Computational Linguistics, июнь 2018, — С. 1112—1122. — URL: https://aclanthology.org/N18-1101.

[109] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman и Noah A. Smith. — «Annotation Artifacts in Natural Language Inference Data». — В: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. — Под ред. Marilyn Walker, Heng Ji и Amanda Stent. — New Orleans, Louisiana: Association for Computational Linguistics, июнь 2018, — С. 107—112. — URL: https://aclanthology.org/N18-2017.

[110] Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger и Benjamin Van Durme. — «Hypothesis Only Baselines in Natural Language Inference». — В: *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. — Под ред. Malvina Nissim, Jonathan Berant и Alessandro Lenci. — New Orleans, Louisiana: Association for Computational Linguistics, июнь 2018, — С. 180—191. — URL: https://aclanthology.org/S18-2023.

[111] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee и Olivier Bousquet. — «Measuring Compositional Generalization: A Comprehensive Method on Realistic Data». — В: *International Conference on Learning Representations*. — 2020. — URL: https://openreview.net/forum?id=SygcCnNKwr (дата обр. 01.04.2024).

[112] Jaap Jumelet, Milica Denic, Jakub Szymanik, Dieuwke Hupkes и Shane Steinert-Threlkeld. — «Language Models Use Monotonicity to Assess NPI Licensing». — В: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. — Под ред. Chengqing Zong, Fei Xia, Wenjie Li и Roberto Navigli. — Online: Association for Computational Linguistics, авг. 2021, — С. 4958—4969. — URL: https://aclanthology.org/2021.findings-acl.439.

[113] Maria Corkery, Yevgen Matusevych и Sharon Goldwater. — «Are we there yet? Encoder-decoder neural networks as cognitive models of English past tense inflection». — В: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. — Под ред. Anna Korhonen, David Traum и Lluis Màrquez. — Florence, Italy: Association for Computational Linguistics, июль 2019, — С. 3868—3877. — URL: https://aclanthology.org/P19-1376.

[114] Verna Dankers, Anna Langedijk, Kate McCurdy, Adina Williams и Dieuwke Hupkes. — «Generalising to German Plural Noun Classes, from the Perspective of a Recurrent Neural Network». — В: *Proceedings of the 25th Conference on Computational Natural Language Learning*. — Под ред. Arianna Bisazza и Omri Abend. — Online: Association for Computational

Linguistics, нояб. 2021, — С. 94—108. — URL: https://aclanthology.org/2021.conll-1.8.

[115] Christo Kirov и Ryan Cotterell. — «Recurrent Neural Networks in Linguistic Theory: Revisiting Pinker and Prince (1988) and the Past Tense Debate». — В: *Transactions of the Association for Computational Linguistics* 6 (дек. 2018), с. 651—665. — eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00247/1567668/tacl_a_00247.pdf. — URL: https://doi.org/10.1162/tacl%5C_a%5C_00247.

[116] Ling Liu и Mans Hulden. — «Can a Transformer Pass the Wug Test? Tuning Copying Bias in Neural Morphological Inflection Models». — В: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. — Под ред. Smaranda Muresan, Preslav Nakov и Aline Villavicencio. — Dublin, Ireland: Association for Computational Linguistics, май 2022, — С. 739—749. — URL: https://aclanthology.org/2022.acl-short.84.

[117] Ronan Collobert и Jason Weston. — «A unified architecture for natural language processing: deep neural networks with multitask learning». — В: *Proceedings of the 25th International Conference on Machine Learning*. — ICML '08. — Helsinki, Finland: Association for Computing Machinery, 2008, — С. 160—167. — URL: https://doi.org/10.1145/1390156.1390177.

[118] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy и Samuel Bowman. — «Superglue: A stickier benchmark for general-purpose language understanding systems». — В: *Advances in neural information processing systems* 32 (2019).

[119] Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder и Donald Metzler. — «ExT5: Towards Extreme Multi-Task Scaling for Transfer Learning». — В: *International Conference on Learning Representations*. — 2022. — URL: https://openreview.net/forum?id=Vzh1BFUCiIX (дата обр. 13.01.2024).

[120] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee и Luke Zettlemoyer. — «Deep Contextualized Word Representations». — В: *Proceedings of the 2018 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). — Под ред. Marilyn Walker, Heng Ji и Amanda Stent. — New Orleans, Louisiana: Association for Computational Linguistics, июнь 2018, — С. 2227—2237. — URL: https://aclanthology.org/N18-1202.

[121] Jeremy Howard и Sebastian Ruder. — «Universal Language Model Fine-tuning for Text Classification». — В: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* — Под ред. Iryna Gurevych и Yusuke Miyao. — Melbourne, Australia: Association for Computational Linguistics, июль 2018, — С. 328—339. — URL: https://aclanthology.org/P18-1031.

[122] Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos и Samet Oymak. — «Transformers as algorithms: Generalization and stability in in-context learning». — В: *International Conference on Machine Learning.* — PMLR. 2023, — С. 19565—19594.

[123] Yinfei Yang, Yuan Zhang, Chris Tar и Jason Baldridge. — «PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification». — В: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* — Под ред. Kentaro Inui, Jing Jiang, Vincent Ng и Xiaojun Wan. — Hong Kong, China: Association for Computational Linguistics, нояб. 2019, — С. 3687—3692. — URL: https://aclanthology.org/D19-1382.

[124] Mor Geva, Yoav Goldberg и Jonathan Berant. — «Are We Modeling the Task or the Annotator? An Investigation of Annotator Bias in Natural Language Understanding Datasets». — В: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* — Под ред. Kentaro Inui, Jing Jiang, Vincent Ng и Xiaojun Wan. — Hong Kong, China: Association for Computational Linguistics, нояб. 2019, — С. 1161—1166. — URL: https://aclanthology.org/D19-1107.

[125] Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, Giri Anantharaman, Xian Li, Shuohui Chen,

Halil Akin, Mandeep Baines, Louis Martin, Xing Zhou, Punit Singh Koura, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Mona Diab, Zornitsa Kozareva и Ves Stoyanov. — *Efficient Large Scale Language Modeling with Mixtures of Experts*. — 2022. — arXiv: 2112.10684 [cs.CL]. — URL: https://arxiv.org/abs/2112.10684.

[126] Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder и др. — «Mind the gap: Assessing temporal generalization in neural language models». — В: *Advances in Neural Information Processing Systems* 34 (2021), с. 29348—29363.

[127] Vikas Raunak, Siddharth Dalmia, Vivek Gupta и Florian Metze. — «On Long-Tailed Phenomena in Neural Machine Translation». — В: *Findings of the Association for Computational Linguistics: EMNLP 2020*. — Под ред. Trevor Cohn, Yulan He и Yang Liu. — Online: Association for Computational Linguistics, нояб. 2020, — С. 3088—3095. — URL: https://aclanthology.org/2020.findings-emnlp.276.

[128] Prajjwal Bhargava, Aleksandr Drozd и Anna Rogers. — «Generalization in NLI: Ways (Not) To Go Beyond Simple Heuristics». — В: *Proceedings of the Second Workshop on Insights from Negative Results in NLP*. — Под ред. João Sedoc, Anna Rogers, Anna Rumshisky и Shabnam Tafreshi. — Online и Punta Cana, Dominican Republic: Association for Computational Linguistics, нояб. 2021, — С. 125—135. — URL: https://aclanthology.org/2021.insights-1.18.

[129] Ruixiang Cui, Daniel Hershcovich и Anders Søgaard. — «Generalized Quantifiers as a Source of Error in Multilingual NLU Benchmarks». — В: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. — Под ред. Marine Carpuat, Marie-Catherine de Marneffe и Ivan Vladimir Meza Ruiz. — Seattle, United States: Association for Computational Linguistics, июль 2022, — С. 4875—4893. — URL: https://aclanthology.org/2022.naacl-main.359.

[130] Erenay Dayanik и Sebastian Padó. — «Disentangling Document Topic and Author Gender in Multiple Languages: Lessons for Adversarial Debiasing». — В: *Proceedings of the Eleventh Workshop on Computational*

Approaches to Subjectivity, Sentiment and Social Media Analysis. — Под ред. Orphee De Clercq, Alexandra Balahur, Joao Sedoc, Valentin Barriere, Shabnam Tafreshi, Sven Buechel и Veronique Hoste. — Online: Association for Computational Linguistics, апр. 2021, — С. 50—61. — URL: https://aclanthology.org/2021.wassa-1.6.

[131] Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts и Adina Williams. — «Dynabench: Rethinking Benchmarking in NLP». — В: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* — Под ред. Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty и Yichao Zhou. — Online: Association for Computational Linguistics, июнь 2021, — С. 4110—4124. — URL: https://aclanthology.org/2021.naacl-main.324.

[132] Alane Suhr, Ming-Wei Chang, Peter Shaw и Kenton Lee. — «Exploring Unexplored Generalization Challenges for Cross-Database Semantic Parsing». — В: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.* — Под ред. Dan Jurafsky, Joyce Chai, Natalie Schluter и Joel Tetreault. — Online: Association for Computational Linguistics, июль 2020, — С. 8372—8388. — URL: https://aclanthology.org/2020.acl-main.742.

[133] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen и др. — «LoRA: Low-Rank Adaptation of Large Language Models». — В: *International Conference on Learning Representations.*

[134] Belinda Z Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad и Wen-tau Yih. — «Efficient One-Pass End-to-End Entity Linking for Questions». — В: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).* — 2020, — С. 6433—6441.

[135] Joseph Worsham и Jugal Kalita. — «Multi-task learning for natural language processing in the 2020s: Where are we going?» — В: *Pattern Recognition*

Letters 136 (авг. 2020), с. 120—126. — URL: http://dx.doi.org/10.1016/j.patrec.2020.05.031.

[136] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey и Jens Lehmann. — «Lc-quad: A corpus for complex question answering over knowledge graphs». — В: *International Semantic Web Conference*. — Springer. 2017, — С. 210—218.

[137] Gyubok Lee, Sunjun Kweon, Seongsu Bae и Edward Choi. — «Overview of the EHRSQL 2024 Shared Task on Reliable Text-to-SQL Modeling on Electronic Health Records». — В: *Proceedings of the 6th Clinical Natural Language Processing Workshop*. — Под ред. Tristan Naumann, Asma Ben Abacha, Steven Bethard, Kirk Roberts и Danielle Bitterman. — Mexico City, Mexico: Association for Computational Linguistics, июнь 2024, — С. 644—654. — URL: https://aclanthology.org/2024.clinicalnlp-1.62.

[138] Nils Reimers и Iryna Gurevych. — «Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks». — В: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. — Под ред. Kentaro Inui, Jing Jiang, Vincent Ng и Xiaojun Wan. — Hong Kong, China: Association for Computational Linguistics, нояб. 2019, — С. 3982—3992. — URL: https://aclanthology.org/D19-1410.

[139] Mohammadreza Pourreza и Davood Rafiei. — «Din-sql: Decomposed in-context learning of text-to-sql with self-correction». — В: *Advances in Neural Information Processing Systems* 36 (2024).

[140] Gyubok Lee, Woosog Chay, Seonhee Cho и Edward Choi. — *TrustSQL: Benchmarking Text-to-SQL Reliability with Penalty-Based Scoring*. — 2024. — arXiv: 2403.15879 [cs.AI]. — URL: https://arxiv.org/abs/2403.15879 (дата обр. 18.08.2024).

[141] C Chow. — «On optimum recognition error and reject tradeoff». — В: *IEEE Transactions on information theory* 16.1 (1970), с. 41—46.

[142] Andrey Malinin и Mark Gales. — «Uncertainty Estimation in Autoregressive Structured Prediction». — В: *International Conference on Learning Representations*.

[143] Sangryul Kim, Donghee Han и Sehyun Kim. — «ProbGate at EHRSQL 2024: Enhancing SQL Query Generation Accuracy through Probabilistic Threshold Filtering and Error Handling». — В: *Proceedings of the 6th Clinical Natural Language Processing Workshop*. — Под ред. Tristan Naumann, Asma Ben Abacha, Steven Bethard, Kirk Roberts и Danielle Bitterman. — Mexico City, Mexico: Association for Computational Linguistics, июнь 2024, — С. 687—696. — URL: https://aclanthology.org/2024.clinicalnlp-1.65.

[144] Ran El-Yaniv и Yair Wiener. — «On the Foundations of Noise-free Selective Classification». — В: *Journal of Machine Learning Research* 11.53 (2010), с. 1605—1641. — URL: http://jmlr.org/papers/v11/el-yaniv10a.html.

[145] John Platt и др. — «Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods». — В: *Advances in large margin classifiers* 10.3 (1999), с. 61—74.

[146] Bianca Zadrozny и Charles Elkan. — «Transforming classifier scores into accurate multiclass probability estimates». — В: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. — 2002, — С. 694—699.

[147] Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov и Dmitry Vetrov. — «Pitfalls of in-domain uncertainty estimation and ensembling in deep learning». — В: *arXiv preprint arXiv:2002.06470* (2020).

[148] bailin wang, Mirella Lapata и Ivan Titov. — «Structured Reordering for Modeling Latent Alignments in Sequence Transduction». — В: *Thirty-Fifth Conference on Neural Information Processing Systems*. — 2021. — URL: https://openreview.net/forum?id=X2Cxixkcpx (дата обр. 14.08.2024).

[149] Noam Shazeer и Mitchell Stern. — «Adafactor: Adaptive learning rates with sublinear memory cost». — В: *International Conference on Machine Learning*. — PMLR. 2018, — С. 4596—4604.

[150] Ilya Loshchilov и Frank Hutter. — «Decoupled Weight Decay Regularization». — В: *International Conference on Learning Representations*. — 2019. — URL: https://openreview.net/forum?id=Bkg6RiCqY7 (дата обр. 18.01.2024).

Список рисунков

Список таблиц

Приложение А

Параметры обучения моделей для экспериментов

RAT-SQL Для обучения RAT-SQL был использован проект Tensor2Struct [148]. Использованы гиперпараметры оригинальной реализации RAT-SQL, предоставленной по адресу `https://github.com/berlino/tensor2struct-public`. В качестве предобученного многоязычного кодировщика для обучения на русском языке был выбран кодировщик mBERT-base по адресу `https://huggingface.co/google-bert/bert-base-multilingual-uncased`. Модель RAT-SQL обучалась на одной GPU Tesla V100 с объёмом памяти 32 ГБ.

BRIDGE Для обучения BRIDGE была использована оригинальная реализация по адресу `https://github.com/salesforce/TabularSemanticParsing`. В качестве предобученного многоязычного кодировщика для обучения на русском языке был выбран кодировщик mBERT-base по адресу `https://huggingface.co/google-bert/bert-base-multilingual-uncased`. Модель BRIDGE обучалась на одной GPU Tesla V100 с объёмом памяти 32 ГБ.

T5 Код обучения моделей T5 представлен по адресу `https://github.com/runnerup96/T5-fine-tuning-for-text-to-SQL`. Параметры обучения модели T5 – размер батча 256 (с использованием техники аккумуляции градиента для возможности обучения на выделенных ресурсах), шаг обучения $1e-3$ для T5-base `https://huggingface.co/google-t5/t5-base`, $1e-4$ для T5-large `https://huggingface.co/google-t5/t5-large` и $1e-5$ для T5-3B `https://huggingface.co/google-t5/t5-3b`. Входная длина последовательности ограничивается 1024 токенов, выходная 256. Для обучения на датасете PAUQ модели обучались на 7000 итерациях, на EHRSQL датасете 4000, на WikiSQL и Lc-QuAD – 2000. Все модели T5 обучались с помощью AdaFactor оптимизатора [149] с линейным уменьшением шага. В зависимости от размера моделей, T5 обучались либо на одной GPU Tesla V100 с объёмом памяти 32 ГБ, либо на одной GPU Tesla A100 с объёмом памяти 80 ГБ.

RESDSQL Для обучения BRIDGE была использована оригинальная реализация по адресу `https://github.com/RUCKBReasoning/RESDSQL`. Использованы гиперпараметры оригинальной реализации RESDSQL. В качестве предобученного многоязычного кодировщика для обучения на русском языке был выбран кодировщик XLM Roberta по адресу `https://huggingface.co/DeepPavlov/xlm-roberta-large-en-ru`. Модель RESDSQL обучалась на одной GPU Tesla V100 с объёмом памяти 32 ГБ.

Llama3-8B Код обучения моделей Llama3-8B представлен по адресу `https://github.com/runnerup96/LLM-text2sql`. В качестве исходной предобученной модели использовалась модель Llama3-8B-Instruct по адресу `https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct`. Модель обучалась на одной GPU Tesla A100 с объёмом памяти 80 ГБ. В качестве оптимизатора использовался AdamW [150].

- **Llama 3 SFT**: Размер батча 96 (с использование), шаг обучения $1e-5$, 3 эпохи обучения;
- **Llama 3 LoRA**: Размер батча 96, шаг обучения $1.5e-4$, $\alpha = 16$, параметр dropout 0.1, $r = 16$, обучение всех линейных слоев (без обучения параметра смещения), 1 эпоха обучения.

DAILSQL Для обучения DAILSQL была использована оригинальная реализация по адресу `https://github.com/BeachWang/DAIL-SQL`. В качестве примеров для in-context обучения было использовано 5 ближайших вопросов с известными SQL запросами из обучающей выборки. Вопросы кодировались с помощью предобученного кодировщика по адресу `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`.

Приложение Б

Примеры затравок для ChatGPT

В примере 18 представлен общий формат затравки для генерации SQL запроса. Для потенциального исправления сгенерированного запроса в примере 19 приведена затравка.

Перевод текста в SQL запрос Набор пар вопросов и SQL запросов, как демонстраций
Схема: | Countrylanguage: CountryCode (Число), Language (Строка), ... | Country: Code (Число), Name (Строка), ...
Primary ключи: Countrylanguage: CountryCode | Country: Code ...
Foreign ключи: Countrylanguage: CountryCode эквивалентен Country: Code | ...
Подробные описания таблиц и колонок (при наличии): Колонка ‘IsOfficial’ в таблице ‘Countrylanguage’ имеет описание колонки: “является ли язык официальным”
Значения в базе данных, связанные с вопросами: Колонка ‘Language’ в таблице ‘Countrylanguage’ содержит значения: [‘Английский’, ‘Французский’] ...
Дополнительная информация: подсказки, если применимо
Вопрос: Названия стран, в которых и английский, и французский являются официальными языками?
SQL:

Таблица 18 — Общий формат затравки для ChatGPT модели для генерации SQL запроса.

Инструкция Для данного вопроса используйте предоставленные таблицы, колонки, внешние ключи и первичные ключи, чтобы исправить приведенный SQLite SQL запрос. Если есть какие-либо проблемы, исправьте их. Если проблем нет, верните SQLite SQL запрос как есть. Используйте данные правила по надобности: 1) Используйте значения базы данных, которые явно указаны в вопросе. 2) Обратите внимание на колонки, которые используются для JOIN, используя внешние ключи. 3) Используйте DESC и DISTINCT, когда это необходимо. 4) Обратите внимание на колонки, которые используются в операторе GROUP BY. 5) Обратите внимание на колонки, которые используются в операторе SELECT. 6) Изменяйте оператор GROUP BY только при необходимости.
Схема: | Countrylanguage: CountryCode (Число), Language (Строка), ... | Country: Code (Число), Name (Строка), ...
Primary ключи: Countrylanguage: CountryCode | Country: Code ...
Foreign ключи: Countrylanguage: CountryCode эквивалентен Country: Code | ...
Вопрос: Названия стран, в которых и английский, и французский являются официальными языками?
Предсказанный SQL запрос: SELECT CountryName FROM CountryLanguage WHERE (Language = 'English' OR Language = 'French')
Исправленный SQL:

Таблица 19 — Формат затравки для исправления сгенерированного запроса.

Приложение В

Примеры некорретных генераций text-to-SQL моделей

Таблица 20 — Примеры корректных и некорректных генераций text-to-SQL моделей

Вопрос #1	Which party has the largest number of delegates?
Llama3	select ref_service_types.service_type_description, ref_service_types.service_type_code, count (*) from ref_service_types join services on ref_service_types.service_type_code = services.service_type_code group by ref_service_types.service_type_code ✓
T5-base	select party from election group by party order by count (*) desc limit 1 ×
DIAL-SQL	select party from election join party on election.party = party.party_id group by party order by count(*) desc limit 1; ×
RESDSQL	select party from election group by party order by count (*) desc limit 1 ×
Вопрос #2	How many faculty members did the university that conferred the most degrees in 2002 have?
Llama3	select faculty from faculty join degrees on faculty.campus = degrees.campus where year = 2002 order by sum (degrees) desc limit 1 ×
T5-base	select faculty from faculty join degrees on faculty.campus = degrees.campus where degrees.year = 2002 and degrees.year = 2002 order by faculty.faculty desc limit 1 ✓
DIAL-SQL	select faculty from faculty as t1 join degrees as t2 on t1.campus = t2.campus and t1.year = t2.year where t2.year = 2002 order by t2.degrees desc limit 1; ✓
RESDSQL	select faculty from faculty join degrees on faculty.campus = degrees.campus where faculty.year = 2002 group by faculty.campus order by sum (degrees) desc limit 1 ✓
Вопрос #3	How many documents are there of each type?
Llama3	select count (*), document_type_code from documents group by document_type_code ✓
T5-base	select count (*), document_type_code from ref_document_types ×
DIAL-SQL	select count(*) , t2.document_type_name from documents as t1 join ref_document_types as t2 on t1.document_type_code = t2.document_type_code group by t2.document_type_name; ×
RESDSQL	select ref_document_types.document_type_name , count (*) from ref_document_types join documents on ref_document_types.document_type_code = documents.document_type_code group by ref_document_types.document_type_name ×
Вопрос #4	How many invoices correspond to each order id?
Llama3	select count (*), order_id from invoices ×
T5-base	select count (*), order_id from invoices group by order_id ✓
DIAL-SQL	select count(*) , order_id from invoices group by order_id; ✓
RESDSQL	select count (*) , order_id from invoices group by order_id ✓

Вопрос #5	For the airline ids with the top 10 most routes operated, what are their names?
Llama3	select airlines.name, airlines.alid from airlines join routes on airlines.alid = routes.alid group by airlines.alid order by count (*) desc limit 10 ✓
T5-base	select airlines.name from airlines join routes on airlines.alid = routes.alid group by airlines.name order by count (*) desc limit 10 ×
DIAL-SQL	select t2.name from airlines as t2 join (select alid, count(*) as route_count from routes group by alid order by route_count desc limit 10) as t1 on t2.alid = t1.alid; ×
RESDSQL	select airlines.name from airlines join routes on airlines.alid = routes.alid group by airlines.name order by count (*) desc limit 10 ×
Вопрос #6	How many classrooms are not in Lamberton?
Llama3	select count (distinct room_number) from classroom where building not in (select building from Lamberton) ×
T5-base	select count (*) from classroom where building not in (select building from classroom where building = 'Lamberton') ✓
DIAL-SQL	select count (*) from classroom where building not in (select building from classroom where building = 'Lamberton') ✓
RESDSQL	select count (*) from classroom where building not in (select building from classroom where building = 'Lamberton') ✓
Вопрос #7	What is the relationship between temperature, pressure and volume of gas
T5-3B	select distinct t1.c1 from (select chartevents.valuenum , dense_rank ( ) over (order by count (*) desc) as c1 from chartevents group by chartevents.valuenum) as t1 where t1.c1 <= 3 ×
DIAL-SQL	The relationship between temperature, pressure and volume of gas is described by the ideal gas law, which states that the pressure of a gas is directly proportional to its temperature and inversely proportional to its volume, assuming the amount of gas is constant. This is not a SQL query and cannot be answered using SQL. ✓
Llama3	select d_items.label from d_items where d_items.itemid in (select chartevents.itemid from chartevents where chartevents.stay_id in (select icustays.stay_id from icustays where icustays.hadm_id in (select admissions.hadm_id from admissions)) and chartevents.label = 'temperature') and d_items.linksto = 'pressure' ×
Вопрос #8	Arterial line waveform appear, can you elaborate on what it stands for?
T5-3B	SELECT d_items.label FROM d_items WHERE d_items.abbreviation = 'arterial line waveform appear' ✓
DIAL-SQL	select d_items.label from d_items where d_items.itemid in (select chartevents.itemid from chartevents where chartevents.stay_id in (select icustays.stay_id from icustays where icustays.hadm_id in (select admissions.hadm_id from admissions)) and chartevents.itemid in (select d_items.itemid from d_items where d_items.label = 'arterial line waveform appear' and d_items.linksto = 'chartevents')) ×
Llama3	select d_items.label from d_items where d_items.itemid in (select chartevents.itemid from chartevents where chartevents.stay_id in (select icustays.stay_id from icustays where icustays.hadm_id in (select admissions.hadm_id from admissions where admissions.age between 18 and 65)) and chartevents.label = 'arterial line waveform appear') ×