Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б11-мм

Система для автоматизированной проверки больших языковых моделей на основе заданий государственных экзаменов

Выродов Михаил Владимирович

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
асс. кафедры СП Е. С. Спирин

Санкт-Петербург
2023

Оглавление

Введение

В последние годы большие языковые модели (БЯМ) получил огромное развитие и стали использоваться повсеместно. Они для хорошей работы должны сначала обучиться на огромном массиве текстовых данных. После обучения модели работники должны провалидировать её работу и если результат валидации удовлетворительный, то языковая модель готова к использованию.

БЯМ способны на многое. Например, они могут осмысленно ответить на произвольный вопрос, составить краткое содержание большого текста, перевести текст с одного языка на другой. Популярным примером БЯМ является GPT-4 [2]. Однако, несмотря на их впечатляющие достижения, внедрение БЯМ в IT проекты требует тщательного процесса валидации.

Валидация языковых моделей (ЯМ) – это процесс проверки их способности к правильному пониманию и генерации текста. ЯМ нуждаются в валидации для того, чтобы проверить их эффективность и способность разумно отвечать не только на вопросы, на которых модель обучалась, но и на вопросы, которые модель видит в первый раз.

Одним из ключевых аспектов успешной валидации является выбор подходящего набора данных. Он должен проверять способность ЯМ отвечать на вопросы по множеству различных тем. Кроме того, задания в наборе данных должны быть различной сложности, т.к. хорошая БЯМ должна уметь отвечать как на вопросы уровня младшей школы, так и на вопросы университетского уровня.

MMLU [3] — пример набора данных для валидации ЯМ на английском языке. Сейчас он является одним из стандартов для валидации ЯМ. Он был впервые представлен на конференции A* уровня ICLR[1] в 2021 году. Он состоит из ≈16 тысяч вопросов на выбор одного варианта ответа из четырёх. Вопросы составлены по 57 различным темам, имеют разделение по уровням сложности и были взяты из американских экзаменов для поступления в магистратуру или аспирантуру или для

[1] https://iclr.cc/

получения медицинской лицензии.

Хорошим набором данных для валидации БЯМ, обученных на русскоязычных текстах, могут послужить задания из ЕГЭ и ОГЭ. Они покрывают множество различных тем и содержат как лёгкие вопросы, так и сложные. Кроме этого, в интернете есть множество сайтов, на которых есть огромные банки заданий из этих экзаменов с ответами и пояснениями к ответу, что сильно упрощает создание большого валидационного набора данных.

1. Постановка задачи

Целью работы является создание системы, предназначенной для валидации БЯМ, которая будет основана на наборе данных, состоящем из заданий государственных экзаменов. Для её выполнения были поставлены следующие задачи:

1. Провести обзор предметной области, который включает в себя анализ популярных валидационных наборов данных, выбор библиотеки для парсинга заданий государственных экзаменов и поиск сайтов с большой базой этих заданий, пригодных для парсинга;

2. Собрать банк вопросов по разным предметам из этих сайтов, провести очистку и классифицировать задания по типам вопросов и способам взаимодействия с ЯМ;

3. Создать систему, которая будет валидировать подаваемую на вход ЯМ на созданном наборе данных; Для этого система должна поддерживать наиболее популярные интерфейсы для взаимодействия с ЯМ;

4. Апробировать систему на существующих популярных БЯМ, построить их качественное сравнение.

2. Обзор

2.1. Нужные понятия из машинного обучения

Языковая модель — это распределение вероятностей по последовательностям слов. Любой последовательности слов длины m присваивается вероятность $P(w_1, w_2, ..., w_m)$. Обычно ЯМ представляет собой рекуррентную нейронную сеть.

Эта нейронная сеть учится предсказывать вероятности $P(w, context)$ $\forall w \in V$, где V — множество всех возможных слов, а $context$ — какой-то набор слов. $P(w, context)$ — вероятность того, что при добавлении к тексту $context$ следующего слова w, получившийся текст больше всего похож на тексты, на которых обучалась ЯМ. То есть модель, получив какой-то текстовый запрос $context$, считает вероятности $P(w, context)$ $\forall w \in V$, после этого выбирает наиболее вероятное следующее слово w_i для текста $context$ и теперь уже считает вероятность $P(w, context + w_i)$ $\forall w \in V$, где $context + w_i$ — строка $context$, сконкатенированная со словом w_i. Далее ЯМ выбирает наиболее вероятное следующее слово для строки $context + w_i$. Так ЯМ предугадывает следующие слова до того момента, пока самым вероятным следующим словом не станет специальное слово, обозначающее конец генерации текста. Накопившаяся последовательность слов является ответом ЯМ на изначальный текстовый запрос $context$.

Возникает вопрос, как ЯМ могут для любого слова w и последовательности слов $context$ посчитать вероятность того, что w является наиболее подходящим следующим словом для текста $context$? Данное распределение вероятностей нейронная сеть учит благодаря корректировки значений своих весов с помощью алгоритма градиентного спуска. Подробнее о самой популярной архитектуре языковой модели на основе нейронной сети написано в статье [1].

Большие языковые модели — неофициальный термин для ЯМ на основе нейронный сетей, которые имеют более миллиарда весов. БЯМ обучаются очень долго и для хорошего обучения им требуется большой

текстовый набор данных, который измеряется петабайтами.

Несмотря на это, БЯМ для лучшей эффективности решения одного конкретного типа задач часто требуют специальной доработки, которая называется fine-tuning. Она позволяет адаптировать БЯМ под конкретную задачу. Сейчас этот подход является одним из самых популярных подходов к решению бизнес-задач с помощью нейросетей, так как он задействует БЯМ, которые на данный момент могут эффективно решать множество задач, и вместо полного обучения этих моделей на наборе данных, содержащем примеры нужной бизнес-задачи, в этом подходе используются уже их обученные на петабайтах текстовых данных версии.

Описание процесса fine-tuning:

- Создание отдельного набора данных, который состоит из задач одного типа, задачи которого ЯМ надо научится решать, и ответов на них;
- Определение, какую часть весов предобученной ЯМ оставить неизменными, а какую часть обучить заново для решения бизнес-задачи;
- Процесс обучения выбранной части весов ЯМ на созданном наборе данных.

Таким образом, fine-tuning позволяет использовать весь потенциал БЯМ для решения бизнес-задач, при этом не тратя месяцы на их полное обучение под решение бизнес-задачи, а лишь изменяя малую часть их обученных на огромных массивах текстов версий.

Валидация ЯМ — важный этап, который помогает оценить качество и эффективность ЯМ. Процесс валидации включает в себя использование отдельного набора данных, отличного от того, на котором проводилось обучение модели. Это необходимо для того, чтобы проверить её способность разумно отвечать не только на вопросы, на которых модель обучалась, но и на вопросы, которые она видит в первый раз.

Набор данных для валидации должен быть репрезентативным, то есть должен отражать разнообразие данных, которые модель может встретить в реальных условиях. Также данные для валидации не должны пересекаться с данными, использованными в процессе обучения модели, так как в ином случае высока вероятность того, что после валидации мы получим ложные представления о работоспособности модели, так как она легко может ответить на вопросы, на которых обучалась, но на вопросы, которые она видит впервые, её ответы с большой вероятностью будут хуже.

Процесс валидации БЯМ:

- Разбиение валидационного набора данных на 3 части — train set, development set, test set;
- Fine-tuning выбранной БЯМ на train set;
- Оценка производительности БЯМ после fine-tuning на development set. Если БЯМ плохо себя показывает на development set, то в набор данных или в fine-tuning модели вносятся изменения до того момента, пока эффективность модели на development set не будет удовлетворительной;
- Окончательная проверка эффективности полученной после fine-tuning БЯМ на test set.

Возникает вопрос, как происходит проверка эффективности ЯМ на данных? Для этого используются метрики. Метрика — способ измерения того, насколько хорошо работает модель на данных (это определение не связано с математическим определением метрики). Метрика для одного задания из валидационного набора данных, правильного ответа на это задание и ответа модели на это задание выдаёт число, которое должно отражать, насколько ответ модели близок к правильному ответу на задание. В зависимости от оценки эффективности модели с помощью метрик, в процессе валидации делается вывод о том, стоит ли дорабатывать модель или набор данных, или же модель достаточно хорошо справляется с заданиями из валидационного набора данных.

В качестве примера метрики на заданиях, в которых надо выбрать один вариант ответа из четырёх, можно привести функцию, которая равна единице, если модель выбрала правильный ответ на задание, или равна нулю, если модель ответила на вопрос неправильно.

2.2. Выбор набора данных для валидации

Мною было изучено несколько популярных русскоязычных и англоязычных валидационных наборов данных для БЯМ. MMLU [3] — один из наиболее часто используемых наборов данных для валидации БЯМ. Он состоит из заданий из экзаменов для поступления в магистратуру и аспирантуру в США, из экзамена на получение медицинской лицензии и из различных задач университетского уровня сложности. Вопросы в этом наборе данных охватывают 57 различных тем и делятся на несколько уровней сложности. Все вопросы имеют один вид — надо выбрать один правильный ответ из четырёх возможных ответов на задание.

As of 2017, how many of the world's 1-year-old children today have been vaccinated against some disease?
(A) 80%
(B) 60%
(C) 40%
(D) 20%

Рис. 1: Пример задания из MMLU [3]

Минус этого набора данных состоит в том, что он не подходит для валидации БЯМ, обученных на русскоязычных текстах, так как MMLU англоязычный. Возникает желание сделать набор данных для валидации БЯМ, нацеленных на русского пользователя. Также хочется, чтобы он был такой же разнообразный по темам заданий и по их сложности, как MMLU.

Таким набором данных могут послужить задания из российских государственных экзаменов. Преимущества такого набора данных заключаются в том, что задания из государственных экзаменов охватывают множество различных тем и делятся на уровни сложности. Задания в

ЕГЭ по одному предмету обычно сложнее, чем задания ОГЭ по этому же предмету. Кроме того, в одном варианте ЕГЭ есть как очень простые задания, которые не требуют длинных рассуждений, так и задания, для правильно ответа на которые надо провести большую цепочку математических вычислений и знать множество теорем. Также в интернете есть общедоступные огромные базы заданий из ЕГЭ и ОГЭ с ответами и с объяснениями того, как ответ был получен. Ещё одним плюсом такого выбора набора данных является то, что задания в ЕГЭ и ОГЭ могут быть разных типов. В экзаменах встречаются как задания на выбор ответа, так и задания на установление соответствия и задания с текстовым ответом. Такие задания позволяют оценить работу модели в различных сценариях использования. Таким образом, задания из государственных экзаменов могут послужить хорошим набором данных для валидации БЯМ.

Было принято решение извлекать задания с сайта sdamgia.ru[2], так как на этом сайте есть наибольшее количество заданий из государственных экзаменов с ответами, сами задания имеют унифицированный вид и на этот сайт можно отправлять множество частых GET запросов с одного ip адреса.

[2] https://sdamgia.ru/

3. Метод

Было принято решение писать парсер сайта sdamgia.ru на языке Python с помощью фреймворка Scrapy[3], потому что у Scrapy удобный интерфейс, в рамках которого можно легко сохранять извлечённые с сайта задания в различных форматах. Также он парсит страницы намного быстрее, чем BeautifulSoup[4], так как парсинг одних и тех же страниц на BeautifulSoup у меня занимал около 30-ти минут, а на Scrapy около пяти минут.

Парсер ищет на сайте sdamgia.ru нужный вариант экзамена, извлекают оттуда задания и сохраняет их в файл. Возникает вопрос, почему парсер не просто извлекает все задания с сайта, а именно парсит вопросы из вариантов экзаменов?

На сайте sdamgia.ru есть множество заданий, которые не входят в варианты нынешних экзаменов и выбиваются из текущих стандартов заданий. Для того, чтобы валидация языковой модели была схожа с реальной проверкой знаний учеников на государственных экзаменах, было принято решение на данный момент извлекать с сайта только актуальные задания, которые могут быть в вариантах государственных экзаменов. Это удобно делать с помощью парсинга вариантов с сайта sdamgia.ru, так как функциональность этого сайта позволяет создать один вариант, который будет содержать все актуальные задания с этого сайта по конкретному предмету и по конкретному экзамену.

Парсер имеет несколько параметров:

- `subject` — предмет, по которому проводится экзамен. На данный момент можно парсить варианты экзаменов по предметам “Обществознание” и “Литература”;
- `exam_type` — тип экзамена, вариант которого будет парситься. На данный момент можно парсить варианты из ЕГЭ и ОГЭ;
- `test_id` — номер варианта экзамена на сайте sdamgia.ru. Задания

[3] https://scrapy.org/

[4] https://www.crummy.com/software/BeautifulSoup/bs4/doc/

из этого варианта будут извлечены с сайта и сохранены в файл;

- `output_file` — название и формат файла, в который будут сохранены извлечённые из варианта задания.

Инструкции по вызову парсера описаны в репозитории проекта[5].

В Scrapy за парсинг сайтов отвечает класс `scrapy.Spider`, поэтому для парсинга сайта sdamgia.ru был создан класс `SdamgiaSpider`, который наследуется от `scrapy.Spider`. При вызове парсера сначала вызывается метод `start_requests()`, внутри которого определяются ссылки на каждое задание из варианта экзамена, номер которого был передан при вызове парсера. После этого из каждого задания извлекается нужная информация в методе `parse(...)`. Вся извлечённая информация записывается в файл, имя которого передаётся при вызове парсера.

Мною с помощью данного парсера были извлечены все актуальные задания из экзаменов ЕГЭ и ОГЭ по предметам “Обществознание” и “Литература” с сайта sdamgia.ru.

Извлечённые задания были очищены от специфичных для html символов и разделены на три класса в зависимости от типа ответа на задание:

- Задания с выбором ответа;
- Задания на установление соответствия;
- Задания с текстовым ответом.

Также из каждого задания была извлечена дополнительная информация, которая может пригодится при поиске по заданиям или может послужить основой для более подробной классификации заданий. В эту информацию входят:

- Номер задания в извлечённом варианте;
- Тип задания в варианте государственного экзамена;

[5] https://github.com/deepvk/goat/

- Булевое значение, которое истинно, когда задание предполагает поиск информации по большому тексту, который есть в варианте государственного экзамена, и ложно, когда для решение задания достаточно знать только текст самого задания;
- Ссылка на задание;
- Раздел кодификатора ФИПИ, к которому принадлежит данное задание;
- Максимальное количество баллов, которое можно получить за задание;
- Таблица с критериями оценивания задания, если такая имеется;
- Html код задания.

Всего с сайта sdamgia.ru было извлечено 6211 заданий по обществознанию и литературе. Результаты парсинга приведены в таблице.

Тип экзамена	Предмет	Количество вопросов	Выбор ответа	Соответствие	Текстовый ответ
ЕГЭ	Обществознание	2936	1751	580	605
ЕГЭ	Литература	1174	108	66	1000
ОГЭ	Обществознание	1673	1130	126	417
ОГЭ	Литература	428	0	0	428

Архив с извлечёнными заданиями доступен по ссылке[6].

[6] https://drive.google.com/file/d/1bjLl2RD3CC6vN6zf4Yvo9K-IHCNRVH6C/view?usp=sharing

Заключение

В рамках данной работы были выполнены следующие задачи:

- Был проведён анализ существующих валидационных наборов данных как на английском, так и на русском языках. Также было проведено сравнение Python библиотек для парсинга web страниц. В проекте было решено использовать фреймворк Scrapy. Был проведён анализ сайтов, содержащих задания государственных экзаменов, и для парсинга был выбран сайт sdamgia.ru;
- Был создан парсер сайта sdamgia.ru, с помощью которого с сайта было извлечено 6211 заданий с экзаменов ЕГЭ и ОГЭ по предметам "Обществознание" и "Литература". Извлечённые задания были разделены на 3 класса, в зависимости от типа ответа на задание. Также с сайта для каждого задания была извлечена дополнительная информация, которая может пригодится при поиске по заданиям или может послужить основой для более подробной классификации заданий.

Задачи на следующий семестр:

- Дополнить банк вопросов заданиями из других сайтов и по другим предметам;
- Создать систему для валидации ЯМ, в которой языковая модель будет валидироваться на целом варианте заданий из государственных экзаменов;
- Проверить устойчивость созданной системы к различным изменениям текста заданий, которые не влияют на смысл задания;
- Провалидировать несколько БЯМ с помощью созданной системы вручную и автоматически;

Текущий код парсера доступен в репозитории проекта: https://github.com/deepvk/goat.

Список литературы

[1] Attention Is All You Need / Ashish Vaswani, Noam Shazeer, Niki Parmar et al. // CoRR. — 2017. — arXiv : 1706.03762.

[2] OpenAI, :, Achiam Josh et al. GPT-4 Technical Report. — 2023. — 2303.08774.

[3] Measuring Massive Multitask Language Understanding / Dan Hendrycks, Collin Burns, Steven Basart et al. // CoRR. — 2020. — 2009.03300.