**Тип работы:** Дипломная работа

**Предмет:** Информационные технологии



Чат-бот (разговорный агент, диалоговая система) — это компьютерная программа, которая действует как интерфейс между людьми пользователи. Он представляет собой программное приложение, использующие разговорный или письменный естественный язык в качестве основного средства общения. В последнее десятилетие большинство компаний во всем мире использует его как удобный сервис, обладающий следующими достоинствами:
• Клиент может получить ответ в любое время суток;
• Использование чат-бота позволяет сократить обслуживающий персонал.
Аналогичный сервис может быть полезным для распределения заданий студентам по следующим причинам:
• Задача распределения заданий, является скорее технической, не требующей высокой квалификации преподавателей;
• Студентам не нужно организовывать встречу с преподавателем, который часто бывает загружен другими

делами.
В данной работе рассмотрены как общие вопросы применения чат ботов, так и особенности их применения для распределения заданий.
1. Методы разработки чат-ботов
1.1 Анализ прототипов
В данном разделе используются книги [1,2].
Чат-боты часто называют революцией во взаимодействии пользователей с технологиями и бизнесом. У них довольно простой интерфейс по сравнению с традиционными приложениями, поскольку они требуют от пользователей только общения в чате, а чат-боты должны понимать и делать все, что от них требует пользователь, по крайней мере, теоретически. Многие отрасли переносят обслуживание клиентов на системы чат-ботов. Это связано с огромным снижением стоимости по сравнению с реальными людьми, а также с надежностью и постоянной доступностью. Чат-боты обеспечивают определенную степень поддержки пользователей без существенных дополнительных затрат. Сегодня чат-боты используются во многих сценариях, начиная от простых задач, таких как отображение данных о времени и погоде, и заканчивая более сложными операциями, такими как элементарная медицинская диагностика и общение/поддержка клиентов. Можно разработать чат-бот, который поможет вашим клиентам, когда они будут задавать определенные вопросы о вашем продукте, или вы можете создать личного чат-бота-помощника, который может выполнять основные задачи и напоминать вам, когда пора идти на встречу или в спортзал. Существует множество вариантов, где вы можете развернуть свой чат-бот, и одним из наиболее распространенных вариантов использования являются платформы социальных сетей, поскольку большинство людей используют их на регулярной основе. То же самое можно сказать и о приложениях для обмена мгновенными сообщениями.
В данном цикле студент задаёт вопрос, который отправляется мессенджеру. Мессенджер перенаправляет вопрос чат-боту, который обрабатывает информацию и готовит ответ. Далее ответ направляется к мессенджеру и мессенджер пересылает ответ студенту. Есть много вариантов системной архитектуры чат-ботов, но каждый из них имеет три основных компонента.
• Компонент понимания естественного языка;
• Диспетчер диалогов - наиболее важный компонент, который получает репрезентативный формат запроса и затем отправляет ответы.
• Генератор ответов, который определяет ответ чат-бота в соответствии с запросом;
Компонент понимания переводит текст естественного языка в семантику (смысл). Синтаксический анализ является основной задачей понимания и предоставляет языковую структуру высказывания. В зависимости от реализации компонент может использовать контекстно-свободные грамматики, сопоставление с образцом или подходы, основанные на данных.
Опишем эти функции более подробно.
1) Обнаружение темы является важным шагом распознавания текста. Точное отслеживание темы разговора может быть ценным сигналом для системы диалога.
2) Анализ намерений: Намерение – это группа высказываний со схожим смыслом. Есть два предложения: «Какое задание Вы мне можете предложить?» и «Мне нужна тема задания». Они оба имеют одно и то же значение. Это означает, что целью анализа намерений является выявление сходства между словами в значение. Если бот не понимает намерение пользователя, он бесполезен.
3) Связывание сущностей выявляет связь текста со словарём или базой знаний. В частности, разработаны системы извлекающие сущности из Интернет и Википедии, в частности.
Отслеживание контекста может существенно повысить эффективность чат-бота. Когда появляется сообщение пользователя, чат-бот получает самые последние высказывания этого пользователя из истории чата. Эти данные могут быть записаны в базу данных, а затем использоваться.
Диспетчер диалогов — это второй основной компонент любого чат-бота. Он состоит из многих частей, которые можно улучшить или расширить в будущем. Данный компонент получает пользовательский ввод от распознавателя речи и формирует ответы системы на концептуальном уровне, отправляемые в генератор естественного языка. Работа диспетчера зависит от выбранной стратегии. Стратегии могут быть основаны на правилах, знаниях, поиске и генерировании. Основанные на правилах стратегии — базируются на предыстории, шаблонах намерений и шаблонах сущностей. Эти шаблоны упорядочены по приоритетам. Поскольку стратегии на основе правил разрабатывают люди они чаще всего дают точные ответы. Если нет подходящего шаблона для ввода, то система может пытаться к получать ответ от базы знаний. В случае неудачи вход обрабатывается ансамблем моделей нейронных сетей и модулей поиска информации.

Он имеет следующие компоненты:
1) Извлечение знаний: этот модуль можно использовать, когда методы, основанные на правилах и знаниях, не работают. Этот метод предоставляет более свежие ответы по сравнению с шаблонами на основе правил.
2) Генерация данных при помощи нейронной сети: Генерация контекстных ответов, связанных с диалоговой справкой, позволит повысить качество чата. Для этого нужно принимать во внимание недавние высказывания клиента и применять глубинное машинное обучение.
Генератор естественного языка - последний основной компонент любого чат-бота. Он получает коммуникативный акт от диспетчера диалога и генерирует соответствующее текстовое представление.
Этими функциями являются:
1) Фильтрация, обеспечивающая отсеивание бессвязных или сомнительных ответов.
2) Ранжирование. Если вопрос существует более одного действительного ответа, ранжирование используется для изменения порядка высказываний-ответов сначала в соответствии с приоритетом, а затем в соответствии рейтингом.
Данные теоретические положения касаются чат бота распределения заданий студентам. В частности, могут быть сформулированы следующие правила:
1. Все студенты должны получать разные задания;
2. Отличники учёбы получают более сложные задания.
Суть приведённой выше схемы состоит в следующем. Чат-бот во время общения со студентом использует базу данных распределения заданий. Он может предложить студенту только те задания, которые не получили другие студенты. По окончании процесса распределения заданий, чат-бот запись в таблицу распределения заданий, содержащую идентификационный номер студента и номер задания. Также чат-бот использует таблицу успеваемости. На основе данной таблицы выявляет отличников и даёт им премиум задания.
Работа компонента понимания естественного языка содержит две стадии:
• Предварительная обработка;
• Векторизация.
Около 90% данных мира не структурированы. Они могут быть представлены в виде изображений, текстов, аудио и видео. Текст может быть представлен в различных формах (например, сообщения и знаки препинания). Также может присутствовать в виде HTML, документов и т. д. Эти данные никогда не бывают чистыми и состоят из большого количества шума. Нужно делать предварительную обработку для приведения их к корректной модели. Если это не сделано, то любые алгоритмы, построенные на основе таких данных, не будут работоспособны. Предварительная обработка включает в себя преобразование необработанных текстовых данных. Данные бывают противоречивыми, наполненными большим количеством шума и часто содержат ошибки. Предварительная обработка подготавливает необработанные текстовые данные для дальнейшей обработки. С другой стороны, предварительная обработка включает операции.
Опишем эти операции более подробно.
1. Преобразование текстовых данных в нижний регистр
Регистр слов не имеет содержательного значения, он представляет мусорную информацию для распознавания речи. По этой причине все слова должны быть переведены в нижний регистр.
2. Удаление знаков препинания
Точно также знаки препинания не содержат релевантной семантической информации.
3. Удаление стоп-слов
Стоп-слова это очень распространенные слова, которые не несут смысла или имеют меньшее значение по сравнению с к другими ключевыми словами. Если мы удалим стоп-слова, то мы можем сосредоточиться на важных ключевых словах. Если, например, мы имеем запрос: «Как разработать чат-бот с помощью Питон», то поисковая система пытается найти веб-страницы, содержащие «как», «чтобы», «разработать», «чат-бот», «использование», «Питон», и найдет намного больше страниц, содержащих термины «как» и «чтобы», чем страницы, которые содержат информацию о разработке чат-бота. Если удалить такие термины, поисковая система может сосредоточиться на поиске страниц, которые содержат ключевые слова: «разработка», «чат-бот», «python» — что бы более внимательно анализировать страницы, которые представляют реальный интерес. Точно так же мы можем удалить и редкие слова.
4. Стандартизация
Большая часть текстовых данных представлена в виде отзывов клиентов блогов или сообщений. Высока вероятность того, что люди будут использовать короткие слова и аббревиатуры. Это может помочь и

понимать семантику текста.

5. Токенизация

Токенизация, как процесс, так и сам термин, значительно эволюционировал с первых дней обработки естественного языка. Несмотря на значительный прогресс, не существует идеального метода обращения с токенизацией. Все варианты — от разделения пробелами выученными под словами и вплоть до байтов — имеют недостатки и сильные стороны. Множество требований могут принципиально противоречить друг другу, например требование простоты и устойчивости противоречит требованию точности. Расширение генеративных моделей с закрытым словарем на модели с открытым словарем, т. е. те, которые могут предсказывать и генерировать новые слова во время теста, несколько труднее, чем использовать открытый словарный запас на входная сторона, потому что должна быть возможность удержаться вероятностная масса для бесконечного множества предложений что содержать полностью слова. Существует вероятностный двухуровневый подход к открытому языку, который, по сути, увеличивает словарь. Для его реализации применяется рекуррентная нейронная сеть на уровне слов с закрытым словарным запасом настройка языковой модели путем регуляризации вложения слов для прогнозирования. Другой подход заключается в наличии более высоких уровней многослойный нейронной сети. Этот подход ведет к одному из множества способов, которыми мы можем на самом деле выучить границы слов и, таким образом, сегментировать. В модели связь между слоями происходит двунаправленно: нижняя сеть сообщает о своем конечном состоянии к более высокому; этот верхний уровень сообщает о своем новом состоянии нижнему уровню. Концептуально простой подход состоит в том, чтобы рассматривать сегментацию строки как скрытую переменную. Многие приложения обрабатывают текст более чем за один язык, самый очевидный пример - это система машинного перевода. В таких случаях можно было бы либо использовать (и потенциально выучить) токенизатор для каждого языка или один токенизатор для обоих языков (также позволяет обмениваться вложения по желанию). Основным фактором, ограничивающим принятие характера, моделей уровней заключается в том, что последовательности символов как правило, намного длиннее, чем их слово.

6. Стемминг

В информационном поиске и лингвистической морфологии стемминг — это процесс отделения суффикса от слова и сведения его к корневой форме, также известной как основная форма. Стемминг используется при обработке текста и естественного языка. Одним из инструментов, используемых для стемминга, является стеммер Портера, где корень может не совпадать с морфологическим. Алгоритм Стеммера описывает процесс удаления флективных окончаний и общих морфов из слов. Он также используется для нормализации текста в информационно-поисковых системах. Стеммер Портера менее агрессивен, чем стеммер Ланкастера, который обрезает большую часть допустимого текста. В лингвистической морфологии стемминг — это процесс нахождения корня или основы слова.

7. Лемматизация

Однако лемматизация заключается в том, чтобы найти лемму в наборе словарей, имеющих одинаковый смысл слова. Основа или словоформа могут быть производными или измененными. Таким образом, стеммер приближает большинство слов к их соответствующим категориям, что дает более высокие оценки классификации, чем эксперименты, включающие сопоставление слов в их исходной форме.

8. Синтаксическая коррекция

Проверка грамматики – это задача обнаружения и исправления грамматических ошибок в тексте. К настоящему времени было предложено и реализовано множество подходов. При написании текста люди могут делать ошибки. Поэтому очень важно уметь обнаруживать эти грамматические ошибки и исправлять их. Проверка грамматики человеком становится неудобной в такие моменты, когда человеческие ресурсы ограничены тем, что размер документа большой или проверка грамматики должна выполняться на регулярной основе. Поэтому было бы полезно автоматизировать процесс проверки грамматики. Инструмент проверки грамматики может обеспечить автоматическое обнаружение и исправление любого ошибочного, нетрадиционного или противоречивого использования базовой грамматики. Тенденция разработки таких инструментов развивалась с 80-х годов по настоящее время. При разработке схемы классификации синтаксических ошибок нужно учитывать следующие моменты:

• Более частые ошибки должны храниться в отдельных группах. Например, пять типов синтаксических ошибок являются наиболее частыми ошибками, возникающими в тексте, поэтому они классифицируются в отдельные группы. Точно так же очень распространены орфографические и пунктуационные ошибки.

• Ошибки должны быть разделены на основе того, как они делают текст недействительным. Например, синтаксическая ошибка делает текст недействительным из-за нарушения правил грамматики. Точно так же

ошибка в структуре предложения делает предложение недействительным из-за нарушения правил построения предложения, а орфографическая ошибка делает недействительным слово, если оно нарушает языковую орфографию.
• Некоторые ошибки обнаруживаются на уровне предложения, в то время как другие могут быть обнаружены на уровне слова, т. е. на уровне двух или трех слов. Например, нет необходимости проверять предложение целиком, чтобы обнаружить орфографические ошибки. Точно так же проверки слов до и после предлога было бы достаточно, чтобы обнаружить ошибку предлога, в то время как фрагменты могут быть обнаружены с использованием шаблона дерева синтаксического анализа полного предложения.
• Ошибки, которые являются более раздражающими и трудными для обнаружения, должны быть отделены от более простых. Например, орфографическая ошибка носит довольно формальный характер, и ее легко обнаружить с помощью программы проверки орфографии, тогда как обнаружение семантической ошибки требует реальных знаний.

1. Steven Bird, Ewan Klein, Edward Loper. Natural Language Processing with Python. Released June 2009 – 504 с.
2. Akshay Kulkarni (Author), Adarsha Shivananda. Natural Language Processing Recipes: Unlocking Text Data with Machine Learning and Deep Learning Using Python. 2019 – 231 с.
3. Гудфеллоу Я., Бенджио И., Курвилль А. − Глубокое обучение -: М.: Издательство «ДМК Пресс», 2017 г – 652 с.
4. Франсуа Шолле − Глубокое обучение на Python, -: П.: Издательство «Питер» 2018 г. – 400 с.
5. Д. М. Златопольский. Основы программирования на языке Python, -: М.: Издательство «ДМК» - 2017, 285 стр.