УДК 004.8

Помощь голосовых ассистентов в управлении и их развитие

Арсентьев Дмитрий Андреевич
доцент, к.т.н.
Бейбутов Марат Октаевич
студент профиля «Информационные системы и технологии»
факультет «Информационных технологий»
«Московский политехнический университет»
Россия, г. Москва

Аннотация. В современном обществе имеются множество голосовых ассистентов, но многие из них не имеют того лил иного функционала необходимого для узких специалистов. Они рассчитаны на массовый сегмент пользователей. Из-за этого возникает нехватка узкого функционала. Данная статья посвящена разработке нового голосового ассистента, направленного на пользование узкими специалистами и помощи их в управлении. Проект, который описывает эта статья, создан, как модернизируемый голосовой ассистент. А цель этого проекта — это создание скелета универсального голосового ассистента для практически безграничного списка задач. Это то, что необходимо пользователям для повышения комфорта работы.

Это пробудит новую волну создания голосовых ассистентов, направленные на узкие сегменты пользователей. Что способствует развитию данной ниши и созданию еще более массовых и продвинутых голосовых ассистентов.

А это в свою очередь поспособствует развитию искусственного интеллекта, машинного обучения и робототехники.

Ключевые слова: управление, искусственный интеллект, ИИ, машинное обучение, программирование, IT, голосовой ассистент.

Assistance of voice assistants in management and their development

Annotation. In modern society, there are many voice assistants, but many of them do not have this or that functionality necessary for narrow specialists. They are designed for a mass segment of users. Because of this, there is a shortage of narrow functionality. This article is devoted to the development of a new voice assistant aimed at using narrow specialists and helping them in management. The project that this article describes was created as an upgraded voice assistant. And the goal of this project is to create a skeleton of a universal voice assistant for an almost limitless list of tasks. This is what users need to improve their work comfort.

This will awaken a new wave of voice assistants aimed at narrow segments of users. Which contributes to the development of this niche and the creation of even more massive and advanced voice assistants.

And this, in turn, will contribute to the development of artificial intelligence, machine learning and robotics.

Keywords: management, artificial intelligence, AI, machine learning, programming, IT, voice assistant.

Введение. Голосовой ассистент – виртуальный помощник, который работает на основе искусственного интеллекта. Искусственный интеллект – это свойство искусственных интеллектуальных систем выполнять творческие функции, которые традиционно считаются прерогативой человека. Он распознает речь пользователя, может анализировать его ответы и выполняет команды пользователя. Данный проект создан, как модернизируемый голосовой ассистент для программистов. В проекте имеются одновременно несколько целей и задач. Основная цель – это создание скелета универсального голосового ассистента для практически безграничного списка задач. Дополнительная цель – это изучение библиотек языка, основ машинного обучения и работы датасетов. Библиотека в программировании — сборник подпрограмм или объектов, используемых для разработки программного обеспечения (ПО). С точки зрения операционной системы (ОС) и прикладного ПО, библиотеки разделяются на динамические и статические, также их называют библиотеки первого и второго уровня соответственно. Машинное обучение (ML) — это использование математических моделей данных, которые помогают компьютеру обучаться без непосредственных инструкций. Оно считается одной из форм искусственного интеллекта (ИИ). При машинном обучении с помощью алгоритмов выявляются закономерности в данных. Датасет — это механизм хранения информации, который предоставляет быстрый доступ к большим объемам данных. Данный голосовой ассистент был создан с самым востребованным функционалом, собранном при опросе нескольких IT-специалистов, но его универсальная архитектура позволяет его совершенствовать и добавлять функционал под новые нужды в любой момент. Под функционалом в данном проекте подразумеваются задачи, которые может выполнить голосовой ассистент. Архитектура программного обеспечения — совокупность важнейших решений об организации программной системы.

Архитектура включает:

- выбор структурных элементов и их интерфейсов, с помощью которых составлена система, а также их поведения в рамках сотрудничества структурных элементов;
- соединение выбранных элементов структуры и поведения во всё более крупные системы;
- архитектурный стиль, который направляет всю организацию — все элементы, их интерфейсы, их сотрудничество и их соединение.

Функционал ничем не ограничен, так как язык программирования Python имеет много open-source библиотек под всевозможные задачи.

Проблемы. Основная проблема в библиотеках, а точнее в их подборе. Многие библиотеки создавались под разные версии языка, а некоторые просто были несовместимы. А также из-за размеров этих библиотек проект невозможно сделать приложением (.exe), так как нет софта, который может собрать проект с таким большим количеством библиотек с их немалыми размерами. Следующие проблемы заключались в обработке запроса, в датасете, архитектуре кода и реализации некоторого функционала. С обработкой запроса было сразу несколько проблем. Основная заключалась в том, что для некоторого функционала такого, как поиск в браузере, изменение громкости, смена окон, закрытие окон, требовалось извлекать из запроса определенную информацию. Второстепенная была связана с датасетом. Она заключалась в том, что запрос на одну и туже задачу мог быть разный. Проблема архитектуры была в том, что она должна быть максимально простая, это позволит легко модернизировать исходный код, не меняя её саму. Исходный код — текст компьютерной программы на каком-либо языке программирования или языке разметки, который может быть прочтён человеком.

Преимущества. Но помимо проблем были и преимущества. Благодаря большому выбору библиотек большая часть функционала реализовывалась только с их помощью. А также сочетание нескольких библиотек позволяет упростить написание многих функций. Функция в Python — это фрагмент кода для повторного использования, который применяется для выполнения одного связанного действия. А простая архитектура позволяет быстро и просто добавлять новый функционал. Одним из преимуществ является дата-сет. В данном проекте он представлен как файл data-set.txt. Этот файл содержит в себе строки в виде «Запрос пользователя/Ответ бота». Запрос пользователя – это набор символов (слов), который пользователь задаёт программе(ПО). Такая конструкция позволяет легко добавлять новые запросы, а также просто их обрабатывать. А также на несколько схожих запросов используется один ответ голосового ассистента. А благодаря библиотеке Scikit-learn в датасет не нужно прописывать все возможные запросы пользователя. Scikit-learn (ранее известная, как scikits.learn, а также известная как sklearn) — это библиотека, предназначенная для машинного обучения, написанная на языке программирования Python и распространяемая в виде свободного программного обеспечения. Эта библиотека самостоятельно создает вектора запроса, ориентируясь на датасет. То есть задав один или два запроса в дата-сете эта библиотека сама определит схожие запросы.

Решения. Главная проблема подбора библиотек для данного проекта была решена путём проб и ошибок, а также чтением документаций. Как итог в проекте используются 12 библиотек, из них 8 являются библиотеками второго уровня. Библиотеками первого уровня (статические) являются: • OS – это библиотека, предоставляющая интерфейс для взаимодействия с базовой операционной системой, под управлением которой работает Python. • Random – этот модуль содержит ряд функций, использующих случайные числа. Он может вернуть случайное число, выбрать случайный элемент списка или

перемешать список случайным образом. Перемешанные списки могут быть выведены в строчку или в виде нумерованных или ненумерованных списков различных видов. • Time – содержит массу полезных методов для работы со временем. С его помощью можно получать информацию о текущей дате и времени с точностью до миллисекунд, выводить эти сведения в необходимом формате, а также управлять ходом выполнения программы, добавляя задержки по таймеру. • WebBrowser — это вшитый в Python модуль, который предоставляет собой высокоуровневый интерфейс, позволяющий просматривать веб-документы. Библиотеками второго уровня (динамические) являются: • SpeechRecognition: - пакет, позволяет создавать сценарии для доступа к микрофонам и обработки аудиофайлов с нуля. Библиотека SpeechRecognition действует как оболочка для нескольких популярных речевых API и, таким образом, является чрезвычайно гибкой. • gTTS – это библиотека Python и инструмент командной строки для взаимодействия с API преобразования текста в речь Google Translate. • Playsound – это модуль Python, с помощью которого пользователи могут воспроизводить звук в одной строке кода. Это кроссплатформенный модуль, который представляет собой единую функцию без каких-либо зависимостей для воспроизведения звуков и аудио. • Sound - это пакет для доступа python к звуковым данным на основе файлов. Создан с учетом научного использования. • Subprocess – это встроенный модуль Python, который можно использовать для создания новых процессов и взаимодействия с их входными и выходными потоками данных. • Pyautogui - это модуль, используемый для управления мышью и клавиатурой. Этот модуль используется для автоматизации задач нажатия клавиш и клавиатуры. Это минимальный набор библиотек для такого проекта. Поэтому решить проблему с оптимизацией для создания приложения (.exe). Максимальная оптимизация дала лишь небольшой результат. За счет подбора схожих по функционалу, но менее тяжелых библиотек, а также импортирования не всей библиотеки sklearn, а лишь некоторого её функционала, а точнее CountVectorizer и LogisticRegression, удалось сократить размер проекта на 10%. Как итог проект весит менее полу-гигабайта. CountVectorizer – это отличный инструмент, предоставляемый библиотекой scikit-learn на Python. Он используется для преобразования заданного текста в вектор на основе частоты (количества) каждого слова, встречающегося во всем тексте. Это полезно, когда у нас есть несколько таких текстов, и мы хотим преобразовать каждое слово в каждом тексте в векторы (для использования в дальнейшем анализе текста). LogisticRegression — это алгоритм машинного обучения, который используется для решения задачи бинарной классификации, то есть разделения данных на два класса. Она получила свое название благодаря тому, что использует логистическую функцию для прогнозирования вероятности принадлежности объекта к одному из классов. Далее нужно было решить проблему запроса пользователя. Так как эта проблема состоит из двух, решалась она в два этапа. Первый этап заключался в том, что необходимо было написать несколько алгоритмов по обработке

запроса. Первый из них это алгоритм очистки от ключевых слов. Он представлен в исходном коде как key_word_clean().

Этот алгоритм основан на двух списках(массивах) arr_comm и key_words, состоящих из слов запроса пользователя и заранее заданного списка слов, определённых как ключевые. Ключевые слова – это слова, которые задаются в запрос пользователя для определения действия. Например, в запросе «Найди в яндексе что-то», «Найди в яндексе» это ключевые слова, по которым бот распознаёт что требует пользователь, а «что-то» это уже не ключевое слово, которое далее передаётся как переменная no_key_word_comm для дальнейшего её использования. Второй из них это алгоритм для определения числовых значений. В данном случае числовые значения – это числа которые могут быть полезны для определенного функционала. Он представлен в исходном коде как get_digits Этот алгоритм основан на стандартном поиске при помощи перебора искомых значений в запросе пользователя. Искомые значения в данном случае – это цифры, которые после передаются в виде переменной c, которая далее используется для некоторого функционала. Проблема с архитектурой была решена способом функционального программирования. Функциональное программирование представляет собой методику написания программного обеспечения, в центре внимания которой находятся функции. В языке программирования Python функции — это объекты первого класса (объект первого класса – это сущность, которая может быть динамически создана, уничтожена, передана в функцию, возвращена как значение и обладать всеми правами, которыми обладают другие переменные в языке программирования), поэтому для них доступны такие операции:

- Присваивать объекты функций переменным в качестве значений.
- Используя аргументы, передавать объекты функций другим функциям в качестве входных данных.
- Хранить функции внутри структур данных, таких как словари.
- Использовать функции в качестве значений, возвращаемых от других функций. Этот способ позволил максимально упростить архитектуру программного обеспечения, что позволяет легко редактировать и модернизировать исходный код.

Заключение. Проект имеет как плюсы, так и минусы. Главным плюсом является простая архитектура, позволяющая легко модернизировать исходный код, что и является главной целью проекта. А также это является главным отличием от многих схожих проектов. Вторым плюсом является использование в проекте библиотеки Scikit-learn, что позволило создать удобный датасет, добавить машинное обучение и фактически создать небольшой искусственный интеллект. Третьим плюсом стал функционал. Он закрывает все базовые потребности пользователя. Что отличает его от распространенных голосовых ассистентов таких как Яндекс Алиса или Google Assistant. Хоть эти два гиганта имеют куда большие мощности, они не имеют того самого функционала, который может помочь сделать более удобное

управление операционной системой и устройством в целом. Четвертый плюс исходит из первых двух. Это неограниченная модернизация голосового ассистента. Это возможно благодаря практически неограниченным возможностям языка программирования Python. А этим Python обязан безграничным набором библиотек, которые открывают все эти возможности. Но, как и у всех проектов, у этого проекта есть и минусы. Главный это объём библиотек. Из-за таких библиотек распространение проекта в открытый доступ практически невозможно. Сейчас проект весит практически половину гигабайта и его уже нельзя собрать в обычное приложение. И на данный момент найдено только одно решение. Оно заключается в создании сервера с необходимыми для работы алгоритма библиотеками и файлами, но такое решение повлечет за собой большие изменение в архитектуре, что сильно усложнит её, а, следовательно, проект потеряет свою суть легко модернизируемого голосового ассистента. Итог таков, что цель проекта была достигнута и его можно считать завершенным

Список использованной литературы

1. Архитектура ИТ решений. Часть 1. Архитектура предприятия // EcTavrida URL: https://ec-tavrida.ru/arxitektura-it-reshenij-chast-1 (дата обращения: 14.07.2023).
2. Введение в архитектурные стримы // SOER.MEDIA URL: https://s0er.ru/documents/workbook/3171 (дата обращения: 14.07.2023).
3. Архитектура приложений и интеграции: гайд по основным понятиям простыми словами // Хабр URL: https://habr.com/ru/companies/itq_group/articles/705598/ (дата обращения: 14.07.2023).
4. Каштанов В.В., Дьяков В. Ф. Искусственный интеллект как ключевая технология цифровой трансформации бизнеса и экономики // Военно-экономический вестник. — 2019 №3-4. — URL: https://voenvestnik.ru/