САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Математико-механический факультет

Кафедра системного программирования

Рандомизированный алгоритм стохастической аппроксимации в задаче распознавания отдельных слов речи

Дипломная работа студента 544 группы

Шалымова Дмитрия Сергеевича

Научный руководитель

профессор ………………………………………………...О. Н. Граничин

Рецензент

аспирант ………..…………………………………………Д. А. Дыдычкин

"Допустить к защите"

Заведующий кафедрой

д. ф.-м. н., профессор ……………………………………А. Н. Терехов

Санкт-Петербург

2006 г.

РЕФЕРАТ

Объем дипломного проекта составляет 100 страниц, в нем содержится 5 таблиц, 6 иллюстраций, 15 приложений и 20 источников литературы.

Ключевые слова: программное обеспечение, веб-ориентированная система, голосовое управление, распознавание речи, обучаемый помощник.

Целью работы является разработка и программная реализация модульной обучаемой веб-ориентированной системы голосового управления, предоставляющей интерфейс для написания пользователями собственных голосовых функций.

Были изучены существующие аналоги, рассмотрены современные технологии распознавания речи и методы классификации текстовых документов. Приведено обоснование выбора и характеристика комплекса технических средств для реализации программного продукта и оценка его информационных характеристик.

Результатом дипломного проектирования стала система голосового управления «Джарвис», разработанная в двух версиях: в виде desktop-приложения под все современные операционные системы и в виде веб-приложения для использования в браузерах, основанные на движке webkit.

Областью применения спроектированной системы является

Все компоненты этого комплекса распространяются под свободными лицензиями, таким образом, Web-ИС «Выпускная работа» может функционировать без использования дорогостоящего проприетарного программного обеспечения

СОДЕРЖАНИЕ

1. Введение
2. Аналитический обзор современного состояния вопроса и обоснование актуальности темы
3. Основная часть, содержащая концепции и решения, сведения о методах, алгоритмах, программах (в целом и по подзадачам); используемые технологии, средства и инструменты разработки; результаты по подзадачам, результаты тестирования разработанных подсистем.
4. Основная части
5. -описание математических методов, обоснование вводимых ограничений и допущений, используемых при решении задачи;
6. -описание алгоритмов и возможного взаимодействия программ с другими программами;
7. -описание и обоснование выбора метода организации входных и выходных данных;
8. -обоснование выбора языка программирования;
9. -описание программ и их применения;
10. -руководство для программиста, программиста и оператора;
11. -описание языка программирования;
12. -характеристика требуемого для реализации комплекса технических средств.
13. Допускается оформлять программную документацию в виде отдельных документов, включенных в состав пояснительной записки:

описание программы, текст программы, описание применения, руководство программиста, описание языка, руководство оператора.

Титульник + аннотация + содержание + введение = 6

Аналитический обзор - 8

Основная часть + ОТ + Экономика– 40 + 10 + 10 = 60

СПИСОК СОКРАЩЕНИЙ

ПО - программное обеспечение

БД – база данных

СУБД - система управления базами данных

ОС – операционная система

API

JS -

Плагин

ВВЕДЕНИЕ

Десятилетиями ученые и инженеры искали способы, которые позволили бы людям общаться с компьютером так же, как они общаются между собой, а не заставлять пользователей приспосабливаться к языку, который понятен машине. Речь - это самое прямое и естественное средство общения между людьми, и как следствие, это должно быть самым естественным способом взаимодействия с машинами.

В настоящее время пользователями вычислительных машин и средств, оснащенных вычислительными машинами, становятся люди, не являющиеся специалистами в области программирования. Проблема речевого управления возникла, кроме того, в связи с тем, что в некоторых областях применения речь стала единственно возможным средством общения с техникой (в условиях перегрузок, темноты или резкого изменения освещенности, при занятости рук, чрезвычайной сосредоточенности внимания на объекте, который не позволяет отвлечься ни на секунду, и т.д.).

Повсеместное внедрение средств вычислительной техники в различные сферы человеческой деятельности делает разработку речевого интерфейса для взаимодействия с ЭВМ очень актуальной. Речевой интерфейс обладает рядом бесспорных и очевидных преимуществ:

* Преодоление физических недостатков. Для пользователей с нарушением зрения или нарушениями моторных функций зачастую трудно работать с компьютерами посредством клавиатуры и мыши. Приложения с качественным речевым интерфейсом предоставляют людям с данными заболеваниями более комфортный способ взаимодействия c машинами.
* Простота. Не требуются навыки работы с компьютером для взаимодействия с системой.
* Возможность одновременного использования стандартных и голосовых средств ввода информации.
* Свобода рук и глаз. Обеспечение мобильности. Речевой интерфейс позволяет человеку взаимодействовать с компьютером с помощью беспроводной гарнитуры, таким образом обеспечивается не зависимое от расположения человека взаимодействие.
* Удобство использования даже при неблагоприятных условиях. Акустические системы имеют большое преимущество когда затрудняется восприятие визуальной информации, например, темнота, слишком большое количество света, блики, вибрации.
* Эффективность. Средний работник офиса печатает со скоростью 50-70 слов в минуту. Однако системы распознавания речи позволяют обрабатывать 120 слов в минуту с высокой точностью. Таким образом, приложения с голосовым интерфейсом позволяют значительно увеличить производительность при написании заметок, отчетов и прочих документов.

Современность темы дипломной работы обусловлена тем, что рынок систем распознавания голоса рос и продолжает стремительно увеличиваться. По данным исследований BBC, размер рынка в 2011 году составлял 47 миллиардов долларов, 53 миллиарда в 2012 году и ожидается рост до 113 миллиардов к 2017 году, что составляет в общей сложности 16.2% прироста. Это обусловлено тем, что такие технологические гиганты как Google, Facebook, Apple и Microsoft продолжают инвестировать в развитие данных технологий. При этом большую часть рынка занимает потребительский сектор, что отражает увеличивающийся интерес у конечных пользователей современных компьютеров и мобильных устройств. Следует отметить, что так же увеличивается спрос на системы распознавания голоса в сфере здравоохранения. Росту также способствуют значительное увеличение вычислительной мощности и доступные объемы памяти среднего настольного персонального компьютера. Теперь нет никаких проблем с техническим обеспечением, необходимым для работы систем распознавания голоса. А увеличение количества поставщиков этой технологии и конкуренция снизили цены разработок до вполне приемлемого уровня.

Целью работы является разработка и программная реализация модульной обучаемой веб-ориентированной системы голосового управления, предоставляющей интерфейс для написания пользователями собственных голосовых функций.

Для достижения цели были поставлены следующие задачи:

* Произвести анализ предметной области, изучить существующие аналоги, определить состояние современного технического обеспечения для решения поставленной задачи, установить возможные проблемы и ограничения
* Разработать схему функционирования системы с голосовым управлением.
* Определить комплекс программных и аппаратных средств для реализации модульной системы с голосовым управлением.
* Разработать модульную систему с голосовым управлением.
* Реализовать систему плагинов для разработанной системы

Дипломная работа состоит из введения, аналитического обзора, основной части, экономического раздела, раздела охраны труда и безопасности жизнедеятельности, заключения.

В введении указывается цель и задачи для ее достижения, обосновывается современность и актуальность решаемой задачи.

В аналитическом обзоре производится обзор литературных источников для анализа предметной области изучаемой проблемы. Обосновывается актуальность выбранной темы и осуществляется выбор способа реализации поставленной перед автором задачи.

В основной части описываются алгоритмы, методы и вводимые ограничения. Производится обоснование выбора и характеристика языка программирования, а так же комплекса программных и аппаратных средств для реализации описанной системы. Приводится схема функционирования разработанной системы, руководство для программиста и оператора.

В экономическом разделе приводятся результаты обоснования принятых решений с точки зрения их экономической целесообразности. Приводятся расчеты себестоимости разработанного продукта и сроков окупаемости, капитальных и эксплуатационных затрат, его конкурентоспособности.

В разделе охраны труда и безопасности жизнедеятельности производится анализ условий труда на рабочем месте инженера-программиста и разрабатываются меры по их улучшению.

* 1. Аналитический обзор

Речь является основным средством человеческого общения. По причинам, начиная от любопытства возможности механической реализации механизмов человеческой речи и заканчивая желанием автоматизировать простые задачи, требующие взаимодействия человека с машиной, исследования в области распознавания и синтеза речи привлекли большое внимание ученых за последние 5 десятилетий. Желание автоматизировать простые задачи не является современным явлением, а уходит назад в многолетнюю историю.

Официальным годом рождения систем распознавания речи можно считать 1952 год. Причина тому – американская компания Bell Laboratories, крупный исследователь в области компьютерных и электронных систем. В этом году эта организация представила свое детище, систему названую Audrey [1]. Откровенно говоря, Audrey даже не обладала словарным запасам в привычном для нас понимании, она распознавало только цифры, сказанные одним голосом. Но и это еще не все. Система обладала рядом ограничений, невыполнение которых снижало точность Audrey до 60-70 процентов. Если все было сделано по требованиям, точность составляла порядка 90%. В современных реалиях это совсем не показатель. Однако был сделан большой шаг на пути понимания голоса человека машиной.

В 1962 году компания IBM продемонстрировала их детище - систему «Shoebox», которая понимала 16 слов на английском. Не трудно заметить, что на переход от цифры к слову понадобилось целых десять лет.

Лаборатории в США, Японии, Англии и СССР разработали еще несколько аппаратов, которые распознавали отдельные произнесенные звуки, расширив технологию распознавания речи поддержкой четырех гласных и девяти согласных звуков [2]. Звучали они не очень хорошо, но эти первые попытки дали впечатляющий старт, особенно если учитывать, насколько примитивными были компьютеры того времени.

#### В 1970-е годы системы распознавания речи сделали большие шаги благодаря интересу и спонсированию от министерства обороны США. Их программа DARPA Speech Understanding Research (SUR) с 1971 по 1976 год была одной из самой большой в истории распознавания речи, и помимо всего остального она отвечала за систему «Harpy» Университета Карнеги Меллона. «Harpy» понимала 1011 слов, что является средним словарным запасом трехлетнего ребенка. «Harpy» была значительной вехой, так как она представила более эффективный подход к поиску, называемый [Beam search](http://en.wikipedia.org/wiki/Beam_search), «демонстрируя сеть возможных предложений с конечным числом состояний» ( [Readings in Speech Recognition](http://books.google.com/books?id=iDHgboYRzmgC&pg=PA551&lpg=PA551&dq=darpa+harpy&source=bl&ots=ja6SEQTlgF&sig=YkBDHyTTK-8_0FnR_94sbsoS6IQ&hl=en&ei=1oupTpekCanZ0QHP6vy_Dg&sa=X&oi=book_result&ct=result&resnum=4&ved=0CDUQ6AEwAw#v=onepage&q=darpa%20harpy&f=false)). 70-е годы также отмечены еще несколькими вехами в данной технологии, например основанием первой коммерческой компании Threshold Technology, которая представила систему, которая могла интерпретировать различные голоса.

#### 1980-е: Распознавание речи оправдывает прогнозы

В следующей декаде благодаря новым подходам и технологиям словарный запас подобных систем вырос с нескольких сотен до нескольких тысяч слов и имел потенциал распознавания неограниченного количества слов. Одной из причин был новый статистический метод, больше известный как [скрытая марковская модель](http://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D1%8B%D1%82%D0%B0%D1%8F_%D0%BC%D0%B0%D1%80%D0%BA%D0%BE%D0%B2%D1%81%D0%BA%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C).   
  
Используя шаблоны для слов и звуковые паттерны, она рассматривала вероятность того, что неизвестные звуки могли быть словами. Эта база использовалась другими системами еще на протяжении двадцати лет ([Automatic Speech RecognitionA Brief History of the Technology Development](http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/354_LALI-ASRHistory-final-10-8.pdf)).   
  
С расширенным словарным запасом распознавание речи начало протаптывать себе дорожку в коммерческие приложения для бизнеса и специализированных отраслей, таких как медицина. Она даже вошла в дома обычных людей в 1987 году в виде куклы [Worlds of Wonder's Julie doll](http://www.squidoo.com/julie-talking-doll), которые дети могли натренировать, чтобы она распознавала их голос («Наконец-то кукла, которая тебя понимает»).  
  
  
  
Хоть ПО по распознаванию могло распознавать до 5000 слов, как, например, программа Kurzweil text-to-speech, в них был огромный недостаток — эти программы поддерживали дискретную надиктовку, то есть вы должны были останавливаться после каждого слова, чтобы программа его обработала.

#### 1990-е: Автоматическое распознавание речи идет в массы

В девяностых компьютеры наконец-то получили быстрые процессоры, и программы по распознаванию речи стали жизнеспособными.  
  
в 1990 году появилась первая общедоступная программа Dragon Dictate c ошеломляющей ценой 9000 долларов. Спустя семь лет вышла улучшенная версия — [Dragon NaturallySpeaking](http://www.pcworld.com/article/4823/naturallyspeaking_voice_recognition_leaps_forward.html). Приложение распознавало нормальную речь, поэтому вы могли говорить в обычном темпе около 100 слов в минуту. Но все равно, вы должны были тренировать программу в течении 45 минут перед использованием, и она имела все еще высокую цену в 695 долларов.  
  
Появление первого голосового портала VAL от BellSouth было в 1996 году. Это была первая интерактивная система распознавания речи, которая давала информацию, основываясь на том, что вы сказали в трубку телефона. VAL вымостила дорогу для всех неточных на то время голосовых меню, которые надоедали звонящим в следующие 15 лет.

#### 2000-е: Застой в распознавании речи — пока не появился Google

К 2001 году распознавание речи поднялось до 80-процентной точности, и прогресс технологии остановился. Системы распознавали работали отлично, когда языковая вселенная была ограниченной, но они до сих пор «догадывались» при помощи статистических моделей среди похожих слов, языковая вселенная росла вместе с ростом Интернета.   
  
Знали ли вы о том, что распознавание голоса и голосовые команды был встроены в Windows Vista и Mac Os? Большинство пользователей даже не догадывалось, что такая функциональность существует. Windows Speech Recognition и голосовые команды OS X были интересными, но [недостаточно точными](http://www.pcworld.com/article/126613/vistas_voice_recognition_stammers.html) и удобными, как клавиатура и мышь.  
  
Технология распознавания речи получила второе дыхание после одного важного события: появления приложения Google Voice Search для iPhone. Влияние этого приложения было значительным по двум причинам. Во-первых, телефоны и другие мобильные девайсы — это идеальные объекты для распознавания речи, и желание заменить крошечные экранные клавиатуры альтернативными методами ввода было очень велико. Во-вторых, у Google была возможность разгрузить этот процесс, используя свои облачные дата-центры, направив всю их мощь для крупномасштабного анализа данных для поиска совпадений между словами пользователей и огромного числа образцов голосовых запросов, которые они получали.   
  
Вкратце, узким местом распознавания речи всегда было доступность данных и возможность эффективной их обработки. Приложение же добавило к анализу данные миллиардов поисковых запросов, чтобы лучше предугадывать, что же вы сказали.  
  
В 2010 году Google добавил персональное распознавание в голосовой поиск телефонов под управлением Android. Программное обеспечение могло записывать голосовые запросы пользователей для построения более точной голосовой модели. Также компания добавила распознавание голоса в свой браузер Chrome в середине 2011 года. Помните как мы начали с 10 слов и продвинулись до нескольких тысяч? Так вот система Google теперь позволяет распознать 230 миллиардов слов.  
  
Потом появилась Siri. Так же, как и система Goggle Voice Search, она полагается на облачные вычисления. Она использует те данные, которые ей известны о тебе, чтобы сгенерировать вытекающий из контекста ответ и отвечает на твой запрос, как некая личность. Распознавание речи превратилось из инструмента в развлечение. 

#### Будущее: Точная и повсеместная речь

Бум приложений по распознаванию речи указывает на то, что время распознавания речи пришло, и мы можем ожидать огромного количества их и в будущем. Эти приложения не только позволят контролировать компьютер с помощью голоса или конвертировать голос в текст — они будут также способны различать разные языки, позволят выбирать голос помощника из различных вариантов.  
  
Вполне вероятно, что технология распознавания речи перейдет и на другие типы девайсов. Нетрудно представить, как в будущем мы будем управлять кофеварками, разговаривать с принтерами и говорить освещению, чтобы оно выключалось.

Одними из известнейших поставщиков ПО в этой области являются Google (Google Voice Search) и Apple (Siri). На отечественном рынке речевыми технологиями в основном занимается Центр Речевых Технологий.

обзор систем с голосовым управлением. В ходе анализа было выяснено, что в мире существует достаточное количество таких систем. Наиболее известными и продвинутыми голосовыми помощниками на текущий момент являются iPhone's Siri, Android's Google Now и Windows 10's Cortana.  
Чем же отличается разрабатываемая система от остальных?

* Кроссплатформенность. Прежде всего, она не привязана к конкретным аппаратным устройствам или операционным системам, а может функционировать как на мобильных устройствах в виде веб-приложения, так и на устройствах под управлением Windows, Linux или OS X в виде desktop-приложения.
* Офлайн. Благодаря стеку используемых технологий система не требует доступа к интернету для выполнения ваших команд.
* Самообучаемость. То есть ассистент буквально умеет обучаться новым командам от своего владельца. Если система не поняла чего-то, то она всегда спросит вас «Что с этим делать?», а вы сможете направить ее, сказав какое из действий, приведенных в подсказке, выполнить.
* Система плагинов. Система предоставляет простое, удобное и вместе с тем, позволяющее использовать все возможности и преимущества фреймворка AngularJS, API для создания собственных голосовых функций (плагинов). Реализована система управления плагинами: система предоставляет интерфейс для доступа к списку всех доступных плагинов и автоматизирует процесс установки / обновления / удаления их по клику без каких-либо дополнительных действий.

Вы когда-нибудь задумывались о том, как работают те же Siri и Cortana? Ваш голос превращается в текст, потом происходит какая-то магия, и программа выполняет ваш запрос. Или не выполняет, если не поняла, что вы имели в виду. Система предоставляет возможность заглянуть внутрь этой «магии» и создать свои собственные голосовые функции. Причём их возможности становятся действительно безграничными.

* Русский язык. На сегодняшний день все рассмотренные системы в первую очередь ориентированы на англоязычное население, а некоторые и вовсе поддерживают только его. В связи с этим новая функциональность появляется с существенной задержкой для русскоговорящих пользователей. Таким образом лучшей альтернативой может стать стороннее решение, а именно то, что изначально создавалось для русского языка.
* Мультиязычность. Несмотря на изначальную направленность разработанной системы на русскоговорящих пользователей, алгоритм функционирования системы реализован таким образом, что система будет так же хорошо выполнять собственные голосовые функции иноязычных пользователей.
  1. Анализ методов реализации веб-ориентированных систем с голосовым управлением.

В общем случае система с голосовым управлением может быть представлена в виде совокупности следующих подсистем:

* система распознавания речи
* система классификации текста
* система извлечения ключевых слов из текста
* база знаний
* система принятия решений
* агент
  + 1. Анализ систем распознавания речи

Дипломная работа автора не предполагает реализацию собственной системы распознавания речи, так как это является очень сложной, трудоемкой и ресурсозатратной задачей, которую сложно выполнить в рамках данной работы. В связи с этим анализ представляет собой обзор уже существующих систем, с целью выбрать наиболее оптимальную для разрабатываемого продукта с возможностью использовать ее на основе лицензионного соглашения.

Были рассмотрены наиболее известные речевые системы преобразования голоса в текст, которые имеют открытое API для интеграции с разрабатываемой систему.

Прежде всего, исходя из требований к разрабатываемой системе, следует отметить основные характеристики, по которым производился выбор оптимальной аудио-системы распознавания речи:

* Поддержка русского языка
* Мультиязычность
* Доступность на всех ОС
* Ориентация на короткие фразы (3-7 слов)
* Скорость и точность обработки
* Высокие нагрузки
* Надежность и отказоустойчивость
* Ограничения на количество обращений
* Сложность интеграции

На сегодняшний день существует достаточно большой выбор среди систем распознавания речи, но лишь немногие соответствуют требованиям, представленным выше. Рассмотрим их:

1. Yandex SpeechKit

Прежде всего, взаимодействие с Yandex SpeechKit ведется через HTTP API. Из этого следует, что технология является доступной для всех платформ и операционных систем, а затраты на развертывание и поддержку собственной инфраструктуры – минимальны. Доля верно распознанных русских слов колеблется в диапазоне от 82 до 95%. Распознавание может происходить в реальном времени одновременно с передачей звуковых данных. Задержка от окончания отправки данных до получения результата распознавания не превышает 1 секунды. Инфраструктура SpeechKit спроектирована с учетом высоких нагрузок, что обеспечивает доступность и безотказную работу системы даже при большом количестве одновременных обращений. Кроме того, технология от Yandex обеспечивает простоту интерфейсов, высокую доступность и надежность. Так же, стоит отметить, что данная технология является одной из лучших для распознавания русского и турецкого языка, однако остальные языки она не поддерживает.

1. Google Cloud Speech

Google Cloud Speech, так же как и Yandex SpeechKit, предоставляет разработчикам удобный для использования web API для конвертации аудио в текст с помощью мощных моделей нейронных сетей. Однако, в отличии от технологии Яндекса, Google Cloud Speech распознает более 80 языков. Данная технология обладает так же рядом других особенностей: обработка шума и потоковая передача данных, т.е. возвращение промежуточных результатов распознавания как только они становятся доступны еще во время разговора. Минусом данной технологии является ограничение на количество запросов: до 10 000 в сутки.

1. HTML5 Speech Recognition

HTML5 Speech Recognition является одной из малоизвестных составляющих спецификации HTML5, которую сейчас поддерживают только браузеры, основанные на Webkit. Таким образом, будучи технологией, встроенной в браузеры, она не накладывает каких-либо ограничений на количество запросов в сутки и даже не требует подключения к сети Интернет. Данная технология способна распознавать как непрерывную речь, так и одиночные фразы, а так же работает с большим количеством языков, включая русский. Соответственно ее минусом является то, что не все современные браузеры ее поддерживают.

Стоит отметить, что в HTML5 спецификации есть еще одна малоизвестная технология - Speech Synthesis, которая позволяет выполнить обратную операцию: синтезировать речь, используя большое разнообразие голосов.

Таким образом, рассмотрев наиболее подходящие, для поставленной задачи, системы распознавания речи, можно сделать вывод, что по своей библиотеке данных наиболее точным следует считать продукт Google Cloud Speech. Он более всего подходит для задач распознавания на базе разрабатываемой системы. Однако необходимо заметить, что данный инструментарий имеет ограничение на максимальное количество запросов в сутки, что является критическим недостатком. В то же время у технологии от Яндекса таким недостатком является поддержка слишком малого количества языком. Следовательно, наиболее оптимальным выбором будет являтся HTML5 Speech Recognition, которая изначально интегрирована в браузер, имеет высокую точность и скорость распознавания, поддерживает множество языком, не накладывает каких-либо ограничений по количеству запросов, а так же может работать с непрерывным речевым потоком

Система классификации текста

После того как речь была распознана, необходимо произвести интеллектуальный анализ текст, а именно – классификацию. Задача классификации - отнесение объекта к одной из заранее определенных категорий на основании его формализованных признаков.

Перед тем, как приступать к любой задаче классификации, одна из наиболее фундаментальных процедур, которую необходимо выполнить, это выбрать представление документа и отобрать признаки. Хоть отбор признаков и применяется довольно часто в других задачах классификации, он особенно важен в задаче классификации текстов ввиду высокой размерности (большого количества признаков) и наличия нерелевантных (шумовых) признаков. В большинстве случаев, представление текста происходит одним из двух способов. Первый – документ как набор слов, в котором документу сопоставляются слова и частота их встречаемости в нем. Такое представление независимо от порядка слов, в котором они встречаются в тексте. Второй метод состоит в представлении текста собственно как набор строк, в котором документ является последовательностью слов. Большинство алгоритмов классификации текстов используют первое представление ввиду его простоты и удобства для задач классификации. Наиболее популярными способами отбора признаков являются удаление стоп-слов и стемминг. При удалении стоп-слов, мы определяем общие слова документов, которые не являются специфичными или разделяющими для разных классов. При стемминге, разные формы одного слова приводятся к одному виду. Например, так объединяются слова разного рода/формы/времени/падежа и пр. Подобная обработка — уже устоявшийся способ уменьшения размерности задачи и улучшения производительности.

Джарвису необходимо знать вероятности того что классифицируемый документ относится к некоторому классу и в случае если она ниже определенного значения предлагать пользователю самому соотнести документ с некоторым классом. Так же необходимо было, чтобы текст внутри классификатора обрабатывался правильно, с учетом особенностей каждого языка, в частности русского.

Так как не хватает времени на разработку нейронных сетей бла бла бла…

Были рассмотрены реализации различных алгоритмов классификации текста: kNN, Naive Bayes, логическая регрессия, n-грамм, SVM, дельта TF-IDF для SVM

Машина опорных векторов (Support Vectors Machine, SVM) была предложена для числовых данных. Но ее можно применять и для классификации документов. Главным принципом SVM является определение разделителя в искомом пространстве, который разделяет классы наилучшим образом. Т.е. SVM умеет определять принадлежность документов только к одному из двух классов. Следовательно, чтобы классифицировать документ на принадлежность к одному из N классов, потребуется N классификаторов для каждого класса. Для распознавания текста машина опорных векторов предполагает создание словаря – списка всех слов в обучающей выборки и представление документов в виде вектора, где i-элемент вектора это мера вхождения i-го слова словаря в документ. По мере обучения классификатора, получаются вектора большой размерности, большинство элементов которых – нули. Таким образом данный метод достаточно медлителен, особенно для данных высокой размерности, таких, как текстовые документы.

Наивный байесовский классификатор

Модель наивного Баера показала достаточную эффективность

Если классов всего два («спам / не спам», «давать кредит / не давать кредит», «красное / черное»), то задача называется бинарной классификацией. Если классов несколько — многоклассовая (мультиклассовая) классификация. Также могут иметься образцы каждого класса — объекты, про которые заранее известно к какому классу они принадлежат. Такие задачи называют *обучением с учителем*, а известные данные называются *обучающей выборкой*.

В конечном счете, после апробирования более 15 моделей, реализованных под платформу Node.js, путем кросс-проверки на небольшой выборке данных, лучший результат показала BrainJS - имплементация на языке JavasScript простой нейронной сети типа многослойный перцептрон, продемонстрировав порядка 85% точности. Такой результат был достигнут во многом благодаря предварительной обработки текста с помощью библиотеки Natural. Так как BrainJS позволяет обучать нейронную сеть только массивами или хэшами чисел от 0 до 1, текст, перед тем как попасть в нейронную сеть, проходит следующие стадии предварительной обработки:

Обучение нейронной сети:

1. Токенизация – разбиение текста на слова

tokenize('Ну и какой был в этом смысл?')

// -> [ 'Ну', 'и', 'какой', 'был', 'в', 'этом', 'смысл']

tokenize('Какая же самая высокая гора в мире?')

// -> [ 'Какая', 'же', 'самая', 'высокая', 'гора', 'в', 'мире']

1. Стемминг и удаление стоп слов

stem([ 'Ну', 'и', 'какой', 'был', 'в', 'этом', 'смысл' ])

// -> [ 'Как', 'эт', 'смысл' ]

stem([ 'Какая', 'же', 'самая', 'высокая', 'гора', 'в', 'мире' ])

// -> [ 'Как', 'самый', 'высок', 'гор', 'мир' ]

1. Объединение без пересечения

join([ 'Как', 'эт', 'смысл' ], [ 'Как', 'самый', 'высок', 'гор', 'мир' ])

// -> [ 'Как', 'эт', 'смысл', 'самый', 'высок', 'гор', 'мир' ]

1. Featurize

// [ 'Как', 'эт', 'смысл', 'самый', 'высок', 'гор', 'мир' ]

features([ 'Как', 'эт', 'смысл' ])

// -> [ 1, 1, 1, 0, 0, 0, 0 ]

features([ 'самый', 'высок', 'гор', 'мир' ])

// -> [ 0, 0, 0, 1, 1, 1, 1 ]

1. Обучение нейронной сети

var net = new brain.NeuralNetwork();

// [ 'Как', 'эт', 'смысл', 'самый', 'высок', 'гор', 'мир' ]

net.train([

// [ 'Как', 'эт', 'смысл' ]

{ input: [ 1, 1, 1, 0, 0, 0, 0 ], output: { 'сорок два': 1 } },

// [ 'Как', 'самый', 'высок', 'гор', 'мир' ]

{ input: [ 1, 0, 0, 1, 1, 1, 1 ], output: { 'Эверест': 1 } }

]);

Классификация текста:

1) tokenizeAndStem('Можешь сказать какая самая высокая гора в мире')

// -> [ 'можеш', 'сказ', 'как', 'самый', 'высок', 'гор', 'мир' ]

2) features([ 'можеш', 'сказ', 'как', 'самый', 'высок', 'гор', 'мир' ])

// -> [ 1, 1, 1, 0, 0, 0, 0 ]

3) net.run([ 1, 1, 1, 0, 0, 0, 0 ]);

// -> { Эверест: 0.932615111520198, "сорок два": 0.0695108432441562 }  
  
По-умолчанию, классификатор игнорирует русские слова, хотя поддержка русского языка в проекте есть. Чтобы заставить классификатор понимать русский язык нужно проинициализировать классификатор, передав в него стиммер для русского языка, заменив таким образом используемый по-умолчанию английский стиммер.

Обучение нейронной сети является ресурсозатратным процессом, однако данная библиотека позволяет сохранять результаты обучения и соответственно загружать, что избавляет от необходимости при каждом запуске повторно тратить время на ее обучение.

Система извлечения ключевых параметров текста

Сравнительно с системой классификации текста, механизм извлечения ключевых параметров довольно примитивна. Исходя из предположения, что классифицируемое предложение, с определенной долей вероятности, является схожим с уже изученным предложением, то механизм представляет собой сравнение слов рядом с меткой. Например, если классифицируемое предложение «Не знаешь, холодно ли в Лондоне» с меткой «Лондон» будет сопоставлено с предложением из обучаемой выборки «Скажи пожалуйста, холодно ли в Севастополе?», то механизм пометит слова после «холодно ли» как местоположение. Разумеется, если предложение будет сопоставлено с несколькими предложениями из обучаемой выборки, то алгоритм пометит то слово, возле которого будет найдено больше всего совпадений. Хоть это и не самый идеальный алгоритм, он работает достаточно хорошо благодаря точности классификации естественного языка.

База знаний

Основное отличие Джарвиса от остальных подобных систем заключается в том, что он предоставляет пользователям возможность самостоятельно обучать ее. Для этого Джарвису необходимо где-то хранить тренировочные данные. Так как формат данных для обучения нейронной сети максимально прост: это текст и соответствующие тексту класс и метка, и никаких дополнительных данных и связей между ними не предвидится, то самым разумным решением, по мнению автора, будет использовать легковесную базу данных DiskDB, основанную на файловой системе и позволяющую хранить данные в json-формате.

Система принятия решений

После того как классификатор

* 1. Разработка алгоритма функционирования веб-ориентированной системы с голосовым управлением

Если вкратце, алгоритм функционирования разрабатываемой системы выглядит так :D



Рисунок 3 – Алгоритм функционирования системы с голосовым управлением

* 1. Обоснование выбора и характеристика комплекса технических средств для реализации ИС

Для разработки серверного веб-приложения была выбрана платформа Node.js - среда выполнения JavaScript, основанная на движке V8 и предназначенная для создания масштабируемых распределённых сетевых приложений, таких как веб-сервер. Так же был использован Express.js – минималистичный и гибкий Node.js фреймворк, который обеспечивает широкий набор функций для веб-приложений.

Для реализации desktop-версии приложения был использован Electron.

Разработка десктопных приложений является достаточно сложной задачей, так как требуется заботится о таких вещах как упаковка и установка приложения, управление обновлениями, а так же о дийзане приложения и портировании его под различные операционные системы. Поддержка различных платформ обычно предполагает использование целого набора специализированных инструментов.

Electron – это фреймворк от компании GitHub, который позволяет создавать нативные кроссплатформенные приложения с использованием исключительно веб-технологий, таких как: JavaScript, HTML и CSS. Он заботится о всех сложных нюансах разработки десктопных приложений, предоставляя унифицированный интерфейс для взаимодействия с различными операционными системами и позволяя сфокусироваться на сути самого приложения. В добавок к своему собственному набору API, Electron включает в себя весь набор Chromium API, все встроенные модули Node.js и полностью поддерживает модули третьих лиц. Electron использует веб-страницы для создания пользовательского интерфейса, что снимает присущие десктопным приложениям ограничения в дизайне. В данном случае мы не ограничены системным набором элементов интерфейса нашей платформы и можем создавать нужные нам UI элементы с помощью веб-технологий. А учитывая, что данный фреймворк использует один из самых продвинутых браузеров, мы можем использовать самые новые веб-технологии. Кроме того, выбор данной технологии обусловлен так же тем, что дальнейшее портирование десктопного приложения на веб-сервер Node.js займет минимум затрат по причине того, что Electon сам использует Node.js для работы с десктопной средой.

Для реализации клиентской части как десктопного так и серверного приложения был выбран фреймворк Angular.js. AngularJS – это производительный и гибкий open source фреймворк для создания веб приложений. AngularJS предназначен для построения single-page веб приложений и является набором JavaScript функций для организации кода на стороне клиента. В основе Angular JS лежит шаблон проектирования Model View Controller, что дает ряд преимуществ при разработке и тестировании приложения. Фреймворк адаптирует и расширяет традиционный HTML, чтобы обеспечить двустороннюю привязку данных для динамического контента, что позволяет автоматически синхронизировать модель и представление. В результате AngularJS уменьшает роль DOM-манипуляций и улучшает тестируемость.

ВЫВОДЫ

Когда голосовой помощник в полной мере станет понимать естественный язык?

Для этого, конечно, потребуется время. Развитие голосовых помощников, в этом плане, станет постоянным процессом, так как любой язык, являясь живой материей, приносит все новые слова и парадигмы, которым нужно время для интеграции. Необходимо также, чтобы технологии распознавания акцентов работали как надо.

Я задумался над тем, чтобы сделать для себя RSS-клиент с обучаемым фильтром статей на NodeJS. В принципе, под ноду есть готовые RSS ридеры, есть готовые нейронные сети с классификаторами, так что написать какой-то прототип мне показалось не особенно сложной задачей.

В конечном счете, после апробирования реализаций более 15 моделей под Node.js, включая многослойные нейронные сети на примере BrainJS и Node-fann, в результате кросс-проверки на небольшой выборке данных, лучший результат показала библиотека Natural, которая использует обычный наивный байесовский классификатор с корнями слов и POS-тегами, дав порядка 85%.

<http://speech-soft.ru/info/raspoznavanie-golosa> (продолжение)

Нерешенные задачи и взгляд в будущее.  
Три основных барьера стоят на пути развития систем распознавание речи:

1. большие объемы словарей  
2. шаблоны непрерывной речи  
3. различные акценты и произношения

Хотя в этой области и достигнуты существенные успехи, тем не менее, системы распознавания еще весьма далеки по своим возможностям от человеческих.

[1] K. H. Davis, R. Biddulph and S. Balashek, “Automatic recognition of spoken digits”, J. Acoust. Soc. Am., 24, pp. 637-642 (1952).

[2] J. Suzuki and K. Nakata, “Recognition of Japanese vowels – preliminary to the recognition of speech”, J. Radio Res. Lab, 37, pp. 193-212 (1961).