Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра технологий
программирования

ОПРЕДЕЛЕНИЕ ДЕМОГРАФИЧЕСКИХ ХАРАКТЕРИСТИК ПО ФОТОГРАФИИ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Белоногова Никиты Александровича

Научный руководитель
доцент, к. ф.-м. н. 10.03.2017 А. А. Кузнецов

Заведующий кафедрой
к.ф.-м.н. 10.03.2017 И. А. Батраева

Саратов 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Сегодня глубинное обучение используется в совершенно разных сферах, но наибольшее своё распространение получила в области обработки изображений.

Функции распознавания лиц существует уже давно, но они требуют ввода большого количества данных и точной настройки оборудования. Поэтому требуется внедрение современных технологий и методов постраения инфраструктуры приложения и поиск более простых и удобных способов общения с пользователем.

Целью данной работы является ознакомление с современными методами и технологиями разработки программного обеспечения, а так же более подробное изучение методов обработки, распознавания и классификации изображений.

Задачей данной работы является реализация приложения для определения демографических характеристик человека по фотографии. Для этого необходимо:

1. получить выборку фотографий людей и их характеристик из сервиса Google+;
2. построить репрезентативный вектор для каждой каждой фотографии;
3. реализовать систему классификации на основе этих векторов;
4. реализовать сервис для удобной работы с клиентом;
5. провести оценку точности и полноты работы классификатора.

1 Основные понятия и представления для реализации приложения

В этой представлен обзор современных средств и методов разработки программного обеспечения, история развития глубинного обучения, инфраструктур приложений и чатботов.

Когда программируемые компьютеры только начинали своё развитие, люди уже задавались вопросом, могут ли такие машины стать разумными. Сегодня искусственный интеллект — это процветающее поле со многими практическими приложениями и активными темами исследований. Мы полагаемся на интеллектуальное программное обеспечение в автоматизации рутинного труда, понимания речи или изображений, постановке диагнозов в медицине и поддержке фундаментальных научных исследований.

Глубинное обучение Глубинное обучение — это особый вид машинного обучения. Чтобы хорошо понимать глубокое обучение, нужно иметь четкое представление об основных принципах машинного обучения.

Во время зарождения искусственный интеллект быстро решал проблемы относительно прямолинейны для компьютеров — проблемы, которые могут быть описаны списком формальных математических правил. Истинным предназначением искусственного интеллекта оказалось решение задач, которые легко людям выполнять, но трудно описать формальным языком, которые мы решаем интуитивно, например, распознавание устных слов или лиц.

Технологии глубинного обучения — это целая группа технологий и решений. Другой пример — это сверточные сети, которые как раз используются для описания того, что видит сеть. Но самый популярный представитель технологий данного класса — это все-таки сверточные нейронные сети ЛяКуна. Они построены по аналогии с принципами работы зрительной коры мозга кошки, в которой были открыты так называемые простые клетки, реагирующие на прямые линии под разными углами, и сложные — реакция которых связана с активацией определенного набора простых клеток.

Несмотря на то, что технологии глубинного обучения уверенно справляются с задачами по распознаванию речи и изображений и обладают значительным коммерческим потенциалом, для них есть немалое число ограничений. Они требуют ввода большого количества данных и точной настройки

оборудования. Поэтому необходимо переходить на новые технологии и методы постраения инфраструктуры приложения.

Программное обеспечение как услуга Программное обеспечение как услуга (SaaS) — это модель распространения программного обеспечения, в которой сторонний поставщик размещает приложения и делает их доступными для клиентов через Интернет.

Другой не менее важной состовляющей модели SaaS является перевод всей вычислительной нагрузки в сторонние сервисы или облака. Одним из наиболее подходящий способов общения с клиентом в этом случае являются чатботы.

Чатботы Чатботы прошли долгий путь. Одним из первых виртуальных собеседников была программа Элиза, созданная в 1966 году Джозефом Вейзенбаумом. Элиза пародировала речевое поведение психотерапевта, реализуя технику активного слушания: переспрашивала пользователя и использовала фразы типа «Пожалуйста, продолжайте». Был проведен эксперимент, во время которого программа общалась со случайно выбранными пользователями. Впервые некоторые люди не смогли догадаться, что с ними общалась машина. И хотя со временем боты значительно эволюционировали, долгие годы они использовались лишь для имитации философских человеческих диалогов. Системы, основанные на шаблонах общения, могут поддерживать беседу длительное время, но практического смысла в этом мало.

2 Реализация приложения

В этой главе подробно описаны использованные технологи. Представлен обзор современных средств и методов разработки программного обеспечениюю.

2.1 Технологии

В этой главе описаны использованные технологи. Представлен обзор современных средств и методов разработки программного обеспечениюю.

В качестве языка программирования был выбран Python так как скорость выполнения программы не является критичной величиной и по сути все самые ресурсоёмкие задачи будут выполнятся внутри библиотек написанных на более быстродейственных языках.

Python Python — один из самых популярных языков программирования в мире, в свежем рейтинге TIOBE [1] он занимает четвёртое место. Python [2] это язык программирования общего назначения, нацеленный в первую очередь на повышение продуктивности самого программиста, нежели кода, который он пишет. На Python можно написать практически что угодно (веб и настольные приложения, игры, скрипты по автоматизации, комплексные системы расчёта, системы управления жизнеобеспечением и многое многое другое) без ощутимых проблем. Более того, порог вхождения низкий, а код во многом лаконичный и понятный даже тому, кто никогда на нём не писал. За счёт простоты кода, дальнейшее сопровождение программ, написанных на Python, становится легче и приятнее по сравнению с Java или C++. А с точки зрения бизнеса это влечёт за собой сокращение расходов и увеличение производительности труда сотрудников.

OpenCV OpenCV (Open Source Computer Vision Library) [3] представляет собой библиотеку программного обеспечения для компьютерного зрения и компьютерного обучения с открытым исходным кодом. OpenCV был создан для обеспечения общей инфраструктуры приложений для компьютерного зрения и ускорения использования восприятия машины в коммерческих продуктах.

Существует специальная архитектура искусственных нейронных сетей, предложенная Яном Лекуном и нацеленная на эффективное распознавание

изображений.

Свёрточные нейронные сети Свёрточные нейронные сети — это такие сети, где основным структурным элементом обучения является группа (сочетание) нейронов (обычно квадрат 3×3,10×10 и т.д.), а не один. И на каждом уровне сети обучаются десятки таких групп. Сеть находит такие сочетания нейронов, которые максимизируют информацию об изображении. На первом уровне сеть извлекает самые базовые, структурно простые элементы картинки — можно сказать, строительные единицы: границы, штрихи, отрезки, контрасты. Повыше — уже устойчивые комбинации элементов первого уровня, и так далее вверх по цепочке. Хочу ещё раз отдельно подчеркнуть главную особенность глубокого обучения: сети сами формируют эти элементы и решают, какие из них более важный, а какие — нет. Это важно, так как в области машинного обучения, создание признаков — является ключевым и сейчас мы переходим на этап, когда компьютер сам учится создавать и отбирать признаки. Машина сама выделяет иерархию информативных признаков. Общий пример работы свёрточной нейронной сети показан на рисунке.

Существуют целые фреймворки для работы с машинным обучением. Torch — один из них.

Torch Torch — это научная вычислительная среда с широкой поддержкой алгоритмов машинного обучения, у которой на первом месте вычисления на GPU. [4] Torch прост в использовании и эффективен благодаря легким и быстрым языкам сценариев LuaJIT и базовой реализации C / CUDA.

CUDA — это параллельная вычислительная платформа и модель программирования, изобретенная компанией NVIDIA. [5] CUDA позволяет значительно увеличить вычислительную производительность за счет использования мощности графического процессора (GPU). На миллионах графических процессоров, поддерживающих CUDA, на сегодняшний день разработчики программного обеспечения, ученые и исследователи используют GPU-ускоренные вычисления для широкомасштабных приложений.

Цель Torch — обеспечить максимальную гибкость и скорость в построении ваших научных алгоритмов, делая процесс чрезвычайно простым. Torch поставляется с большой экосистемой пакетов, ориентированных на сообще-

ства, машинное обучение, компьютерное зрение, обработку сигналов, параллельную обработку изображений, видео, аудио, сетевое взаимодействие и другие, и основывается на сообществе Lua.

На основе Torch и OpenCV написана с библиотека для распознавания лиц OpenFace.

OpenFace OpenFace — это бесплатная библиотека с открытым исходным кодом для распознавания лиц с помощью глубоких нейронных сетей. [6]

Google+ API API Google+ — это программный интерфейс для Google+. Можно использовать API для интеграции своего приложения или веб-сайта с Google+. Это позволяет пользователям подключаться друг к другу для максимального взаимодействия с помощью функций Google+ из вашего приложения.

Для языка Python написано очень обширное множество различных фреймворков и библиотек. Приложение не будет отличаться особой Поэтому в качаестве фреймворка для написания веб-приложения блыл выбран микрофреймворк Flask.

Flask Flask — фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2. [7,8] Относится к категории так называемых микрофреймворков — минималистичных каркасов веб-приложений, сознательно предоставляющих лишь самые базовые возможности.

2.2 Основные модули и сервисы

Структура приложения состоит из следующих сервисов:

— `MyFlaskApp`;
— `MyTelegramBot`.

Клиент сможет взаимодействовать с приложением через браузер, так и через месенджер Telegram. В свою очередь сервис `MyTelegramBot` пересылает запросы на серверную часть, то есть на сервис `MyFlaskApp`. Схема структуры показана на Рис. 1.

Структура серверной части приложения `MyFlaskApp` состоит из следующих модулей:

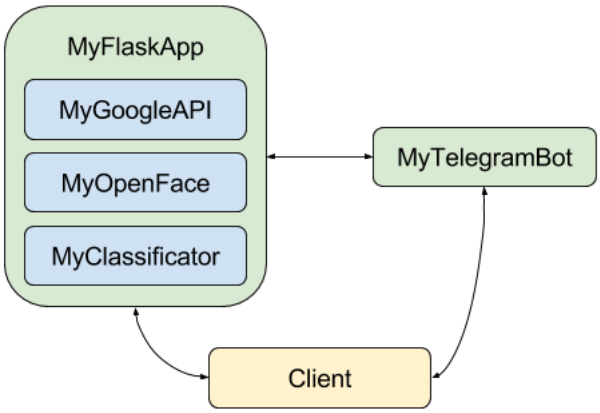

Рисунок 1 – Схема структуры

1. `MyGoogleAPI` — работа с Google api.
2. `MyOpenFace` — работа с библиотеками OpenCV и OpenFace.
3. `MyClassificator` — работа с библиотекой sklearn, сохранение данных.

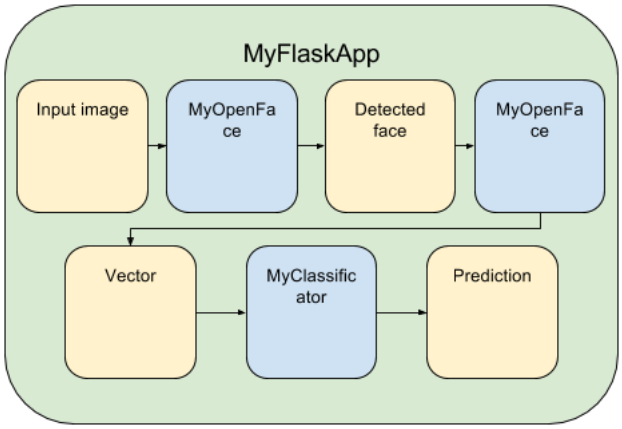

Рисунок 2 – Схема общего алгоритма работы серверной части приложения

Сервис MyFlaskApp Серверная часть приложения написана с помощью фреймворка Flask. Она представляет из себя веб приложение, которое поддерживает два маршрута с запросами типа GET и POST.

MyGoogleAPI Модуль `MyGoogleAPI` будет отвечать за возможность получения информации о пользователях с сервиса Google+. Модуль будет использовать API key.

MyOpenFace У этого модуля есть две основных функции. Первая из них это функция face_align — выравнивание лица. Предварительная обработка,

подготовка изображения на вход нейронной сети. Лица изменяются до одинакового размера (например, 96x96) и преобразуются для создания ориентиров (например, глаз и носа) в одном месте на каждом изображении.

Вторая основная функция, это функция forward — выполняет прямой сетевой проход изображения по нейронной сети.

```
30      def forward(self, aligned_face):
31          """
32
33          :param aligned_face:
34          :return:
35          """
36          return self.net.forward(aligned_face)
```

MyClassificator `MyClassificator` — Модуль классификации. В этом модуле будет происходить работа с библиотекой sklearn [9], сохранение и загрузка данных.

Сервис MyTelegramBot В качестве метода взаимодействия с пользователем будет использоваться telegram бот.

Бот для Telegram — это по своей сути серверное приложение, запущенное на стороне разработчика бота и осуществляющее запросы к Telegram Bot API [10]. У бота будет всего одна команда `"/help"`, которая будет возвращать сообщение с подсказкой. За это будет отвечать метод `help`:

Конфигурация приложения Так как приложение состоит из нескольких отдельных Docker контейнеров, целесообразно будет воспользоваться инструментом Docker Compose [11].

Использование Compose — это в основном трехэтапный процесс:

— определить среду приложения с помощью `Dockerfile`, чтобы он мог запускаться где угодно;

— определить службы, которые составляют ваше приложение в файле `docker-compose.yml`, чтобы они могли работать вместе в изолированной среде;

— запустить `docker-compose` и Compose соберёт и запустит все приложение.

2.3 Оценка классификатора

Эти массивы можно использовать для дальнейшей оценки классификатора.

Точность В простейшем случае такой метрикой может быть доля документов по которым классификатор принял правильное решение.

$$Accuracy = \frac{Positive}{Total},$$

где

- $Positive$ — количество документов по которым классификатор принял правильное решение;
- $Total$ — размер обучающей выборки.

Матрица неточностей Точность системы в пределах класса — это доля документов действительно принадлежащих данному классу относительно всех документов которые система отнесла к этому классу. Полнота системы — это доля найденных классификатором документов принадлежащих классу относительно всех документов этого класса в тестовой выборке.

Имея такую матрицу точность и полнота для каждого класса рассчитывается очень просто. Точность равняется отношению соответствующего диагонального элемента матрицы и суммы всей строки класса. Полнота — отношению диагонального элемента матрицы и суммы всего столбца класса. Формально:

$$Precision_c = \frac{A_{c,c}}{\sum_{i=1}^{N} A_{c,i}}$$

F-мера Понятно что чем выше точность и полнота, тем лучше. Но в реальной жизни максимальная точность и полнота не достижимы одновременно и приходится искать некий баланс. Поэтому, хотелось бы иметь некую метрику которая объединяла бы в себе информацию о точности и полноте нашего алгоритма. Именно такой метрикой является F-мера.

ЗАКЛЮЧЕНИЕ

В данной работе предложен подход для определения демографических признаков и характеристик по изображению лица человека. Экспериментально показано, что предложенный алгоритм обладает достаточной степенью точности. Причем предложенный метод обладает рядом преимуществ перед похожими методами идентификации, так как не восприимчив к малым углам поворота лица и не требует дорогостоящего оборудования для съемки. Данный метод является перспективным для дальнейших исследований, так как позволяет решать актуальную на сегодняшний день задачу классификации и идентификации изображений.

В ходе практики было реализовано веб-приложение а так же чатбот для определения демографических характеристик человека по фотографии, а именно:

— была получена выборка фотографий и характеристик из сервиса Google+;
— была реализованна и обучена система классификации изображений;
— был изучен и использован фреймворк для распознавания лиц OpenFace;
— были изучены основы микрофреймворка Flask для создания веб-приложений;
— были изучены основы концепции создания бота для мессенджера Telegram;
— был реализован и протестирован чатбот для мессенджера Telegram;
— была произведена оценка точности и полноты классификатора;
— был протестирован программный продукт.

Данное приложение может быть использованно как подсистема в обширном спектре сервисов: от социальных сетей и сайтов знакомств до систем наблюдения и сбора статистики.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 TIOBE Index | TIOBE - The Software Quality Company [Электронный ресурс]. — 2017. — URL: https://www.tiobe.com/tiobe-index/ (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

2 About Python™ | Python.org [Электронный ресурс]. — 2017. — URL: https://www.python.org/about/ (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

3 OpenCV [Электронный ресурс]. — 2016. — URL: http://www.opencv.org/about.html (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

4 Torch | Scientific computing for LuaJIT. [Электронный ресурс]. — 2016. — URL: http://torch.ch/ (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

5 CUDA Zone | NVIDIA Developer [Электронный ресурс]. — 2017. — URL: https://developer.nvidia.com/cuda-zone (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

6 *Amos, B.* Openface: A general-purpose face recognition library with mobile applications: Tech. rep. / B. Amos, B. Ludwiczuk, M. Satyanarayanan: CMU-CS-16-118, CMU School of Computer Science, 2016.

7 Flask Documentation (0.12) [Электронный ресурс]. — 2016.

8 *Grinberg, M.* Flask Web Development. Developing web applications with Python / M. Grinberg. — O'Reilly Media, 2014.

9 Documentation scikit-learn: machine learning in Python — scikit-learn 0.18.1 documentation [Электронный ресурс]. — 2016. — URL: http://scikit-learn.org/stable/documentation.html (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

10 Telegram Bot API [Электронный ресурс]. — 2016. — URL: https://core.telegram.org/bots/api (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

11 Overview of Docker Compose [Электронный ресурс]. — 2016. — URL: https://docs.docker.com/compose/overview/ (Дата обращения 31.05.2017). Загл. с экр. Яз. англ.

Бел
10.06.17