Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра дискретной математики и
информационных технологий

Разработка диагностической системы оценки состояния ТС методом машинного обучения

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 – Информатика и вычислительная техника
факультета КНиИТ
Евтеева Алексея Николаевича

Научный руководитель
к.ф.-м.н., доцент _____ И.Д. Сагаева

Заведующий кафедрой
к.ф.-м.н., доцент _____ Л.Б. Тяпаев

Саратов 2018 год

Введение. Вопросы применения систем виртуального общения на основе искусственного интеллекта исследуют на протяжении многих лет. На сегодняшний день проблема виртуального общения актуальна из-за быстрого доступа к информации, возможности одновременной работы в системе многих пользователей, обмена информацией, взаимодействия с целью решения любых вопросов, поддержки обучения, коммуникации с клиентами и партнерами по бизнесу, проведения аналитических исследований, сбора необходимой информации, повышения квалификации и других преимуществ.

Актуальной задачей является предоставление сервиса автолюбителям, который позволит на этапе первичного осмотра подержанного автомобиля отсеивать нежелательные к покупке варианты.

В настоящее время нет аналогов данной диагностической системе, работающей в виде чат-бота.

Целью данной работы является создание программной диагностической системы оценки состояния транспортного средства, которая будет помогать автолюбителю с вопросами диагностики основных неисправностей во время эксплуатации автомобиля и возможных причин их возникновения, а также осуществлять помощь при подборе подержанного автомобиля.

В ходе выпускной квалификационной работы описана разработка виртуального собеседника, работающего с нейронной сетью. Для этого были выделены и решены следующие задачи:

- изучение литературы по методам машинного обучения;
- разработка чат-бота для взаимодействия системы и пользователя;
- построение системы для диагностики и оценки состояния автомобиля на основе изученных алгоритмов;
- реализация взаимодействия чат-бота и построенной системы.

В главе 1 «Описание диагностической системы» содержится описание модели системы, ее составляющих. Также в ней описаны требования выдвигаемые к разработке системы.

В главе 2 «Подключения чат-бота» дается определение бота в «Telegram», методы получения обновлений. Также рассматривается метод отправки сообщений и список состояний чат-бота.

В главе 3 «Создание базы данных для хранения информации о пользователях и ответов на возможные запросы» описывается структура базы данных и способ ее создания на сервере Heroku.

В главе 4 «Введение в машинное обучение» даются основные определения, касающиеся машинного обучения и способов обучения.

В главе 5 «Искусственные нейронные сети» содержится описание нейронных сетей, основных элементов, функций активаций, методов обучения и рассматриваются вопросы глубокого обучения. Также рассматриваются основные архитектуры нейронных сетей, виды многослойных сетей.

В главе 6 «Библиотека TensorFLow» содержится описание библиотеки, графа вычислений и основных элементов графа.

В главе 7 «Оценка состояния ТС» дается постановка задачи. Рассматривается архитектура нейронной сети, а также процессы загрузки обучающей выборки, обучения сети и ее запуск.

В главе 8 «Формирование ответа» содержится описание соединения всех составляющих диагностической системы.

1 Описание диагностической системы.

Составляющие части системы:

- Telegram-аккаунт - это точка входа для пользователей, которые будут общаться с ботом.
- Telegram Bot API – это программный интерфейс, позволяющий программировать собственный бот.
- Python Script - скрипт чат-бота. В нем выполняются следующие функции:
  - обработка входящих сообщений;
  - переходы между состояниями;
  - обращение к базе данных;
  - отправка сообщения в качестве ответа.
- Deep learning - глубокое обучение.

Требования к системе:

- прием данных об идентификаторе пользователя;
- хранение полученных от пользователя данных и истории обращений к чат-боту в специализированной БД;
- база данных основных неисправностей и методов их устранения;
- реализация задачи классификации состояния автомобиля на основе данных предоставленных пользователем с помощью машинного обучения.

2 Подключение чат-бота

2.1 Боты в Telegram.

В Telegram боты — специальные аккаунты, созданные для того, чтобы автоматически обрабатывать и отправлять сообщения. Пользователи могут взаимодействовать с ботами при помощи сообщений, отправляемых через обычные или групповые чаты. Логика бота контролируется при помощи HTTPS запросов к нашему API для ботов.

API (application programming interface) - это набор готовых классов, функций, процедур, структур и констант. Вся эта информация предоставляется самим приложением (или операционной системой).

Все запросы к Telegram Bot API должны осуществляться через HTTPS в следующем виде: https://api.telegram.org/bot<token>/НАЗВАНИЕ_МЕТОДА. Допускаются GET и POST запросы [1].

2.2 Получение обновлений.

Существует два диаметрально противоположных по логике способа получать обновления от вашего бота: getUpdates и вебхуки. Входящие обновления будут храниться на сервере до тех пор, пока пользователь их не обработает, но не дольше 24 часов.

Независимо от способа получения обновлений, в ответ пользователь получит объект Update, сериализованный в JSON.

2.3 Отправка сообщений. Для отправки сообщений пользователю используется метод sendMessage. Данный метод формирует следующий запрос: https://api.telegram.org/bot<token>/sendMessage?chat_id=<ID чата>\&text=<text>, в котором необходимо указать:

- <token> - Токен бота;
- <ID чата> - ID чата;
- <text> - текст сообщения [2].

2.4 Переход между состояниями. Для того, чтобы чат-бот мог общаться с пользователями и поддерживать диалог, необходимо реализовать переходы между состояниями.

Список состояний чат-бота:

- start - начальное состояние присваивается, когда пользователь впервые обращается к боту или при вызове команды start;
- rpl1 - Выбор действия пользователя. Переход в состояние rpl2, eval или rpl1;
- rpl2 - Ответ на пользовательский запрос о диагностике автомобиля. После ответа пользователю переход в состояние rpl1;
- eval - Начало опроса пользователя для оценки состояния автомобиля. Вопрос о типе кузова. Переход в состояние eval_rpl1;
- eval_rpl1 - Вопрос о пробеге автомобиля. Переход в состояние eval_rpl2;
- eval_rpl2 - Вопрос о возрасте автомобиля. Переход в состояние eval_rpl3;
- eval_rpl3 - Вопрос о количестве владельцев. Переход в состояние eval_rpl4;
- eval_rpl4 - Вопрос о дефектах кузова. Переход в состояние eval_rpl5;
- eval_rpl5 - Вопрос о шуме в двигателе. Переход в состояние eval_rpl6;
- eval_rpl6 - Вопрос о подтеках масла. Переход в состояние eval_rpl7;
- eval_rpl7 - Вопрос о цвете выхлопных газов. Переход в состояние eval_rpl8;
- eval_rpl8 - Сформированный вектор признаков проходит через нейронную сеть, определяется состояние автомобиля и формируется ответ. Переход в состояние rpl1;

- help - Список доступных команд. Переход в состояние rpl1.

3 Создание базы данных для хранения информации о пользователях и ответов на возможные запросы. В созданной базе данных используются следующие таблицы:

- Chats - таблица, в которой хранится информация о пользователях с которыми общался бот;
- History - таблица, в которой хранится запросов пользователя;
- Answers - таблица, в которой хранится перечень ответов на запросы пользователей.

Для создания базы данных на сервере Heroku в командной строке необходимо ввести команду heroku addons:add heroku-postgresql:dev. В результате будет сообщение об успешном создании базы данных. Параметры для подключения к базе данных можно узнать с помощью команды heroku pg:credentials:url [3].

4 Введение в машинное обучение. Машинное обучение — обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться. Различают два типа обучения. Обучение по прецедентам, или индуктивное обучение, основано на выявлении общих закономерностей по частным эмпирическим данным. Дедуктивное обучение предполагает формализацию знаний экспертов и их перенос в компьютер в виде базы знаний.

Способы обучения:

- Обучение с учителем – для каждого прецедента существует пара «ситуация, решение»;
- Обучение без учителя – система группирует объекты в кластеры и понижает размерность входной информации, используя данные о попарном сходстве;
- Обучение с подкреплением – для каждого прецедента существует пара «ситуация, реакция среды» [4].

5 Искусственные нейронные сети. Искусственные нейронные сети (ИНС) — математические модели, а также их программные или аппаратные реализации, построенные по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма.

Составным элементом нейронных сетей являются нейроны. Каждый нейрон обладает группой синапсов - однонаправленных входных связей, соединенных с выходами других нейронов. Каждый синапс характеризуется величиной синоптической связи или ее весом w_i. Нейрон имеет текущее состояние, которое обычно определяется, как взвешенная сумма его входов:

$$s = \sum_{i=1}^{n} x_i w_i$$

Нейрон имеет аксон - выходную связь данного нейрона, с которой сигнал (возбуждения или торможения) поступает на синапсы следующих нейронов.

Выход нейрона есть функция его состояния:

$$y = f(s)$$

Функция f называется функцией активации и может иметь разный вид:

- пороговый;
- кусочно-линейный;
- сигмоид.

Множество всех нейронов искусственной нейронной сети можно разделить на подмножества - слои. Взаимодействие нейронов происходит послойно[5].

5.1 Архитектура ИНС. Существуют различные классификации ИНС по ряду признаков. По одному из топологических признаков ИНС можно классифицировать как:

- Полносвязные ИНС – каждый нейрон связан с остальными нейронами в сети, включая себя самого;
- Слоистые ИНС – нейроны объединяются в слои, нейроны предыдущего слоя связаны с нейронами следующего слоя;
- Слабосвязные ИНС – нейроны расположены в узлах прямоугольной или гексогональной решётки.

5.2 Виды многослойных ИНС. Статические ИНС прямого распространения:

- Перцептрон;
- Нейронная сеть Кохонена;
- Когнитрон;

- Неокогнитрон;
- Современная свёрточная нейронная сеть.

Динамические рекуррентные ИНС с обратными связями:

- Нейронная сеть Хопфилда;
- Нейронная сеть Коско;
- Нейронная сеть Джордана;
- Нейронная сеть Элмана.

5.3 Обучение нейронных сетей. Процесс обучения ИНС рассматривается как настройка архитектуры и весов связей между нейронами(параметров) для эффективного выполнения поставленных перед ИНС задач. Существует два обширных класса обучения ИНС: класс детерминированных методов и класс стохастических методов.

5.3.1 Обучение с учителем. Во время обучения ИНС с учителем каждому примеру из обучающей выборки соответствует вектор, характеризующий однозначный правильный ответ, подаваемый сразу на выход сети в обход всей её архитектуры. После получения собственного результата сети, алгоритм сравнивает результирующий вектор с правильным ответом, на основе чего происходит коррекция дальнейшая ошибки [6].

5.3.2 Обучение без учителя. Обучение без учителя в случае ИНС реализуется естественным образом в процессе обучения, когда автоматическая настройка параметров сетью приводит к появлению одинаковых результатов её функционирования при достаточно близких входных значениях, что на практике можно сравнить с понижением размерности данных в результате итерационного метода главных компонент [7].

5.3.3 Метод обратного распространения ошибки. Метод является классическим методом обучения с учителем, основан на правиле коррекции по ошибке и был разработан как метод обучения многослойного перцептрона. Основная идея метода состоит в распространении сигналов ошибки после её вычисления на выходе сети в направлении, обратном прямому распространении сигналов во время обычного вычислительного процесса, от выходов сети к её входам. При обратном проходе синаптические веса настраиваются с целью минимизации ошибки [8].

5.4 Глубокое обучение. Глубокое обучение — совокупность методов машинного обучения (с учителем, с частичным привлечением учителя, без учителя, с подкреплением), основанных на обучении представлениям, а не на специализированных алгоритмах, разработанных для конкретных задач.

6. Библиотека TensorFlow. TensorFlow - это библиотека программного обеспечения с открытым исходным кодом для задач машинного обучения, разработанная Google. Она позволяет создавать и обучать нейронные сети различной архитектуры для обнаружения и распознавания образцов и поиска взаимосвязей. Граф вычислений является моделью, описывающей как будут выполняться вычисления. Граф состоит из плэйсхолдеров(tf.Placeholder), переменных(tf.Variable) и операций [9].

7 Оценка состояния ТС.

7.1 Постановка задачи. Имеется множество объектов, разделенных на два класса. Каждый объект задается в виде числового вектора (вектора признаков) фиксированной длины. Требуется построить алгоритм, способный классифицировать произвольный объект, сформированный во время общения чат-бота и пользователя.

На стадии оценки состояния ТС существует два класса:

- класс 0 - Автомобиль в хорошем состоянии;
- класс 1 - Автомобиль в плохом состоянии.

7.2 Архитектура нейронной сети. В работе используется нейронная сеть с тремя скрытыми слоями. Задача каждого скрытого слоя заключается в том, чтобы превратить данные во что-то, что мог бы использовать слой вывода для определения результата.

Для получения результатов выходного слоя используется унитарное кодирование.

Унитарный код - это двоичный код фиксированной длины, содержащий только одну единицу или только один нуль. Для кодирования двух классов необходим код длины 2.

7.3 Обучение нейронной сети. Веса и смещения хранятся в переменных tf.Variable, которые содержат состояние в графе между вызовами run().Чтобы tensorflow мог обучать модель также необходимы функция потерь и алгоритм оптимизации.

Существует много методов, как посчитать потерю. Для задач на классификацию, лучшим способом вычисления ошибки является метод перекрестной энтропии.

Для обучения сети используется алгоритм оптимизации Adadelta. Данный метод динамично адаптируется с течением времени, используя только первый заказ информации и имеет минимальные вычислительные расходы за пределами стохастического градиентного спуска.

7.4 Входные данные. Набором данных, который используется надо управлять, чтобы передать нейронной сети. Для реализации необходимо преобразовать входную строку в вектор. Это выполняется при помощи функции text_to_vector().

То же самое необходимо сделать с классами, но с использованием унитарного кодирования. Для этого используется функция category_to_vector.

7.5 Запуск графа и получение результата. Для запуска графа необходимо загрузить набор данных, содержащий тысячу векторов с признаками.

Далее необходимо обучение системы на основе полученных данных. В терминологии нейронных сетей одна эпоха - один передний проход (получение значений вывода) и один обратный проход (обновление весов) всех тренировочных примеров.

Для запуска модели используется метод tf.Session.run(fetches, feed_dict=None, options=None, run_metadata=None).

8 Формирование ответа. Полный скрипт для чат-бота представлен в приложении Г. Заключительной частью данной выпускной квалификационной работы является соединение всех составляющих, рассмотренных в предыдущих главах.

Одна из функций чат-бота - ответ пользователю на вопрос о наиболее характерных неисправностей ТС в состоянии rpl2.

При оценке состояния ТС чат-бот проходит от состояния eval до состояния eval_rpl8, при этом на основе ответов пользователя формируется строка с признаками. Для взаимодействия чат-бота с нейросетью используется функция eval_car.

Заключение. В рамках данной ВКР дается описание работы диагностической системы оценки состояния ТС, ориентированной на автолюбителей. Описывается интерфейс Telegram Bot API, его основные параметры и возможности. Реализовано обращение к чат-боту, с помощью Telegram Bot API. Выстроена структура диалога между пользователем и ботом.

При помощи технологии TensorFlow, была разработана многослойная нейронная сеть, рассмотрена ее архитектура и обучение. Показан запуск модели, обучение и результат ее тестирования.

Результатом данной работы является диагностическая система оценки состояния ТС, работающая в виде чат-бота в мессенджере «Telegram». Чат-бот имеет выстроенную структуру диалога и интеграцию с многослойной нейронной сетью.

Поставленная задача была решена достаточно полно. Программу можно дорабатывать, дообучать нейросеть для более точного предсказания результата. Обращаться к чат-боту может любой пользователь, для этого достаточно воспользоваться приложением «Telegram».

Список использованных источников

1. Телеграм-бот как простой и удобный способ получения информации [Электронный ресурс]. URL: https://cyberleninka.ru/article/v/telegram-bot-kak-prostoy-i-udobnyy-sposob-polucheniya-informatsii. (Дата обращения: 1.06.2018).
2. Документация Telegram: API [Электронный ресурс]. URL: https://tlgrm.ru/docs/bots/api (Дата обращения: 2.06.2018)
3. Heroku Postgres | Heroku Dev Center [Электронный ресурс]. URL: https://devcenter.heroku.com/categories/heroku-postgres (Дата обращения: 2.06.2018)
4. Машинное обучение [Электронный ресурс]. URL: http://www.machinelearning.ru/wiki/index.php?title=Machine_Learning (Дата обращения: 2.06.2018)
5. Что такое искусственные нейронные сети? / Хабр [Электронный ресурс]. URL: https://habr.com/post/134998/ (Дата обращения: 2.06.2018)
6. НОУ ИНТУИТ | Лекция | Основы искусственных нейронных сетей [Электронный ресурс]. URL: https://www.intuit.ru/studies/courses/88/88/lecture/20527?page=4 (Дата обращения: 2.06.2018)
7. Нейронные сети: обучение без учителя [Электронный ресурс]. URL: http://www.codenet.ru/progr/alg/ai/htm/gl3_4.php (Дата обращения: 2.06.2018)
8. Обучение нейронной сети. Алгоритм обратного распространения ошибок. | MicroTechnics [Электронный ресурс]. URL: http://microtechnics.ru/obuchenie-nejronnoj-seti-algoritm-obratnogo-rasprostraneniya-oshibok/ (Дата обращения: 2.06.2018)
9. Variables | TensorFlow [Электронный ресурс]. URL: https://www.tensorflow.org/programmers_guide/variables (Дата обращения: 2.06.2018)