

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

Кафедра математической кибернетики и компьютерных наук

**РЕШЕНИЕ ЗАДАЧИ КЛАССИФИКАЦИИ С ПОМОЩЬЮ
НЕЙРОННЫХ СЕТЕЙ**

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные
технологии
факультета КНиИТ
Леушкиной Дарьи Александровны

Научный руководитель
к. ф.-т. п. доцент

Г. Г. Наркайтис

Заведующий кафедрой
к. ф.-м. н.

С. В. Миронов

Саратов 2019

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 Нейронные сети	5
1.1 Основные понятия	5
2 Решение задач классификации с помощью нейронных сетей	7
2.1 Основные понятия	7
2.2 TensorFlow	7
2.3 Keras	8
2.4 Применение глубокого MLP к MNIST	8
2.5 Распознавание речи с помощью нейронных сетей	9
ЗАКЛЮЧЕНИЕ	11
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	13

ВВЕДЕНИЕ

Технология машинного обучения на основе анализа данных берёт начало в 1950 году, когда начали разрабатывать первые программы для игры в шашки. За прошедшие десятилетия общий принцип не изменился, но благодаря росту вычислительных мощностей компьютеров усложнились закономерности и прогнозы, создаваемые ими, расширился круг проблем и задач, которые решаются с использованием машинного обучения.

Вопросы человеко-машинного взаимодействия являются одними из важнейших при создании новых компьютеров. Наиболее эффективными средствами взаимодействия человека с машиной были бы те, которые являются естественными для него: через визуальные образы и речь. Создание речевых интерфейсов могло бы найти применение в системах самого различного назначения: голосовое управление для людей с ограниченными возможностями, надежное управление боевыми машинами, «понимающими» только голос командира, автоответчики, обрабатывающие в автоматическом режиме сотни тысяч звонков в сутки (например, в системе продажи авиабилетов) и т.д. При этом речевой интерфейс должен включать в себя два компонента: систему автоматического распознавания речи для приема речевого сигнала и преобразования его в текст или команду, и систему синтеза речи, выполняющего противоположную функцию — конвертацию сообщения от машины в речь.

Однако, не смотря на стремительно возрастающие вычислительные мощности, создание систем распознавания речи остается чрезвычайно сложной проблемой. Это обуславливается как ее междисциплинарным характером (необходимо обладать знаниями в филологии, лингвистике, цифровой обработке сигналов, акустике, статистике, распознавании образов и т.д.), так и высокой вычислительной сложностью разработанных алгоритмов. Последнее накладывает существенные ограничения на системы автоматического распознавания речи — на объем обрабатываемого словаря, скорость получения ответа и его точность. Нельзя также не упомянуть о том, что возможности дальнейшего увеличения быстродействия ЭВМ за счет совершенствования интегральной технологии рано или поздно будут исчерпаны, а все возрастающая разница между быстродействиями памяти и процессора только усугубляет проблему.

Актуальность выбранной темы дипломной работы неоспоримо важна. Существуют области применения систем автоматического распознавания ре-

чи, где описанные проблемы проявляются особенно остро из-за жестко ограниченных вычислительных ресурсов, например, на мобильных устройствах. Производители мобильных телефонов и планшетов нашли выход в переносе ресурсоемких вычислений с устройств пользователей на серверы в облаке, где фактически и производится распознавание. Пользовательское приложение только отправляет туда речевые запросы и принимает ответы, используя подключение к интернету. По этой схеме успешно работают системы Siri от Apple и Google Voice Search от Google. Однако, для такой реализации необходимы определенные условия, например, непрерывный доступ к интернету, которые в ряде случаев недостижимы, и требуется создать компактное и надежное самостоятельное устройство, эксплуатирующее только доступные «на месте» вычислительные мощности.

Предмет исследования. В данной работе будут введены основные понятия нейронных сетей, машинного обучения и задач, которые можно решить с их помощью. В практической части будут реализованы две задачи: задача распознавания рукописных цифр и задача распознавания речи.

Цель бакалаврской работы — обучение модели для распознавания цифр при помощи библиотеки глубокого машинного обучения Keras на языке Python. Демонстрация важности контроля обучаемой нейронной сети. Для достижения данной цели были определены следующие задачи:

- подготовка набора данных для обучения модели;
- обучение нейронной сети на тренировочных данных;
- демонстрация работы обученной модели для тестовом наборе данных;
- демонстрация погрешности предсказаний на недоученных и переобученных моделях.

Структура и объем работы. Бакалаврская работа состоит из введения, двух разделов, заключения, списка использованных источников и двух приложений. Общий объем работы — 58 страниц, из них 48 страницы — основное содержание, включая 20 рисунков, цифровой носитель в качестве приложения, список использованных источников информации — 23 наименования.

1 Нейронные сети

1.1 Основные понятия

Нейронная сеть (искусственная нейронная сеть) — это система соединенных и взаимодействующих между собой простых процессоров (искусственных нейронов). Каждый процессор подобной сети имеет дело только с сигналами, которые он периодически получает, и сигналами, которые он периодически посылает другим процессорам. И, тем не менее, будучи соединёнными в достаточно большую сеть с управляемым взаимодействием, эти процессоры вместе способны выполнять довольно сложные задачи, поскольку нейронные сети обучаются в процессе работы [1].

Нейрон — вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше. Они делятся на три основных типа: входные, скрытые и выходные .

Функция активации — способ нормализации входных данных. То есть если на входе будет большое число, то, пропустив его через функцию активации, можно получить выход в нужном диапазоне. Функций активации достаточно много, главным их отличием друг от друга является диапазон значений. Основные виды функций активации:

Синапс — связь между двумя нейронами. У синапсов есть один параметр — вес, благодаря которому входная информация изменяется, когда передается от одного нейрона к другому. Пусть есть 3 нейрона, которые передают информацию следующему нейрону. Информация того нейрона, у которого вес будет больше, и будет доминирующей в следующем нейроне. Например, смешение цветов.

Существуют также другие виды нейронов, а именно нейрон смещения и контекстный нейрон [2]. Особенность нейронов смещения заключается в том, что их входы и выходы всегда равняются единице, и они никогда не имеют входных синапсов. Нейроны смещения могут либо присутствовать в нейронной сети по одному на слое, либо полностью отсутствовать. Соединения у нейронов смещения такие же, как у обычных нейронов, то есть со всеми нейронами следующего уровня. Таким образом, нейроны смещения можно размещать на входном слое и всех скрытых слоях, на выходное слое их размещать нельзя, так как им не с чем будет формировать связь [3].

Нейрон смещения позволяет осуществить сдвиг по оси координат та-

ким образом, чтобы нейросеть смогла захватить область, недоступную ей без сдвига. При этом сдвиг может быть осуществлён как вправо, так и влево. Схематически нейроны сдвига обычно не обозначаются, их вес учитывается по умолчанию при расчёте входного значения [7].

Градиентный спуск — способ нахождения локального минимума или максимума функции с помощью движения вдоль градиента. Градиент — это вектор, который определяет крутизну склона и указывает его направление относительно какой-либо из точек на поверхности или графике. Для того чтобы найти градиент нужно взять производную от графика по данной точке. Двигаясь по направлению этого градиента, мы будем плавно скатываться в «низину». Построим график, где по оси x будут значения веса нейрона (w), а по оси y — ошибка, соответствующая этому весу (e).

2 Решение задач классификации с помощью нейронных сетей

2.1 Основные понятия

Фреймворк — заготовки, шаблоны для программной платформы, определяющие архитектуру программной системы; программное обеспечение, которое облегчает разработку и объединение разных модулей программного проекта [5].

Тензор — объект линейной алгебры, линейно преобразующий элементы одного линейного пространства в элементы другого.

2.2 TensorFlow

TensorFlow — открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронных сетей с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия. Название происходит от операций с многомерными массивами данных, которые также называются тензорами. TensorFlow является относительно молодым фреймворком для глубокого машинного обучения.

Работа с TensorFlow строится на основе построения и выполнения графа вычислений. Граф вычислений — конструкция, описывающая то, каким образом будут проводиться вычисления. Основой TensorFlow является создание структуры, которая будет задавать порядок вычислений.

Граф вычислений состоит из плеисхолдеров (объекты, которые параметризуют граф и отмечают места для подстановки внешних значений), переменных и операций. Из этих элементов собирается граф, в котором будут вычисляться тензоры (многомерные массивы данных).

Описание тензора показывает несколько важных аспектов:

1. У тензоров есть имена.
2. Существует понятия формы тензора (похоже на размерность массива).
3. Тензоры типизированы, типа для них задаются из библиотеки.

После получения модели может потребоваться её сохранение. В TensorFlow существует специальный объект-сериализатор, который сохраняет текущий граф, его состояние и значения переменных в файл и читает всё то же самое из файла [8].

2.3 Keras

Keras — открытая нейросетевая библиотека, написанная на языке программирования Python. Она представляет собой надстройку над фреймворками DeepLearning4j, TensorFlow и Theano. Нацелена на оперативную работу с сетями глубинного обучения, при этом спроектирована так, чтобы быть компактной, модульной и расширяемой. Эта библиотека содержит многочисленные реализации широко применяемых строительных блоков нейронных сетей, таких как слои, целевые и передаточные функции, оптимизаторы и множество инструментов для упрощения работы [6].

2.4 Применение глубокого MLP к MNIST

База данных MNIST — объемная база данных образцов рукописного написания цифр. Данные состоят из заранее подготовленных примеров изображений, на основе которых проводится обучение и тестирование систем [11].

Реализуем простейшую глубокую нейронную сеть — MLP с двумя слоями — и применим ее к одной из самых распространенных задач классификации — задаче распознавания рукописных цифр из набора данных MNIST.

Для начала определим некоторые параметры нашей модели — гиперпараметры. Гиперпараметры — значения, которые нужно подбирать вручную и зачастую методом проб и ошибок. В данной модели зададим следующие параметры:

1. `batchSize=200` — количество обучающих образцов, обрабатываемых одновременно за одну итерацию градиентного спуска;
2. `epochs=10` — количество итераций обучающего алгоритма по всему обучающему множеству.

Далее загружаем базу данных MNIST с помощью Keras — данные просто считываются с удаленного сервера напрямую в массивы библиотеки `numpy`. Код, отвечающий за загрузку MNIST:

```
1 (imgstrain, numstrain), (imgstest, numstest) = mnist.loaddata()
```

Затем нужно провести предварительную обработку данных. Представим изображения в виде одномерных массивов (так как считаем каждый пиксель отдельным входным признаком), а затем разделим значение интенсивности каждого пикселя на 255, чтобы новое значение попадало в отрезок $[0, 1]$. Код предварительной обработки данных:

```

1 pixel_count = imgs_train.shape[1] * imgs_train.shape[2]
2 imgs_train = imgs_train.reshape(imgs_train.shape[0], pixel_count).astype('float32')
3 imgs_test = imgs_test.reshape(imgs_test.shape[0], pixel_count).astype('float32')
4 imgs_train = imgs_train / 255
5 imgs_test = imgs_test / 255

```

Хорошим подходом к задаче классификации является вероятностная классификация, при которой имеется один выходной нейрон для каждого класса, выдающий вероятность того, что входной элемент принадлежит данному классу. Это подразумевает необходимость преобразования обучающих выходных данных в прямое кодирование.

Для определения модели воспользуемся стеком из трёх Dense-слоев, который соответствует полносвязному MLP, где все выходы одного слоя связаны со всеми входами последующего. Будем использовать ReLU для нейронов первых двух слоев, а для последнего слоя — softmax.

Осталось определить функцию потерь, алгоритм оптимизации и метрики, которые будем собирать. Когда используется вероятностная классификация, в качестве функции потерь лучше использовать перекрестную энтропию. В качестве оптимизатора будем использовать оптимизатор Адама, который показывает хорошую производительность, а в качестве метрики — точность — доля входных данных, отнесенных к правильному классу [10]. Фрагмент кода, отвечающий за последний этап подготовки модели к обучению.

Затем следует обучение модели, которое выполняется с помощью функции `fit`. Она принимает на вход обучающую выборку вместе с метками `xTrain` и `yTrain`, размером `batchSize`, который ограничивает количество примеров, подаваемых за раз, и количеством эпох для обучения `epochs`. Точность составила 99%.

2.5 Распознавание речи с помощью нейронных сетей

У нас есть набор файлов с аудиоданными в формате WAV. Формат подразумевает наличие в файле двух блоков. Первый блок — заголовок с информацией об аудиопотоке: битрейте, частоте, количестве каналов, длине файла и т.д. Второй блок состоит из данных, представляющих собой цифровой сигнал, то есть набора значений амплитуд.

Сначала считываем заголовок, проверяем некоторые ограничения (к примеру, отсутствие сжатия), сохраняем данные в специально выделенный массив.

Подход должен быть устойчив к изменению тембра голоса человека, произносящего слово, громкости и скорости произношения. Поэлементным сравнением двух аудиосигналов этого добиться нельзя, поэтому разбиваем данные на небольшие временные промежутки — фреймы. При этом фреймы должны идти не строго друг за другом, а «внахлёст», то есть конец одного фрейма должен пересекаться с началом другого, потому что это позволяет сгладить результаты анализа фреймов, превращая идею фреймов в некоторое «окно», движущееся вдоль исходной функции (значений сигнала). Для дальнейшего анализа будем использовать мел-частотные кепстральные коэффициенты [4].

После того, как модель сформирована, нужно подготовить её к работе. С помощью функции `librosa.feature.mfcc` подсчитываются мел-кепстральные частотные коэффициенты с частотой дискретизации 8 кГц.

После этого используется функция `compile`, параметрами которой являются:

1. `loss` — функция ошибки;
2. `optimizer` — используемый оптимизатор;
3. `metrics` — метрики, по которым считается качество модели.

Далее следует обучение модели, которое выполняется с помощью функции `fit`. Она принимает на вход обучающую выборку вместе с метками `x_train` и `y_train`, размером `batch_size`, который ограничивает количество примеров, подаваемых за раз, количеством эпох для обучения `epochs`, параметром `verbose`, который принимает значение 0 или 1 и отвечает за то, будут ли подробно расписаны эпохи обучения или нет, и параметром `validation_split`, который показывает, какую долю обучающей выборки отдать под валидацию (проверку данных) [9].

Далее необходимо сохранить обученную модель в файл. Полностью реализация распознавания речи представлена в Приложении Б.

Точность составила 94%.

ЗАКЛЮЧЕНИЕ

Потенциальными областями применения искусственных нейронных сетей являются те, где человеческий интеллект малоэффективен, а традиционные вычисления трудоёмки или физически неадекватны (т.е. не отражают или плохо отражают реальные физические процессы и объекты). Актуальность применения нейронных сетей многократно возрастает, когда появляется необходимость решения плохо формализованных задач. Основные области применения нейронных сетей:

- автоматизация процесса классификации;
- автоматизация прогнозирования;
- автоматизация процесса распознавания;
- автоматизация процесса принятия решений;
- управление, кодирование и декодирование информации;
- аппроксимация зависимостей и др.

С помощью нейронных сетей успешно решается важная задача в области телекоммуникаций — проектирование и оптимизация сетей связи (нахождение оптимального пути трафика между узлами). Кроме управления маршрутизацией потоков, нейронные сети используются для получения эффективных решений в области проектирования новых телекоммуникационных сетей. Распознавание речи — одна из наиболее популярных областей применения нейронных сетей. В ходе написания данной работы были введены основные понятия нейронных сетей, машинного обучения, также были описаны области применения нейронных сетей и возможности их применения для решения задач различной направленности: классификации, распознавания и предсказания. В данной работе был успешно реализован простой двухслойный глубокий MLP с помощью открытой нейросетевой библиотеки Keras, написанной на языке программирования Python. Реализованный простой двухслойный глубокий MLP был применён к набору данных MNIST для решения задачи классификации рукописных цифр. Точность распознавания при заданных условиях достигает 99%. Также в практической части с помощью нейронных сетей была реализована задача распознавания речи, а именно была решена задача классификации десяти цифр, произносимых на английском языке. Была описана модель решения данной задачи. Для получения вектора свойств сигнала использовался метод мел-частотных кепстральных коэффициентов (MFCC). Точность распо-

знавания при заданных условиях достигает 94%.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 Беркинблит М. Б. Нейронные сети. — М.: МИРОС и ВЗМШ РАО, 1993. — 96 с. (дата обращения: 15.12.2018).
- 2 Виды нейронных сетей [Электронный ресурс] / Блог-платформа «Наука об искусственном интеллекте» [Электронный ресурс]. URL: <https://ai-science.ru/vidy-nejronnyx-setej/> (дата обращения: 03.02.2019). Загл. с экрана. Яз. рус.
- 3 Нейронные сети для начинающих. Часть 2 [Электронный ресурс] / Блог-платформа «habr» [Электронный ресурс]. URL: <https://habr.com/ru/post/313216/> (дата обращения: 05.02.2019). Загл. с экрана. Яз. рус.
- 4 Цирезан Д. (июнь 2012 г.). «Глубокие нейронные сети для классификации изображений». Конференция IEEE 2012 года по компьютерному зрению и распознаванию образов: 3642—3649. DOI: 10,1109/cvpr.2012.6248110. (дата обращения: 11.03.2019).
- 5 Ахтырченко К. В., Сорокваша Т. П. Методы и технологии реинжиниринга ИС. Институт системного программирования РАН (дата обращения: 14.04.2019).
- 6 Джулли А., Пал С. Библиотека Keras — инструмент глубокого обучения = Deep learning with Keras. — ДМК Пресс, 2017. — 294 с. — ISBN 978-5-97060-573-8. (дата обращения: 23.04.2019).
- 7 Нейронные сети: виды, принцип работы и области применения [Электронный ресурс] / Интернет-журнал livesurf [Электронный ресурс]. URL: <https://livesurf.ru/zhurnal/6067-nejronnye-seti-vidy-princip-raboty-i-oblasti-primeneniya.html> (дата обращения: 27.03.2019). Загл. с экрана. Яз. рус.
- 8 Tensorflow [Электронный ресурс] / Официальный сайт «Tensorflow» [Электронный ресурс]. URL: <https://www.tensorflow.org/> (дата обращения: 28.04.2019). Загл. с экрана. Яз. Англ.
- 9 Шолле Ф. Глубокое обучение на Python — Питер, 2018. — 400 с. — ISBN 978-5-4461-0770-4. (дата обращения: 28.04.2019).

- 10 Рашка С. Python и машинное обучение. — ДМК Пресс, 2017. — 418 с. — ISBN 978-5-97060-409-0. (дата обращения: 16.12.2018).
- 11 Лекун Я., Берджес К., Кортес К. База данных рукописных цифр MNIST(дата обращения: 15.12.2018).