

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Ярославский государственный университет им. П.Г. Демидова»

Кафедра дискретного анализа

«Допустить к защите»
Заведующий кафедрой
д.ф.-м.н., профессор
_____ Бондаренко В.А.
«__» _____ 2020 г.

Выпускная квалификационная работа бакалавра
по направлению 01.03.02 Прикладная математика и информатика

**Анализ алгоритмов глубокого машинного обучения в задачах
распознавания изображений**

Научный руководитель
к.т.н., старший преподаватель
_____ Матвеев Д.В.
«__» _____ 2020 г.

Студент группы ИВТ-41БО
_____ Коротков А.С.
«__» _____ 2020 г.

Ярославль, 2020

РЕФЕРАТ

Выпускная квалификационная работа: 21 стр., 6 гл., 2 рис., 1 таблиц, 9 источников.

Ключевые слова: машинное обучение, глубокие нейронные сети, распознавание изображений, TensorFlow, OpenCV, Keras.

Объектом исследования являются методы на основе глубоких нейронных сетей для задач распознавания изображений.

Цель работы – изучить применение глубокого машинного обучения в задачах распознавания изображений.

В результате работы была разработана и реализована нейронная сеть для решения ????. Проведен анализ полученных результатов и сделан вывод о качестве работы нейронных сетей в задачах распознавания изображений.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	4
1 ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ	5
1.1 Задача компьютерного зрения	5
1.2 Искусственные нейронные сети	5
1.2.1 Понятие искусственной нейронной сети	5
1.2.2 Активационная функция	6
1.2.3 Структура нейронной сети	7
1.2.4 Глубокое машинное обучение	8
1.2.5 Сверточные нейронные сети	8
1.2.6 Проблемы обучения нейронных сетей	8
1.3 Применение нейронных сетей в задачах распознавания изображений	8
1.4 Инструменты машинного обучения	9
2 ПОСТАНОВКА ЗАДАЧИ	10
3 ПРОЕКТИРОВАНИЕ СИСТЕМЫ ДЛЯ ???	11
4 ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ	12
5 ПРИМЕРЫ РАБОТЫ ПРОГРАММЫ	13
6 АНАЛИЗ РАБОТЫ ПРОГРАММЫ	14
ЗАКЛЮЧЕНИЕ	15
СПИСОК ЛИТЕРАТУРЫ	16
СПИСОК ИЛЛЮСТРАТИВНОГО МАТЕРИАЛА	17
СПИСОК ТАБЛИЧНОГО МАТЕРИАЛА	18
ПРИЛОЖЕНИЕ А	19

ВВЕДЕНИЕ

В настоящее время, в связи со стремительным развитием цифровых технологий, использование автоматизированных и роботизированных систем распространилось на множество областей как в промышленности, науке, так и в повседневной жизни. В следствие этого, возрастает необходимость в эффективной обработке информации, представленной, в частности, в формате видео и изображений.

На текущий момент изображения тесно влились в жизнь человека. Поэтому многие автоматизированные системы используют их в качестве основного источника информации. Нахождение, локализация, классификация и анализ объектов на изображении компьютером – сложная задача компьютерного зрения.

В процессе обработки информации, получаемой из глаз, человеческий мозг پردازывает колоссальный объем работы. Человек без труда сможет описать что находится и что происходит на случайно взятой фотографии. Изображения могут нести в себе колоссальное количество деталей и отличаться множеством параметров, таких как: разрешение, цветность, качество, яркость, наличие шума и т.д. Объекты на изображениях также могут обладать множеством особенностей: масштаб, положение, цвет, поворот, наклон и т.д. Однако, в цифровом формате, каждое изображение представляет собой лишь массив числовых данных. Научить компьютер находить и классифицировать образы на изображении с учетом всех факторов – очень сложная алгоритмическая задача. Для её решения активно применяют технологии машинного обучения.

В данной работе был проведен анализ алгоритмов глубокого машинного обучения для решения задач распознавания изображений, а также разработана система для ???.

В первой главе проведен обзор основных тем и задач ???

Во второй главе ???

В третьей главе ???

1 ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Задача компьютерного зрения

Компьютерное (машинное) зрение – это совокупность программно-технических решений в сфере искусственного интеллекта (ИИ), нацеленных на считывание и получение информации из изображений, в реальном времени и без участия человека.

Большое количество информации человек получает при помощи зрения. Изображения способны хранить огромное количество информации. Как следствие их использование в компьютерных системах способствует увеличению их производительности. Однако такие технологии требуют сложных вычислений. Задачей компьютерного зрения является разработка эффективных алгоритмов, выполняющих извлечение и анализ данных из изображений или видео.

В настоящий момент, такие технологии применяются для решения таких сложных задач как:

- OCR – Optical character recognition (Оптическое распознавание символов): преобразование текста на изображении в редактируемый.
- Фотограмметрия – технология создания трехмерной модели объекта на основе фотографий, сделанных с различных ракурсов.
- Motion capture – технология, широко применяемая в киноиндустрии, позволяющая преобразовывать движения реальных людей в компьютерную анимацию.
- Дополненная реальность (AR) – технология, позволяющая в реальном времени проецировать виртуальные объекты на изображение реального окружения.
- Медицинская диагностика – обнаружение раковых клеток на ранней стадии, увеличение качества МРТ изображений, их анализ и т.д.

1.2 Искусственные нейронные сети

1.2.1 Понятие искусственной нейронной сети

Машинное обучение – раздел исследований в сфере ИИ, в основе которых лежат методы разработки систем способных к обучению. ???

Искусственная нейронная сеть (ИНС) – компьютерная модель, в основе которой лежат принципы работы биологической нейронной сети - совокупности связанных между собой нервных клеток - нейронов. Каждый нейрон имеет набор входных связей - синапсов, по которым он получает информацию, представленную в виде импульсов, от других нейронов. По полученным данным нейрон формирует своё состояние и, с помощью аксона, сообщает его другим нейронам, обеспечивая функционирование системы. В процессе формирования системы одни нейронные связи укрепляются, а другие ослабляются, обеспечивая обучаемость сети.

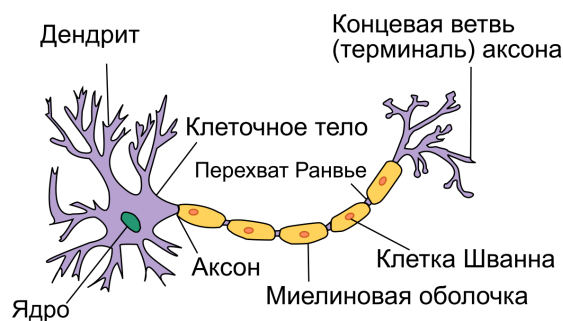


Рисунок 1.1 – Типичная структура нейрона

Искусственный нейрон представляет собой упрощенную модель биологического нейрона. На вход подаются n -мерные вектор значений $X = (x_1, \dots, x_n)$ и вектор весов $W = (w_1, \dots, w_n)$. Входные значения несут информацию, а веса указывают на "силу" межнейронных связей. Процесс подбора весов называется обучением нейронной сети. Значение выхода нейрона вычисляется по формуле:

$$out(x) = \sigma\left(\sum_{i=1}^n x_i w_i\right)$$

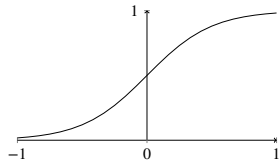
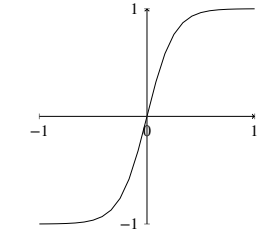
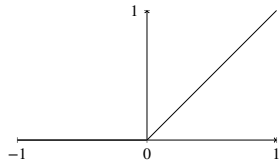
Где σ - функция активации.

1.2.2 Активационная функция

При вычислении взвешенной суммы входов нейрона, результат может принимать абсолютно любое значение $x \in (-\infty; +\infty)$, что препятствует дальнейшим вычислениям. Активационная функция нейрона обеспечивает нормализацию посчитанной суммы, таким образом, что значение выхода нейрона всегда принадлежит некоторому, заранее заданному, диапазону, часто: $(0; 1)$ или $(-1; 1)$. Для многих моделей нейронных сетей также требуется, чтобы активационная функция была нелинейной, монотонной и непрерывно-дифференцируемой на всей области определения.

Существует большое количество функций активации. Наиболее распространенные из них представлены в табл. 1.1

Таблица 1.1 – Нелинейные активационные функции

Название	Функция	Вид
Сигмоидная	$\sigma(x) = \frac{1}{1+e^{-x}}$	
Гиперболический тангенс	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	
ReLU	$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0. \end{cases}$	

1.2.3 Структура нейронной сети

Множества нейронов формируют слои, слои в свою очередь формируют нейронную сеть. Входной слой получает данные, обрабатывает и передает нейронам скрытого слоя. Аналогично сработает каждый последующий слой вплоть до выходного.

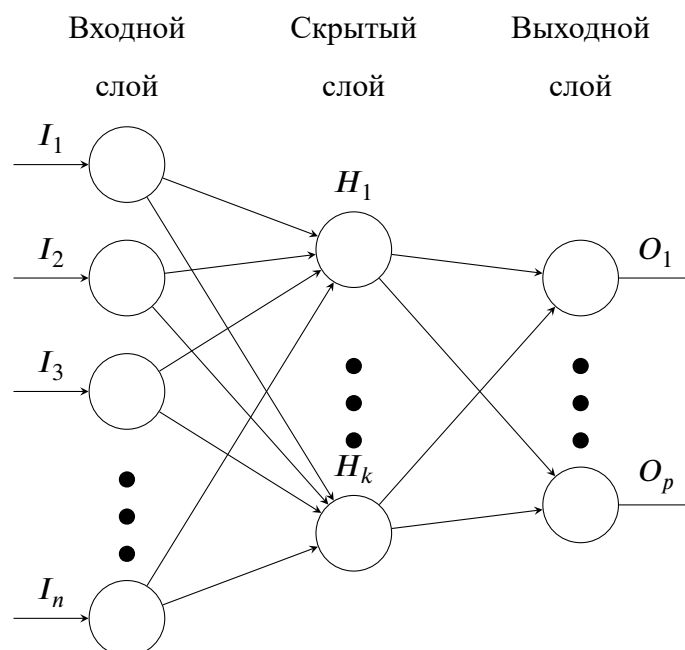


Рисунок 1.2 – Схема простой нейронной сети

1.2.4 Глубокое машинное обучение

Глубокое обучение (Deep Learning) - область машинного обучения, в которой для решения задач используются глубокие нейронные сети. Под глубокими нейронными сетями понимают сети с большим количеством скрытых слоев.

1.2.5 Сверточные нейронные сети

Сверточные нейронные сети (Convolutional Neural Network) активно применяются в задачах распознавания образов.

1.2.6 Проблемы обучения нейронных сетей

???

1.3 Применение нейронных сетей в задачах распознавания изображений

???

1.4 Инструменты машинного обучения

TensorFlow ???

Keras ???

???

2 ПОСТАНОВКА ЗАДАЧИ

3 ПРОЕКТИРОВАНИЕ СИСТЕМЫ ДЛЯ ???

4 ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ

5 ПРИМЕРЫ РАБОТЫ ПРОГРАММЫ

6 АНАЛИЗ РАБОТЫ ПРОГРАММЫ

ЗАКЛЮЧЕНИЕ

СПИСОК ЛИТЕРАТУРЫ

1. *Farinella G. M., Battiato S., Cipolla R.* Advanced Topics in Computer Vision. — Springer, 2013. — С. 475.
2. *Michelucci U.* Applied Deep Learning: A Case-Based Approach to Understanding Deep Neural Networks. — Apress, 2018. — С. 431.
3. *Nielsen M. A.* Neural Networks and Deep Learning. / Determination Press. — 2015. — URL: <http://neuralnetworksanddeeplearning.com/>.
4. *Singh P., Manure A.* Learn TensorFlow 2.0: Implement Machine Learning and Deep Learning Models with Python. — Apress, 2020. — С. 195.
5. *Жерон О.* Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow. — ООО "Альфа-книга", 2018. — С. 688.
6. *Клетте Р.* Компьютерное зрение. Теория и алгоритмы. — ДМК Пресс, 2019. — С. 506.
7. *Конец Д.* Классические задачи Computer Science на языке Python. — Питер, 2020. — С. 256.
8. *Нишант Ш.* Машинное обучение и TensorFlow. — Питер, 2019. — С. 336.
9. *Шолле Ф.* Глубокое обучение на Python. — Питер, 2018. — С. 400.

СПИСОК ИЛЛЮСТРАТИВНОГО МАТЕРИАЛА

Рисунок 1.1	Типичная структура нейрона	6
Рисунок 1.2	Схема простой нейронной сети	8

СПИСОК ТАБЛИЧНОГО МАТЕРИАЛА

Таблица 1.1	Нелинейные активационные функции	7
-------------	--	---

ПРИЛОЖЕНИЕ А

Листинг 1

```
1 from __future__ import absolute_import, division, print_function,
    unicode_literals
2
3 import numpy as np
4 import cv2
5 from tensorflow.keras.callbacks import ModelCheckpoint
6 from tensorflow.keras.layers import Conv2D, Flatten, MaxPooling2D,
    Dense, Dropout
7 from tensorflow.keras.models import Sequential
8 from tensorflow.keras.preprocessing.image import ImageDataGenerator,
    img_to_array, load_img, array_to_img
9 import random, os, glob
10 import matplotlib.pyplot as plt
11
12 DIR_PATH = '/home/alexandr/dev/datasets/garbage-classification/
    garbage_classification/Garbage_classification'
13 img_list = glob.glob(os.path.join(DIR_PATH, '*/*.jpg'))
14
15
16 def load_data():
17     train=ImageDataGenerator(
18         horizontal_flip=True, vertical_flip=True, validation_split
19         =0.1, rescale=1./255,
20         shear_range=0.1,
21         zoom_range=0.1,
22         width_shift_range=0.1,
23         height_shift_range=0.1)
24     test=ImageDataGenerator(rescale=1/255, validation_split=0.1)
25     train_generator=train.flow_from_directory(DIR_PATH, target_size
        =(300,300), batch_size=32,
        class_mode='categorical',
        subset='training')
```

```

26 test_generator=test.flow_from_directory(DIR_PATH,target_size
    =(300,300),batch_size=32,
27                                     class_mode='categorical',
                                     subset='validation')
28 labels = (train_generator.class_indices)
29 labels = dict((v,k) for k,v in labels.items())
30
31 print(labels)
32
33 model=Sequential()
34
35 model.add(Conv2D(32,(3,3), padding='same',input_shape=(300,300,3)
    ,activation='relu'))
36 model.add(MaxPooling2D(pool_size=2))
37 model.add(Conv2D(64,(3,3), padding='same',activation='relu'))
38 model.add(MaxPooling2D(pool_size=2))
39 model.add(Conv2D(32,(3,3), padding='same',activation='relu'))
40 model.add(MaxPooling2D(pool_size=2))
41 model.add(Flatten())
42 model.add(Dense(64,activation='relu'))
43 model.add(Dense(6,activation='softmax'))
44
45 filepath="trained_model.h5"
46 checkpoint1 = ModelCheckpoint(filepath, monitor='val_acc',
    verbose=1, save_best_only=True, mode='max')
47 callbacks_list = [checkpoint1]
48 model.summary()
49 model.compile(loss='categorical_crossentropy', optimizer='adam',
    metrics=['acc'])
50 model.fit(train_generator, epochs=150, steps_per_epoch=2276//32,
    validation_data=test_generator,
51             validation_steps=251//32,callbacks=callbacks_list
    )
52 model.save("model.h5")
53
54 if __name__ == '__main__':

```

```
55     os.environ['TF_FORCE_GPU_ALLOW_GROWTH'] = 'true'
56     load_data()
```