МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**
**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**
**Кафедра анализа данных и искусственного интеллекта**

**КУРСОВАЯ РАБОТА**

# АВТОМАТИЗАЦИЯ СБОРА ДАННЫХ И ОБУЧЕНИЯ НЕЙРОННОЙ СЕТИ

Работу выполнил _____ М.И. Ковалев
(подпись)

Направление подготовки 09.03.03 Прикладная информатика

Направленность (профиль) Прикладная информатика в экономике

Научный руководитель
д-р. техн. наук, зав. каф. _____ А.В. Коваленко
(подпись)

Нормоконтролер
канд. физ.-мат. наук, доц. _____ Г.В. Калайдина
(подпись)

Краснодар
2023

# РЕФЕРАТ

Курсовая работа 28 с., 6 рис., 8 источников.

НЕЙРОННАЯ СЕТЬ, СВЕРТОЧНЫЕ НЕЙРОННЫЕ СЕТИ, ОБУЧЕНИЕ НЕЙРОННОЙ СЕТИ, SELENIUM, КЛАССИФИКАЦИЯ

Объектом исследования данной курсовой работы являются автоматический процесс сбора и предобработки данных, разработка архитектуры нейронной сети и ее обучение на полученном наборе данных. Весь этот комплекс задач направлен на создание автоматической системы, собирающей данные, на которых нейронная сеть будет обучаться без вмешательства человека.

Цель работы: разработка автоматической системы сбора данных, а также разработка алгоритма, по которому созданная нейронная сеть автоматически обучается на полученных данных.

В результате курсовой работы были созданы автоматическая система сбора данных и нейронная сеть, самостоятельно обучающаяся на новых данных.

**СОДЕРЖАНИЕ**

# ВВЕДЕНИЕ

В настоящее время получило широкое применение решение задач с помощью нейронных сетей. Но для каждой нейронной сети нужен свой обучающий набор данных. Обучающий набор данных – это набор наблюдений, где записаны все значения как входных, так и выходных переменных.

Достаточно актуальной становится проблема сбора данных для обучения, ведь данные зачастую подбираются вручную.

Целью курсовой работы является разработка автоматической системы сбора данных, а также разработка нейронной сети, автоматически обучающаяся на полученных данных.

Данная цель предопределила решение следующих задач:

– изучение различных нейронных сетей, правил их функционирования,

– создание алгоритма сбора данных «Scrap» из браузера «Google Chrome»,

– построение нейронной сети «CatsDogs», которая распознает изображение кота или пса,

– создание автоматической программы «AutoNN», которая использует разработанный алгоритм для получения данных и с их помощью обучает созданную нейронную сеть.

Курсовая работа состоит из трех глав, введения, заключения, списка использованной литературы, содержащего 8 наименований.

В первой главе подробно описаны основные понятия нейронной сети, современных нейронных сетей, сбора данных для обучения нейронной сети.

Во второй главе описаны основные программные средства для разработки нейронной сети «CatsDogs» и алгоритма сбора данных «Scrap»,

дана краткая характеристика языка программирования Python и описаны основные библиотеки.

В третей главе описано создание нейронной сети «CatsDogs», алгоритма сбора данных «Scrap» и система автоматизации данных процессов «AutoNN».

В заключение подведен основной итог работы.

## 1 Нейронная сеть

Нейронная сеть – это математическая модель, которая стремится имитировать работу нейронов в человеческом мозге. Она состоит из множества искусственных нейронов, соединенных между собой. Каждый нейрон принимает входные данные, а затем обрабатывает их и передает результаты следующим нейронам.

Искусственный нейрон, в свою очередь, состоит из нескольких элементов: сумматор, функции активации и весовых коэффициентов. На вход сумматора поступают взвешенные данные, затем происходит их обработка с помощью функции активации, и результат передается следующим нейронам. Весовые коэффициенты определяют силу взвешивания входных данных в каждом нейроне.

Нейронные сети могут использоваться для различных задач, таких как распознавание образов, обработка естественного языка, рекомендательные системы, анализ данных и многие другие. Например, для всех этих задач могут использоваться сверточные нейронные сети, которые хорошо справляются с обработкой изображений, изменением размера изображений и классификацией объектов.

Обучение нейронной сети может быть реализовано различными методами, включая обратное распространение ошибки, метод опорных векторов и генетические алгоритмы. При обучении важно выбрать правильный тип нейронной сети для решения задачи и настроить параметры обучения, включая весовые коэффициенты, тренировочный набор данных и функцию активации.

Современные нейронные сети обычно строятся с использованием глубокого обучения, которое позволяет построить многократные слои иерархии признаков, которые необходимы для решения сложных задач. Этот подход позволяет создавать нейронные сети, которые могут обучаться на

огромном объеме данных и достигать высокого уровня точности в решении задач.

### 1.1 Современные нейронные сети

Современные нейронные сети – это компьютерные системы, которые имитируют работу человеческого мозга. Они используются для решения широкого круга задач, включая распознавание образов, анализ данных, генерацию текстов и изображений, управление роботами и машинное обучение.

Современные нейронные сети основаны на глубоком обучении (deep learning), которое позволяет алгоритмам автоматически анализировать и находить закономерности в больших объёмах данных, выполняя сложные задачи куда более эффективно, чем человеческий мозг.

Среди самых продвинутых нейронных сетей, которые используются в настоящее время, можно выделить например:

– Сверточные нейронные сети (Convolutional Neural Networks, CNN), которые широко используются для обработки изображений.

– Рекуррентные нейронные сети (Recurrent Neural Networks, RNN), которые используются для обработки последовательностей данных (например, текста или звука).

– Трансформеры (Transformers), которые используются для обработки текстов.

– Генеративно-состязательные сети (Generative Adversarial Networks, GAN), которые используются для генерации изображений, звука и других типов данных.

Существует огромное количество различных типов нейронных сетей, каждая из которых предназначена для решения определенных задач. Развитие

нейронных сетей продолжается, и высокие ожидания относительно их применения делают их одной из ключевых технологий будущего.

В настоящее время люди делают упор на создание новых более эффективных видах нейросетей, и мало кто занимается модернизацией алгоритмов сбора данных.

### 1.2 Сбор данных для обучения нейронной сети

Автоматизация сбора данных для нейронной сети и обучения нейронной сети на полученных данных важны для достижения высокой точности нейронных сетей в решении задач.

Для автоматизации сбора данных можно использовать различные методы, включая веб-скрэпинг, API запросы, парсинг данных и т. д. Например, при сборе текстовых данных, веб-скрэпинг позволяет собирать данные из различных источников, а API запросы – получать данные из сторонних приложений и сервисов.

Для обучения нейронной сети на полученных данных нужно сначала подготовить данные, что может включать в себя предобработку данных, как очистка и преобразование, а также разбиение на обучающую и тестовую выборки.

Зачастую даже при создании нейронных сетей со сложной архитектурой, таких как «ChatGPT» процесс формирования обучающих данных выполняется вручную, что является трудозатратной операцией.

# 2. Технологии разработки

## 2.1 Язык программирования Python

Python – один из самых популярных языков программирования, с помощью которого можно решать самые разные задачи. Именно поэтому он так распространен и для создания нейронных сетей [6].

Язык позволяет разрабатывать сложные алгоритмы за короткое время. Его отличают простота, лаконичность и выразительность. Помимо этого, он позволяет производить быстрые вычисления.

Нейронные сети – преимущественно небольшие программы, но при этом существует необходимость часто изменять их, подбирая наилучшую архитектуру, предобработку данных и другие параметры. Именно на Python есть библиотеки для более простого написания нейронных сетей.

## 2.2 Основные библиотеки

### 2.2.1 Keras

Keras – это библиотека для языка программирования Python, которая предназначена для глубокого машинного обучения. Она позволяет быстрее создавать и настраивать модели – схемы, по которым распространяется и подсчитывается информация при обучении. Но сложных математических вычислений Keras не выполняет и используется как надстройка над другими библиотеками [7].

Keras с версии 2.3 – это надстройка над библиотекой TensorFlow, которая нужна для машинного обучения. TensorFlow выполняет все низкоуровневые вычисления и преобразования и служит своеобразным движком, математическим ядром. Keras же управляет моделями, по которым проходят вычисления.

До версии 2.3 Keras мог использовать в качестве движка вычислительные различные библиотеки. Но в новых версиях поддержка прекратилась, теперь библиотека работает только с TensorFlow.

Keras создавалась как гибкая модульная библиотека, которую легко настраивать и модифицировать. Она бесплатная, у нее открытый исходный код, который может посмотреть любой человек.

Keras имеет узкую специализацию. Это инструмент для специалистов по машинному обучению, которые работают с языком Python: именно его чаще всего используют благодаря удобству математических вычислений. Keras применяют разработчики, которые создают, настраивают и тестируют системы машинного обучения и искусственного интеллекта, в первую очередь нейронные сети.

### 2.2.2 TensorFlow

TensorFlow – это библиотека с открытым исходным кодом для разработки и обучения нейронных сетей. Она предоставляет широкий спектр инструментов и возможностей для создания и оптимизации различных моделей глубокого обучения. Вот некоторые особенности и функции TensorFlow:

– Графовое представление: TensorFlow использует графовое представление для определения и вычисления операций. Операции представляют собой узлы графа, а данные – ребра графа. Это позволяет оптимизировать выполнение операций и эффективно использовать ресурсы.

– Автоматическое дифференцирование: TensorFlow автоматически вычисляет градиенты для обратного распространения ошибки. Это облегчает обучение моделей с использованием градиентного спуска и других методов оптимизации.

– Многоуровневые API: TensorFlow предлагает различные уровни

абстракции для создания моделей нейронных сетей. Он содержит высокоуровневые API, такие как Keras, которые упрощают создание моделей для быстрого прототипирования. Одновременно TensorFlow предоставляет и низкоуровневые API, позволяющие более гибко настраивать модели и выполнять более сложные операции.

– Распределенное обучение: TensorFlow поддерживает распределенное обучение на нескольких устройствах или кластерах с использованием TensorFlow Distributed. Это позволяет эффективно использовать вычислительные ресурсы и ускорить обучение моделей.

– Поддержка различных типов моделей: TensorFlow поддерживает различные типы моделей, включая сверточные нейронные сети (CNN), рекуррентные нейронные сети (RNN), генеративные модели (GAN), модели с подкреплением (RL) и другие. Это позволяет реализовывать разнообразные архитектуры и решать различные задачи машинного обучения.

– Высокая производительность: TensorFlow предлагает оптимизированные вычисления, включая использование графических процессоров (GPU) и специализированных аппаратных средств, таких как Google Tensor Processing Units (TPU). Это позволяет обрабатывать большие объемы данных и ускорять обучение моделей.

В целом, TensorFlow является мощной и гибкой библиотекой для разработки нейронных сетей и выполнения задач машинного обучения. Она предлагает разнообразные функции, поддерживает различные типы моделей и обладает обширной экосистемой, что делает ее популярным выбором для исследователей и разработчиков в области глубокого обучения.

### 2.2.3 Selenium

Selenium – это библиотека с открытым исходным кодом, используемая для автоматизации веб-браузера. Она предоставляет разработчикам

инструменты для взаимодействия с веб-страницами, выполнения действий пользователя, извлечения данных и автоматизации веб-скрэппинга. Вот некоторые особенности и функции Selenium:

– Управление браузером: С помощью Selenium можно контролировать различные веб-браузеры, включая Chrome, Firefox, Safari и другие. Библиотека предоставляет API для открытия браузера, загрузки веб-страниц, выполнения действий пользователя (например, клики, заполнение форм) и навигации по страницам.

– Взаимодействие с элементами страницы: Selenium позволяет находить элементы на веб-странице с использованием различных селекторов, таких как CSS-селекторы и XPath. Это позволяет получать доступ к тексту, атрибутам и другим свойствам элементов страницы, а также выполнять действия, такие как клики, ввод текста и отправка форм.

– Ожидание и временные задержки: Selenium предоставляет возможности ожидания, которые позволяют ждать определенных событий на странице, таких как загрузка элементов или изменение состояния. Это полезно для обработки динамических страниц или ожидания появления определенного элемента перед его взаимодействием.

– Управление окнами и фреймами: Selenium позволяет переключаться между окнами браузера и фреймами внутри веб-страницы. Это полезно, когда необходимо выполнить действия в определенном окне или фрейме.

– Обработка исключений и ошибок: Selenium обеспечивает обработку исключений и ошибок, возникающих во время выполнения операций. Это позволяет эффективно обрабатывать ситуации, когда элемент не может быть найден или действие не может быть выполнено.

– Поддержка различных языков программирования: Selenium поддерживает несколько языков программирования, включая Python, Java, C#,

Ruby и другие. Это дает возможность выбрать предпочтительный язык для разработки скриптов веб-скрэппинга.

– Интеграция с другими инструментами: Selenium может быть интегрирован с другими инструментами и библиотеками для обработки данных, анализа и визуализации результатов веб-скрэппинга. Например, данные, извлеченные из веб-страницы, могут быть сохранены в базе данных или файле для последующей обработки. Также Selenium может быть использован с библиотеками для анализа данных, такими как NumPy, pandas или Matplotlib, для обработки и визуализации полученных данных.

– Поддержка тестирования: Selenium часто используется для автоматизации тестирования веб-приложений. Он позволяет создавать и запускать автоматические тесты, которые могут проверять работу веб-сайта на различных устройствах и в различных условиях.

– Расширяемость: Selenium предоставляет возможность расширения его функциональности через плагины и расширения. Это позволяет добавлять дополнительные возможности, такие как блокировка рекламы, поддержка новых браузеров и т. д.

Selenium является мощной библиотекой для веб-скрэппинга и автоматизации браузера, которая предоставляет широкий спектр функций для работы с веб-страницами и элементами. Его гибкость и расширяемость делают его популярным инструментом не только для веб-скрэппинга, но и для других задач, таких как тестирование, анализ и визуализация данных.

### 2.2.4 Urllib

Библиотека urllib.request является частью стандартной библиотеки Python и предоставляет инструменты для работы с URL-адресами и сетевыми запросами. Она используется для получения данных из интернета, отправки

HTTP-запросов и управления сетевыми соединениями. Вот некоторые особенности и функции библиотеки urllib.request:

– Открытие URL-адресов: Модуль urllib.request позволяет открывать URL-адреса и получать данные из них. Вы можете передать URL-адрес в функцию urlopen(), и она вернет объект HTTPResponse, из которого вы сможете прочитать данные.

– Отправка HTTP-запросов: urllib.request позволяет отправлять различные типы HTTP-запросов, такие как GET, POST, PUT, DELETE и другие. Вы можете создать объект Request и передать его в функцию urlopen() для выполнения запроса.

– Обработка ответов: Полученные HTTP-ответы могут быть обработаны с помощью методов и свойств объекта HTTPResponse. Вы можете прочитать данные ответа, получить статусный код, заголовки и другую информацию.

– Работа с параметрами URL: urllib.request предоставляет функции для работы с параметрами URL, такие как кодирование и декодирование URL, извлечение параметров из URL и т. д.

– Управление сетевыми соединениями: urllib.request предоставляет функциональность для управления сетевыми соединениями, такую как установка пользовательских заголовков, установка таймаутов и управление прокси-серверами.

– Работа с файлами: urllib.request позволяет сохранять полученные данные в файлы на локальном диске. Вы можете использовать функцию urlretrieve() для загрузки файла по URL-адресу и сохранения его на диск.

– Обработка ошибок: urllib.request обрабатывает различные типы ошибок, такие как ошибки соединения, ошибки HTTP-ответов и другие. Вы можете использовать обработку исключений для обработки этих ошибок и выполнения соответствующих действий.

– Библиотека urllib.request предоставляет простой и удобный способ работать с сетевыми запросами и получать данные из интернета. Она широко используется для создания веб-скрэпперов, загрузки файлов, обмена данными с веб-серверами и других сценариев, требующих работы с HTTP-запросами.

### 2.2.5 Psutil

Библиотека psutil (Process and System Utilities) – это библиотека для Python, которая предоставляет удобный интерфейс для получения информации о процессах и системе компьютера. Она позволяет получать информацию о ресурсах процессора, памяти, дискового пространства, сетевой активности и других системных параметрах. Вот некоторые особенности и функции библиотеки psutil:

– Получение информации о процессах: psutil позволяет получить информацию о запущенных процессах на компьютере. Вы можете получить список процессов, их идентификаторы, имена, использование CPU и памяти, статусы и другие свойства. Также можно получить информацию о дочерних процессах, родительских процессах и т.д.

– Мониторинг системных ресурсов: psutil предоставляет функции для мониторинга использования ресурсов процессора, памяти, дискового пространства и сетевой активности. Вы можете получить информацию о загрузке процессора, доступном и использованном объеме памяти, статусе дисков, сетевых соединениях и т.д.

– Управление процессами: psutil позволяет выполнять действия над процессами, такие как запуск, остановка, приостановка и возобновление процессов. Вы можете отправлять сигналы процессам, изменять их приоритеты, аффинитеты и другие параметры.

– Работа с сетевыми соединениями: psutil позволяет получить информацию о сетевых соединениях, таких как IP-адреса, порты, протоколы и

состояния соединений. Вы также можете получить статистику сетевой активности, такую как количество отправленных и принятых пакетов, количество ошибок и т.д.

– Работа с дисками и файловыми системами: psutil предоставляет функции для получения информации о дисках, монтированных файловых системах, свободном и занятом дисковом пространстве. Вы можете получить список дисков, информацию о каждом диске, использование файловых систем и другую связанную информацию.

– Информация о системе: psutil позволяет получить информацию о системе компьютера, такую как имя узла, версия операционной системы, количество и тип процессоров, архитектура процессора, количество и доступную память и другую системную информацию.

– Управление энергопотреблением: psutil предоставляет функции для управления энергопотреблением компьютера. Вы можете получить информацию о текущем режиме энергосбережения, установить режим энергосбережения, перезагрузить или выключить компьютер и другие операции.

– Работа с процессором и системным временем: psutil позволяет получить информацию о текущей загрузке процессора, количестве ядер, частоте работы процессора и других связанных параметрах. Также можно получить информацию о текущем системном времени, продолжительности работы компьютера и других временных параметрах.

Библиотека psutil предоставляет мощные инструменты для мониторинга и управления процессами и ресурсами системы. Она широко используется в различных областях, таких как системное администрирование, мониторинг, разработка программного обеспечения и других сценариях, где требуется получение информации о системе и процессах. Благодаря простому и понятному интерфейсу psutil становится удобным инструментом для работы с системными данными в Python.

# 3 Разработка системы для обучения нейронной сети без вмешательства человека

## 3.1 Создание нейронной сети «CatsDogs»

### 3.1.1 Модель нейронной сети

Разработанная модель представляет собой сверточную нейронную сеть (Convolutional Neural Network, CNN). Вот подробное описание архитектуры этой модели:

– Слой Conv2D с 32 фильтрами размером (3, 3), функцией активации 'relu' и с тем же размером входных данных (input_shape=(50, 50, 3)). Этот слой выполняет операцию свертки на входных данных, извлекая признаки изображений.

– Слой MaxPooling2D с размером пулинга (2, 2) и методом заполнения 'same'. Этот слой уменьшает размерность выходных данных сверточного слоя путем выбора максимального значения из каждой области пулинга.

– Слой Conv2D с 64 фильтрами размером (3, 3), функцией активации 'relu' и методом заполнения 'same'. Этот слой выполняет еще одну операцию свертки для извлечения более высокоуровневых признаков изображений.

– Еще один слой MaxPooling2D с размером пулинга (2, 2) и методом заполнения 'same'. Этот слой продолжает уменьшать размерность выходных данных.

– Следующий слой Flatten преобразует многомерные выходные данные предыдущего слоя в одномерный вектор. Это необходимо для передачи данных в полносвязный слой.

– Полносвязный слой Dense с 500 нейронами и функцией активации 'relu'. Этот слой выполняет операцию матричного умножения между

входными данными и весами, а затем применяет нелинейную функцию активации 'relu' к полученным результатам.

– Второй полносвязный слой Dense с 2 нейронами и функцией активации 'softmax'. Этот слой используется для классификации и возвращает вероятности принадлежности входных данных к двум классам (бинарная классификация).

– Наконец, модель компилируется с оптимизатором 'adam', функцией потерь 'binary_crossentropy' (так как это бинарная классификация) и метрикой 'accuracy' для оценки производительности модели.

Код, реализующий создание нейронной сети представлен на рисунке 1:

```
cnn = Sequential()
cnn.add(Conv2D(32, (3,3), input_shape = (50, 50, 3), padding='same', activation = 'relu'))
cnn.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
cnn.add(Conv2D(64, (3,3), padding='same', activation = 'relu'))
cnn.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
cnn.add(Flatten())
cnn.add(Dense(500, activation = 'relu'))
cnn.add(Dense(2, activation = 'softmax'))
cnn.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])
print(cnn.summary())
```

Рисунок 1 – Код нейронной сети

Вызов print(cnn.summary()) выводит сводку (summary) архитектуры модели, включающую информацию о форме входных и выходных данных каждого слоя и общее количество обучаемых параметров.

### 3.1.2 Алгоритм обучения

Оптимизатор 'adam' (Adaptive Moment Estimation) является одним из популярных алгоритмов оптимизации, широко используемых в глубоком обучении. Он сочетает в себе преимущества двух других оптимизаторов: адаптивного градиентного спуска (Adaptive Gradient Descent, AdaGrad) и стохастического градиентного спуска с инерцией (Stochastic Gradient Descent with Momentum).

Алгоритм 'adam' обновляет веса модели, основываясь на оценках первого и второго моментов градиентов. Вот некоторые особенности и параметры 'adam':

– Адаптивный шаг обучения: 'adam' автоматически адаптирует шаг обучения (learning rate) для каждого параметра модели на основе оценок первого момента (среднего) и второго момента (накопленной суммы квадратов) градиентов. Это позволяет алгоритму эффективно обновлять параметры с разными шагами в зависимости от их значимости.

– Коррекция смещения: В начале обучения оценки первого и второго моментов градиентов могут быть смещены к нулю из-за инициализации весов. Для борьбы с этим эффектом 'adam' применяет коррекцию смещения, чтобы устранить смещение инициализации и обеспечить более точные оценки моментов.

– Градиентный спуск с инерцией: 'adam' также использует инерцию, которая помогает ускорить обучение, особенно в случае больших и плохо обусловленных функций потерь. Инерция позволяет накапливать импульс градиентов из предыдущих обновлений, чтобы плавно двигаться вдоль градиентного спуска, преодолевая локальные минимумы.

Вот некоторые параметры 'adam', которые можно настроить:

– learning_rate (или `lr`): Шаг обучения. Определяет, насколько сильно обновляются веса модели при каждом шаге. Значение по умолчанию

обычно составляет 0.001.

– beta_1: Значение по умолчанию составляет 0.9. Этот параметр контролирует влияние оценки первого момента градиента на обновление весов модели. Чем ближе значение beta_1 к 1, тем больше влияние предыдущих градиентов на текущее обновление.

– beta_2: Значение по умолчанию составляет 0.999. Этот параметр контролирует влияние оценки второго момента градиента на обновление весов модели. Чем ближе значение beta_2 к 1, тем больше влияние предыдущих квадратов градиентов на текущее обновление.

– epsilon: Значение по умолчанию составляет 1e-7. Это небольшое число добавляется к знаменателю при делении для численной стабильности. Оно предотвращает деление на ноль и сглаживает результаты.

– decay: Параметр, контролирующий скорость уменьшения шага обучения. Значение по умолчанию равно 0. Установка ненулевого значения позволяет постепенно уменьшать шаг обучения во время обучения.

Оптимизатор 'adam' является эффективным выбором для многих задач глубокого обучения. Он демонстрирует хорошую скорость сходимости и устойчивость к различным градиентным сценариям.

### 3.2 Разработка алгоритма для сбора данных

Была разработана программа «Scrap», предназначенная для сохранения изображений кошек и собак из результатов поиска в Интернете. Вот как она работает:

1) Импортируются необходимые модули, такие как os, time, urllib.request и selenium. os используется для работы с файловой системой, time - для задержек между операциями, urllib.request – для загрузки изображений, selenium – для автоматизации веб-браузера Chrome.

2) Определяются константы и переменные:

– number_of_images – количество изображений, которые нужно скачать для каждого запроса.

– GET_IMAGE_TIMEOUT – таймаут для получения изображения.

– SLEEP_BETWEEN_INTERACTIONS – пауза между взаимодействиями с браузером.

– SLEEP_BEFORE_MORE – пауза перед загрузкой большего количества изображений.

– IMAGE_QUALITY – качество сохраняемых изображений.

– search_terms – список поисковых запросов, в данном случае "dog" и "cat".

– wd – экземпляр веб-драйвера Chrome, используемый для автоматизации браузера.

3) Создается класс timeout, который будет использоваться для обработки таймаутов.

4) Определяются функции image_urls_to_file и image_urls_from_file, которые служат для сохранения и загрузки списка URL изображений в файл.

5) Функция fetch_image_urls используется для получения URL изображений из результатов поиска Google. Она принимает следующие параметры:

– filename – имя файла, в котором хранятся ранее полученные URL.

– query – поисковый запрос.

– max_links_to_fetch – максимальное количество URL для получения.

– wd – экземпляр веб-драйвера Chrome.

– sleep_between_interactions – пауза между взаимодействиями с браузером.

– Функция выполняет следующие действия:

а) Открывает страницу поиска Google с заданным запросом.

б) Получает список URL миниатюр изображений.

в) Прокручивает страницу и получает дополнительные URL до достижения максимального количества.

г) Возвращает список полученных URL.

6) Функция persist_image сохраняет изображение по указанному URL. Она принимает следующие параметры:

– folder_path – путь к папке для сохранения изображений.

– search_term – поисковый запрос.

– url – URL изображения.

– index – индекс изображения.

– Функция выполняет следующие действия:

а) Открывает URL изображения.

б) Создает файловый путь для сохранения изображения на основе поискового запроса и индекса.

в) Сохраняет изображение по указанному пути.

г) Выводит сообщение об успешном сохранении.

7) Функция search_and_download выполняет поиск и загрузку изображений для заданного поискового запроса. Она принимает следующие параметры:

– search_term – поисковый запрос.

– number_images – количество изображений для загрузки.

– Функция выполняет следующие действия:

а) Создает папку для сохранения изображений, если она не существует.

б) Используя веб-драйвер Chrome, вызывает функцию fetch_image_urls для получения URL изображений.

в) Если получены URL изображений, происходит сохранение каждого изображения с помощью функции persist_image.

8) В основной части программы происходит итерация по списку search_terms, и для каждого поискового запроса вызывается функция search_and_download, указывая количество изображений для загрузки.

Эта программа выполняет запросы к поисковой системе Google, получает URL изображений и сохраняет их на локальном диске.

### 3.3 Разработка программы для автоматизации процесса скрэппинга и обучения нейронной сети «AutoNN»

Для использования в автоматизированном были программы скомпилированы программы «Scrap» и «CatsDogs».

Это было сделано, чтобы оставить возможность использования данных программ по отдельности.

Разработанная программа «AutoNN» выполняет следующие действия:

1) Импортируются необходимые модули: os, cv2, subprocess, time и psutil. os используется для работы с операционной системой, cv2 – для обработки изображений, subprocess – для выполнения внешних команд, time – для задержек, а psutil – для работы с процессами.

2) Определяются функции:

– processIsWorkin() – проверяет, работает ли процесс с именем "scrap.exe" в системе. Функция использует модуль psutil для получения списка выполняющихся процессов и проверяет наличие процесса с именем "scrap.exe".

– process() – запускает исполняемый файл "scrap.exe" в папке "output\scrap" с помощью функции os.startfile().

– process1() – запускает исполняемый файл "catsdogs.exe" в папке "output\catsdogs" с помощью функции os.startfile().

– processIsWorkin1() – аналогично processIsWorkin(), но проверяет наличие процесса с именем "catsdogs.exe".

– start_process() – функция, в которой происходит запуск и проверка двух процессов. Внутри цикла while происходит запуск процесса "scrap.exe", а затем проверяется его работа с помощью processIsWorkin(). Если процесс завершен, выводится сообщение "scrapping is done" и цикл прерывается. Затем запускается процесс "catsdogs.exe" и также проверяется его работа с помощью processIsWorkin1(). Если процесс завершен, выводится сообщение "nn create" и цикл завершается.

3) В конце программы вызывается функция start_process() для запуска последовательности процессов.

Программа выполняет запуск и проверку двух исполняемых файлов ("scrap.exe" и "catsdogs.exe") в определенных папках и выводит соответствующие сообщения о выполнении процессов.

### 3.4 Апробация и тестирование

Программа «AutoNN» была успешно запущена и работала на протяжении 6 часов (рисунок 2 и 3).

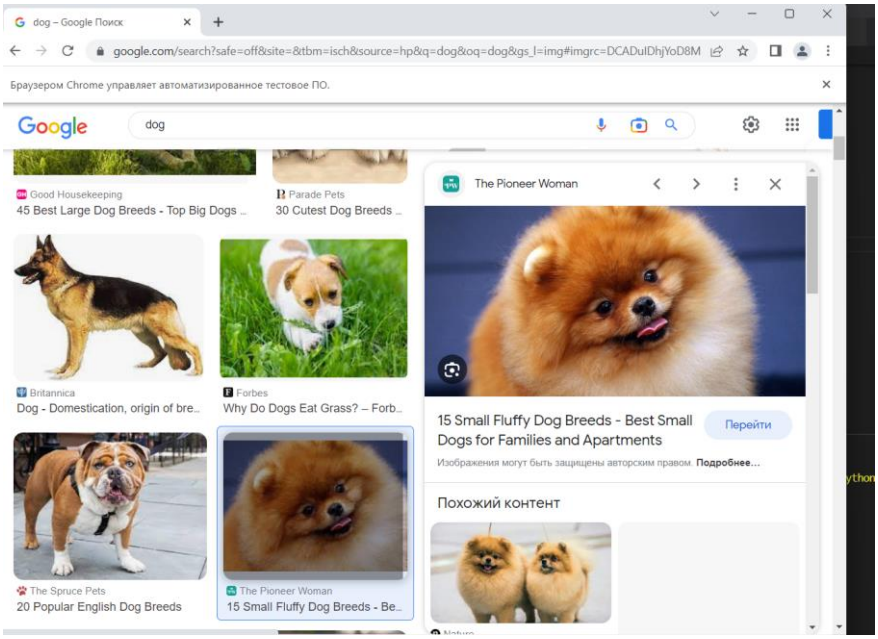


Рисунок 2 – Процесс парсинга

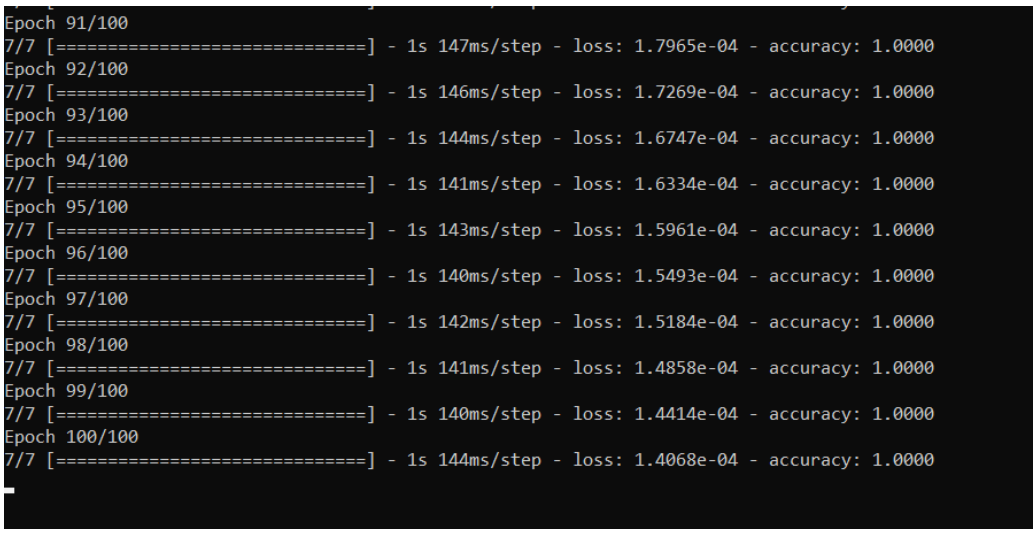


Рисунок 3 – Процесс обучения нейронной сети

После завершения работы размер обучающей выборки составил 3000 элементов, а размер тестовой составил 750 элементов (рисунок 4).

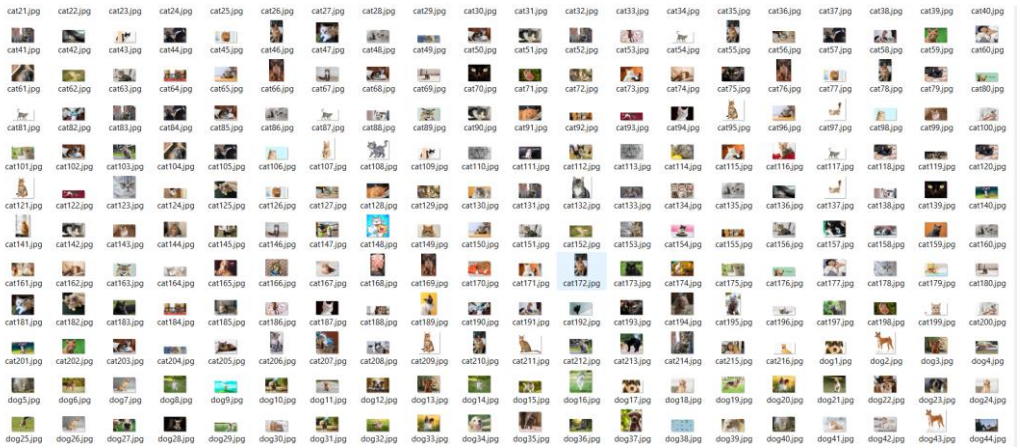

Рисунок 4 – Фрагмент обучающей выборки

Процент распознавания составил 95% и 91% для обучающей и тестовой выборок соответственно. Также был проведен тест полученной после работы программы «AutoNN» нейронной сети на изображении, которой не было ни в тестовой, ни в обучающей выборке (рисунок 5). Оно было распознано верно (рисунок 6).

Рисунок 5 – Изображение кота, которого не было ни в обучающей, ни в тестовой выборке

```
PS C:\Users\mishk\Downloads\nnscrap-main\nnscrap-main> &
/nnscrap-main/test.py
cat
PS C:\Users\mishk\Downloads\nnscrap-main\nnscrap-main>
```

Рисунок 6 – Результат теста на изображении

# ЗАКЛЮЧЕНИЕ

В курсовой работе была реализована задача создания автоматической системы, собирающей данные, на которых нейронная сеть будет обучаться без вмешательства человека.

Была осуществлена разработка автоматической системы сбора данных «Scrap», а также разработка алгоритма «AutoNN», по которому созданная нейронная сеть «CatsDogs» автоматически обучается на полученных данных.

В итоге разработки с помощью автоматизации процесса сбора данных и обучения нейронной сети удалось добиться высокой точности предсказаний изображения котов и собак.

Стоит подчеркнуть актуальность выбранного подхода. Из-за необходимости для создания корректных нейронных сетей возникает необходимость формирования объемных обучающих и тестовых выборок. Данный процесс занимает много времени и высоких мощностей ЭВМ. Предложенный подход позволяет избежать данных проблем. Кроме того, используемый метод можно использовать в похожих задачах для сохранения актуальности данных. Создание более сложных автоматизированных алгоритмов сбора данных в нейросетевых технологиях может позволить формировать наборы данных сколь-угодно больших размеров и, как следствие, поспособствовать развитию искусственного интеллекта, способного обучаться без вмешательства человека.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Эндрю Тарсак, Грокаем глубокое обучение / Эндрю Тарсак. СПБ.: Питер, 2021 – 352 с. – ISBN 978-5-4461-1334-7.

2. Из чего состоит нейросеть: сайт – URL: https://forklog.com/cryptorium/ai/chto-takoe-nejronnaya-set (дата обращения 02.05.2023).

3. Структура и принцип работы полносвязных нейронных сетей: сайт – URL: https://proproprogs.ru/neural_network/struktura-i-princip-raboty-polnosvyaznyh-neyronnyh-setey (дата обращения 03.05.2023).

4. Нейронные сети. – викиконспекты: сайт – URL: https://neerc.ifmo.ru/wiki/index.php?title=Нейронные_сети,_перцептрон (дата обращения 20.12.2022).

5. Нейронные сети: распознавание образов и изображений с помощью ИИ: сайт – URL: https://center2m.ru/ai-recognition (дата обращения 03.05.2023).

6. Нейронные сети на Python: как всё устроено: сайт – URL: https://gb.ru/blog/nejronnye-seti-python/(дата обращения 03.05.2023).

7. Распознавание изображений на Python с помощью TensorFlow и Keras: сайт – URL: https://evileg.com/ru/post/619/#header_TensorFlow_/_Keras (дата обращения 20.12.2022).

8. Selenium автоматизация браузера в Python: сайт – URL: https://selenium-python.com/ (дата обращения 03.05.2023).