УДК 004

Распознавание изображений с помощью библиотеки brain.js на языке программирования JavaScript

Стрельцова Марина Николаевна
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
В настоящий момент в мире бурно развивается новая прикладная область математики, специализирующаяся на искусственных нейронных сетях. Актуальность исследования в этом направлении обусловлена тем, что применение нейронных сетей возможно во всех сферах жизни. Распознавание текста с фотографий, видео, аудио и перевод рукописных текстов, оцифровка с бумажных носителей и это далеко не все возможности применения нейронных сетей для облегчения жизни человека. В данной статье рассмотрена реализация интерфейса по распознаванию изображений с помощью библиотеки brain.js на языке программирования JavaScript.
Ключевые слова: нейронная сеть, распознавание, JavaScript, brain.js.

Pattern recognition using the brain.js library in the JavaScript programming language

Streltsova Marina Nikolaevna
Sholom-Aleichem Priamursky State University
Student

Abstract
At the moment, a new applied area of mathematics, specializing in artificial neural networks, is rapidly developing in the world. The relevance of research in this direction is due to the fact that the use of neural networks is possible in all spheres of life. Recognition of text from photos, video, audio and translation of handwritten texts, digitization from paper media and far from all the possibilities of using neural networks to make human life easier. This article discusses the implementation of an interface for image recognition using the brain.js library in the JavaScript programming language.
Keywords: neural network, recognition, JavaScript, brain.js.

1. Введение

1.1 Актуальность исследования

В настоящий момент в мире бурно развивается новая прикладная область математики, специализирующаяся на искусственных нейронных сетях. Актуальность исследования в этом направлении обусловлена тем, что применение нейронных сетей возможно во всех сферах жизни.

Распознавание текста с фотографий, видео, аудио и перевод рукописных текстов, оцифровка с бумажных носителей и это далеко не все возможности применения нейронных сетей для облегчения жизни человека. Широта задач, решаемых нейронными сетями, объясняется во многом тем, что возможность обучения позволяет сделать функционирование системы на порядок более эффективным.

1.2 Обзор исследований

В научном исследовании Р. Н. Федоров и Д. В. Ильин разработали программу в среде Borland C++ Builder для распознавания рукописных букв русского алфавита, представленных как растровое черно-белое изображение [1]. С. В. Маркова и К. Ю Жигалов рассматривают вопросы, затрагивающие тему применения нейронных сетей для построения систем идентификации изображений, имеющих широкое применение в системах обеспечения безопасности [2]. В статье В. Гаврищук описывается использовании нейронных сетей для решения задач распознавания изображений. Результатом практической работы является приложение, написанное на JavaScript, которое сочетает в себе нейронную сеть и визуализатор трехмерных моделей [3]. V. T. Nguyen, F. F. Pashchenko в своей работе предлагают архитектуру сверточной нейронной сети, которая создает нейросетевую систему для распознавания объектов на изображениях, используя собственный подход к классификации с применением иерархического классификатора [4]. Обзор литературы, целью которого является выявление современного уровня использования сверточных нейронных сетей в процессе идентификации и классификации болезней растений, определение тенденций и указание пробелов рассмотрен A. Abade и др. [5]. В статье Y. Li предложена нейронная сеть с возможностью передачи внимания для распознавания эмоций на электроэнцефалографе [6]. И. А. Зиганшин и Д. И. Валиуллина повествуют о наиболее актуальной библиотеке для создания и обучения нейронных сетей – PyTorch [7].

1.3 Цель исследования

Целью данного исследования является написание интерфейса по распознаванию изображений с помощью библиотеки brain.js на языке программирования JavaScript.

2. Методы исследования

Для реализации интерфейса были выбраны следующие средства:

- JavaScript - мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript [8].
- Brain.js — Javascript библиотека для искусственных нейронных

сетей, заменяющая «мозговую» библиотеку, которая предлагает разные типы сетей в зависимости от конкретных задач [9].

- PhpStorm - это интегрированная среда разработки на PHP с интеллектуальным редактором, которая глубоко понимает код, поддерживает PHP 5.3-7.3 для современных и классических проектов, обеспечивает лучшее в индустрии автодополнение кода, рефакторинги, предотвращение ошибок налету и поддерживает смешивание языков [10].

3. Результаты исследования

Изначально создаем новый проект в PHPStorm, который будет состоять из двух файлов: index.html, описывающий интерфейс общения с нейронной сетью, и файл библиотеки brain.js, который необходимо скачать с github репозитория (Рис. 1).

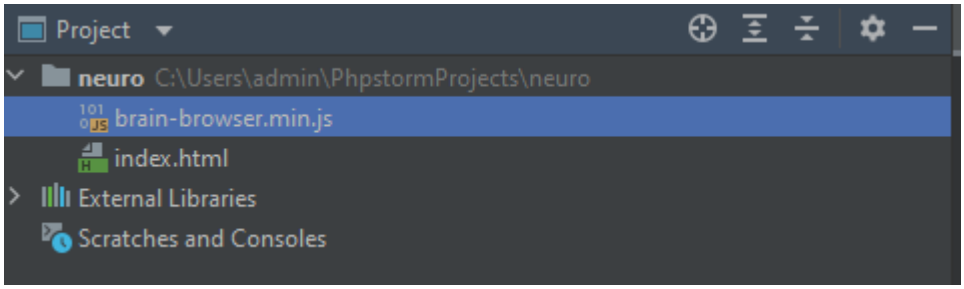

Рисунок 1 – Структура проекта

Открываем файл index.html, прописываем стандартную структуру, подключаем библиотеку brain.js и описываем стили (Рис. 2).

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>НС</title>

    <script src="brain-browser.min.js"></script>

    <style>
        body {
            background-color: #333;
        }

        #canv {
            position: absolute;
            top: 0;right: 0;bottom: 0;left: 0;
            margin: auto;
            background-color: white;
        }
    </style>
</head>
<body>

<canvas id="canv" style="display: block;">Ваш браузер устарел, обновитесь.</canvas>
```

Рисунок 2 – Подключение библиотеки и описание стилей

Далее напишем функцию, описывающую параметры холста, на котором будем изображать рисунки, задав его высоту, ширину. А также опишем функции, с помощью которых в будущем разобьем холст на ячейки (Рис. 3).

```
<script>
    function DCanvas(el)
    {
        const ctx = el.getContext('2d');
        const pixel = 20;

        let is_mouse_down = false;

        canv.width = 500;
        canv.height = 500;

        this.drawLine = function(x1, y1, x2, y2, color = 'gray') {
            ctx.beginPath();
            ctx.strokeStyle = color;
            ctx.lineJoin = 'miter';
            ctx.lineWidth = 1;
            ctx.moveTo(x1, y1);
            ctx.lineTo(x2, y2);
            ctx.stroke();
        }

        this.drawCell = function(x, y, w, h) {
            ctx.fillStyle = 'blue';
            ctx.strokeStyle = 'blue';
            ctx.lineJoin = 'miter';
            ctx.lineWidth = 1;
            ctx.rect(x, y, w, h);
            ctx.fill();
        }
```

Рисунок 3 – Отрисовка холста интерфейса

Теперь необходимо описать функции очистки холста и отрисовки ячеек с помощью ранее написанных функций (Рис. 4).

```
        this.clear = function() {
            ctx.clearRect(0, 0, canv.width, canv.height);
        }

        this.drawGrid = function() {
            const w = canv.width;
            const h = canv.height;
            const p = w / pixel;

            const xStep = w / p;
            const yStep = h / p;

            for( let x = 0; x < w; x += xStep )
            {
                this.drawLine(x, 0, x, h);
            }

            for( let y = 0; y < h; y += yStep )
            {
                this.drawLine(0, y, w, y);
            }
        }
```

Рисунок 4 – Функции очистки холста и отрисовки ячеек

Затем напишем обработчики событий движения компьютерной мыши, что позволит рисовать на холсте (Рис. 5).

```
    el.addEventListener('mousedown', function(e) {
        is_mouse_down = true;
        ctx.beginPath();
    })

    el.addEventListener('mouseup', function(e) {
        is_mouse_down = false;
    })

    el.addEventListener('mousemove', function(e) {
        if( is_mouse_down )
        {
            ctx.fillStyle = 'red';
            ctx.strokeStyle = 'red';
            ctx.lineWidth = pixel;

            ctx.lineTo(e.offsetX, e.offsetY);
            ctx.stroke();

            ctx.beginPath();
            ctx.arc(e.offsetX, e.offsetY, pixel / 2, 0, Math.PI * 2);
            ctx.fill();

            ctx.beginPath();
            ctx.moveTo(e.offsetX, e.offsetY);
        }
    })
}
```

Рисунок 5 – Написание обработчиков движения компьютерной мыши

Так как нейронная сеть не понимает рисунков, то необходимо создать функцию, которая будет переводить изображения в двоичных код, где закрашенные ячейки будут обозначать 1, а не закрашенные 0. Объявляем необходимые константы и применяем ранее созданные функции и drawGrid drawCell для закрашивания ячеек, которые выделил мышкой пользователь при рисовании, а также с помощью цикла переводим сетку ячеек в двоичный код (Рис. 6).

```
this.calculate = function(draw = false) {
    const w = canv.width;
    const h = canv.height;
    const p = w / pixel;
    const xStep = w / p;
    const yStep = h / p;
    const vector = [];
    let __draw = [];
    for( let x = 0; x < w; x += xStep )
    {
        for( let y = 0; y < h; y += yStep )
        {
            const data = ctx.getImageData(x, y, xStep, yStep);
            let nonEmptyPixelsCount = 0;
            for( i = 0; i < data.data.length; i += 10 )
            {
                const isEmpty = data.data[i] === 0;
                if( !isEmpty )
                {
                    nonEmptyPixelsCount += 1;
                }
            }
            if( nonEmptyPixelsCount > 1 && draw )
            {
                __draw.push([x, y, xStep, yStep]);
            }
            vector.push(nonEmptyPixelsCount > 1 ? 1 : 0);
        }
    }
    if( draw )
    {
        this.clear();
        this.drawGrid();
        for( _d in __draw )
        {
            this.drawCell( __draw[_d][0], __draw[_d][1], __draw[_d][2], __draw[_d][3] );
        }
    }
    return vector;
}
```

Рисунок 6 – Функция calculate

Присвоим кнопке «c» функцию очистки холста. При нажатии кнопки «v» будем спрашивать пользователя о том какой смысл несет нарисованное изображение (положительный или отрицательный) и результат будем передавать в функцию перевода изображения в двоичный код (calculate). Кнопка «b» будет отвечать за передачу массива обучающих данных от функции calculate библиотеке brain.js (Рис. 7).

```
    let vector = [];
    let net = null;
    let train_data = [];
    const d = new DCanvas(document.getElementById('canv'));
    document.addEventListener('keypress', function(e) {
        if( e.key.toLowerCase() == 'c' )
        {
            d.clear();
        }
        if( e.key.toLowerCase() == 'v' )
        {
            vector = d.calculate(true);
            if( confirm('Результат положительный?') )
            {
                train_data.push({
                    input: vector,
                    output: {positive: 1}
                });
            } else
            {
                train_data.push({
                    input: vector,
                    output: {negative: 1}
                });
            }
        }
        if( e.key.toLowerCase() == 'b' )
        {
            net = new brain.NeuralNetwork();
            net.train(train_data, {log: true});

            const result = brain.likely(d.calculate(), net);
            alert(result);
        }
    });
</script>
</body>
</html>
```

Рисунок 7 – Описании событий нажатия на клавиши

Проверим работу нейронной сети, обучив ее для начала на входных данных. Нарисуем крестик на холсте, нажмем клавишу «v» и, отмечая данное изображение кнопкой «OK», дадим понять нейронной сети, что данное изображение несет положительный смысл. Повторим данное действие 5-6 раз (Рис. 8).

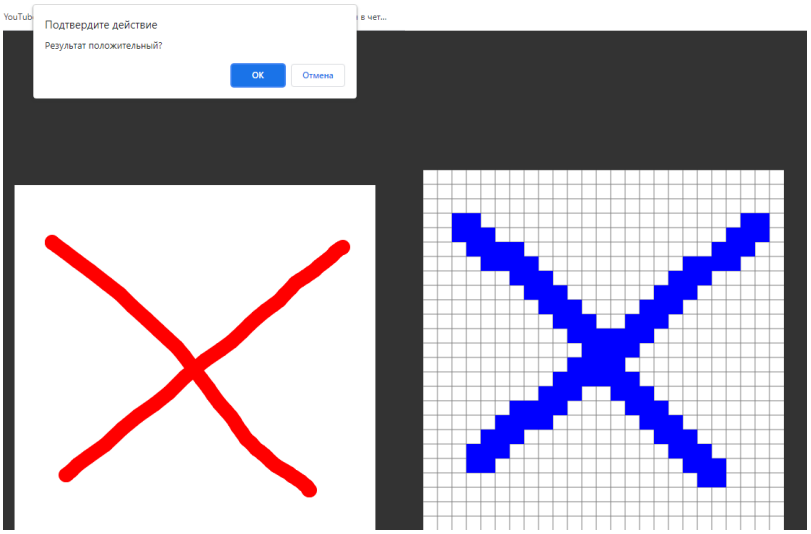

Рисунок 8 – Обучение сети распознаванию крестика

Проделаем те же действия, рисуя на холсте галочку и давая понять нейронной сети, что изображение несет отрицательный смысл кнопкой «Отмена» (Рис. 9).

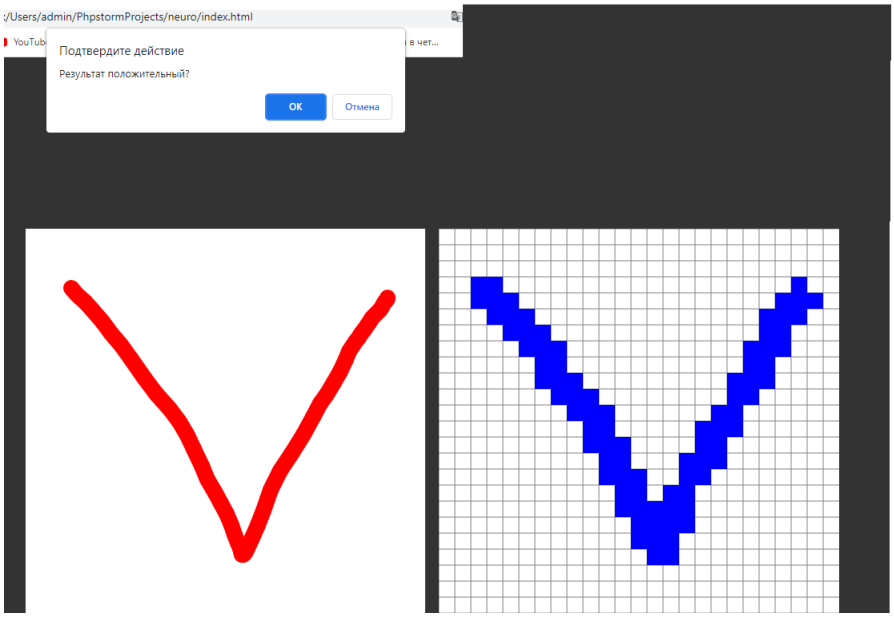

Рисунок 9 – Обучение сети распознаванию галочки

Теперь нарисуем на выбор либо крестик, либо галочку и нажмем клавишу «b» для того, чтобы обученная нейронная сеть распознала какое изображение нарисовал пользователь (крестик - positive, галочка - negative) (Рис. 10-11).

Рисунок 10 – Распознанный нейронной сетью крестик

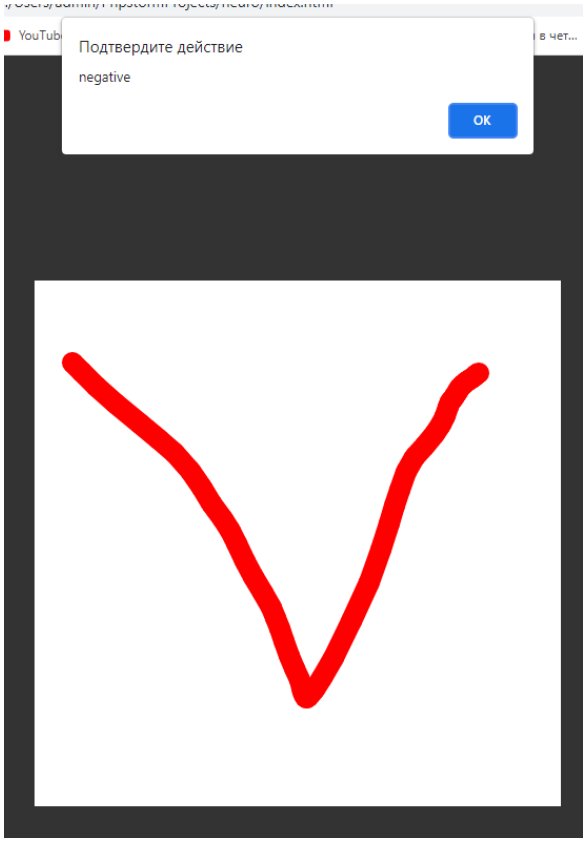

Рисунок 11 – Распознанная нейронной сетью галочка

4. Выводы

Нейронные сети быстрыми темпами вливаются в жизнь современного человека и захватывают всевозможные сферы деятельности людей. Распознавание речи и изображений, решение различных задач, прогнозирование всевозможных событий – все это лишь малая часть примеров применения нейронных сетей. В данной статье написан интерфейс взаимодействия с библиотекой brain.js для обучения нейронной сети и распознавания с ее помощью различных изображений.

Библиографический список

1. Федоров Р. Н., Ильин Д. В. Распознавание букв с помощью нейронной сети // Информатика и вычислительная техника. 2015. С. 107-108.
2. Маркова С. В., Жигалов К. Ю. Применение нейронной сети для создания системы распознавания изображений // Фундаментальные исследования. 2017. №. 8-1. С. 60-64.
3. Гаврищук В. Применение нейронных сетей в компьютерной графике // Analele ştiinţifice ale USM. Științe ale naturii și exacte. Științe umanistice. 2019. С. 52-55.
4. Nguyen V. T., Pashchenko F. F. Development of an object recognition algorithm based on neural networks With using a hierarchical classifier // Procedia Computer Science. 2021. Т. 184. С. 438-444.
5. Abade A., Ferreira P. A., de Barros Vidal F. Plant diseases recognition on

images using convolutional neural networks: A systematic review // Computers and Electronics in Agriculture. 2021. Т. 185. С. 106125.
6. Li Y. et al. A novel transferability attention neural network model for EEG emotion recognition // Neurocomputing. 2021. Т. 447. С. 92-101.
7. Зиганшин И. А., Валиуллина Д. И. Применение библиотеки Pytorch для обучения нейронных сетей // Лучшие научные исследования 2021. 2021. С. 17-19.
8. JavaScript URL: https://ru.wikipedia.org/wiki/JavaScript (дата обращения: 04.09.2021).
9. BrainJS URL https://github.com/BrainJS (дата обращения: 04.09.2021).
10. PHPStorm URL: https://www.jetbrains.com/ru-ru/phpstorm/ (дата обращения: 04.09.2021).