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 информационных технологий

КУРСОВАЯ РАБОТА

РАСПОЗНАВАНИЕ НА ИЗОБРАЖЕНИИ ЯДОВИТЫХ РАСТЕНИЙ

Работу выполнил _____ В. А. Ардышев
(подпись)
Направление подготовки 01.04.02 «Прикладная математика и информатика»

Направленность (профиль) «Технологии программирования и разработки информационно-коммуникационных систем»

Научный руководитель
канд. физ.-мат. наук, доц._____ Е. П. Лукащик
(подпись)
Нормоконтролер
ст. преп. _____ А. В. Харченко
(подпись)

Краснодар
2020

РЕФЕРАТ

Курсовая работа 41 с., 4ч., 18 рис., 10 источников.

НЕЙРОСЕТЬ, НЕЙРОННАЯ СЕТЬ, СВЕРТОЧНАЯ НЕЙРОННАЯ СЕТЬ, ОБРАЗ, КЛАССИФИКАЦИЯ, ОБУЧЕНИЕ

Объектом исследования данной работы является модель искусственной нейронной сети со свёрточной архитектурой для распознавания сложных визуальных образов.

Предмет исследования – способность к качественному распознаванию сложных визуальных образов растительности нейронными сетями.

Цель работы – разработка мобильного приложения для распознавания ядовитых растений, включающая разработку модели сверточной сети и ее обучение.

В результате проведенной работы было разработано мобильное приложение «Ядовитые растения Кавказа», которое работает без доступа к сети интернет, определяя на изображении с камеры устройства ядовитые растения. Также приложение предоставляет справочную информацию об определенном ядовитом растении.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день набирают популярность системы искусственного интеллекта, машинное обучение и глубокое обучение. Такие интеллектуальные системы находят свое применение как в различных областях науки, таких как экономика, медицина, биология и даже астрономия, так и в повседневной жизни человека.

В работе представлено мобильное приложение, позволяющее по изображению с камеры распознавать ядовитые растения Кавказа. Для работы приложения используется обученная свёрточная нейронная сеть, наилучшим образом подходящая для задач распознавания визуальных образов.

Целью данной работы является разработка мобильного приложения для распознавания ядовитых растений, включающая разработку модели сверточной сети и ее обучение.

Объектом данного исследования является модель искусственной нейронной сети со свёрточной архитектурой для распознавания сложных визуальных образов.

Предмет исследования – способность к качественному распознаванию сложных визуальных образов растительности нейронными сетями.

Перед исследованием стоят следующие задачи:

1) изучение устройства и сфер применения некоторых распространенных архитектур нейронных сетей;

2) выбор подходящей архитектуры нейронной сети для распознавания визуальных образов на изображениях;

3) изучение способов обучения нейронных сетей;

4) исследование особенностей сверточных нейронных сетей;

5) проектирование и обучение собственной модели;

6) анализ результатов обучения и работы полученной модели.

Нейронные сети часто используются для распознавания образов на изображениях, однако чаще всего такие сети обучаются на подготовленных

данных, где на изображениях обучающей выборки четко и крупно запечатлен объект. Актуальность данной работы состоит в том, что она демонстрирует возможности обучения сети даже на изображениях с разным качеством, на которых объект представляет собой сложный визуальный образ и может плохо выделяться на фоне окружения.

Результаты исследования могут быть применены в туристической сфере, для предотвращения отравлений, ожогов и прочих последствий от контакта с ядовитыми растениями.

1 История развития искусственных нейронных сетей

Для того чтобы проследить как человек пришел к идее создания нейронных сетей нужно рассматривать ее в контексте, связанном с попытками создать искусственный интеллект (ИИ). Здесь можно выделить две основных линии развития идей ИИ [1]. Первая линия восходит к Аристотелю и Евклиду. Система рассуждений в рамках евклидовой геометрии строится на системе постулатов и правил логического вывода. На наборе постулатов и правил доказываются утверждения, то есть на основе существующих знаний извлекаются новые. Именно такой подход моделирования мышления использовался при первых попытках реализации систем искусственного интеллекта. Данный подход предполагал, что для любой области человеческой деятельности можно найти достаточно полную и верную систему аксиом и, используя правила логического вывода, можно было бы использовать ИИ для получения ответов на любые вопросы из выбранной области.

Бурное развитие такого подхода происходило с момента возникновения современных компьютеров и породило так называемые экспертные системы. Экспертная система – это совокупность методов и средств организации накопления и применения знаний для решения сложных задач в некоторой предметной области [2]. Однако практика показала, что эффективность систем подобного рода сильно зависит от полноты знаний о предметной области, извлеченных из специалистов, а так же непротиворечивости полученной системы знаний. На сегодняшний момент существуют лишь переборные алгоритмы подбора удачной системы аксиом для таких систем, да и логический вывод новых знаний в системах с известными аксиомами также является очень трудоемкой задаче с точки зрения вычислений.

В связи с этим, параллельно с подходом к реализации искусственного интеллекта через воспроизведение рассуждений человека, развивалось альтернативное направление, которое было связано с развитием идей

машинного обучения. Задачей машинного обучения является автоматическое извлечение некоторого обобщенного знания из обучающего набора (набора примеров). Такие обобщенные знания могут быть использованы для решения новых задач за пределами тех данных, на которых происходило обучение.

В то время как направление экспертных систем восходит к идеям Евклида, в основе систем машинного обучения лежит идея «координатизации» окружающего мира Декарта. Благодаря этой идее мы можем представлять любые окружающие нас объекты как точки многомерного пространства параметров (признаков). Машинное обучение использует ограниченное представление (выбирается только часть важных для исследования параметров) объекта в виде числовых векторов его параметров и работает с входными данными путем математических преобразований, которые приводят к результату. Реализуя идею машинного обучения, компьютер не моделирует процесс мыслительных процессов человека, он минимизирует ошибку на обучающих данных, корректируя внутренние параметры алгоритма. Такой ИИ обучается решать задачи и выполнять разнообразные функции человека, пропуская этап выяснения того, каким способом решает такую задачу человек. Эта идея и является основополагающей в теории машинного обучения.

Искусственные нейронные сети (ИНС) входят во второе направление искусственного интеллекта. ИНС развивались как попытка моделирования естественных нейронных сетей, из которых состоит нервная система человека и животных, откуда они и получили свое название.

Естественный (биологический) нейрон – сложный биологический объект, устройство и функционирование которого все еще продолжают изучать. Однако общее представление о нейронах человечество получило довольно давно. Структура биологического нейрона представлена на рисунке Рисунок 1 .

Рисунок 1 – Биологический нейрон

Естественный нейрон имеет ядро и отростки нервных волокон двух типов – дендриты, по которым принимаются импульсы, и единственный аксон, по которому нейрон может передавать импульс. Контакт аксона с дендритами других нейронов осуществляется через специальные образования – синапсы, которые влияют на силу передаваемого импульса. Структура, построенная из совокупности большого количества таких нейронов, получила название биологической (или естественной) нейронной сети.

1.1 Модель МакКалока-Питтса

В 1943 году выдающиеся ученые Уорен Маккалок и Уолтер Питтс формализовали понятие нейронной сети, предложив дискретную модель искусственного нейрона, которая легла в основу теории логических сетей и конечных автоматов [4].

Ученые строили модель исходя из предположения бинарности признаков (сигналов) поступающих в искусственный нейрон, которым придается значение величины импульсов, поступающих на вход биологического нейрона через n входных синапсов. Для регулирования силы сигнала подобно тому, как это происходит при прохождении импульса через синапс естественного нейрона, были выбраны веса ω_i, которые умножаются соответственно на входные признаки x^i, формируя при этом значения

сигнала, поступающего в тело искусственного нейрона. При этом считалось, что если значение веса ω_i положительное, то соответствующий синапс возбуждающий, а если отрицательный – тормозящий. В теле нейрона анализируются величины импульсов, а именно проверяется, что суммарный импульс превышает заданный фиксированный порог активации ω_0. Если это так, то нейрон возбуждается и на выходе выдается импульс равный 1, иначе 0. Модель искусственного нейрона МакКалока-Питтса представлена на рисунке Рисунок 2 .

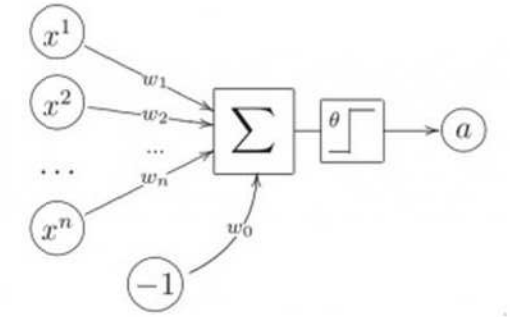

Рисунок 2 – Модель нейрона МакКалока-Питтса

В данной схеме искусственный нейрон вычисляет n-арную булеву функцию вида (1).

$$a(x) = \varphi\left(\sum_{i=1}^{n} \omega_i x^i - \omega_0\right), \text{где } \varphi(z) = [z \geq 0] \qquad (1)$$

Здесь φ(z) – ступенчатая функция Хевисайда, которая принимает значение 1, когда z неотрицательное и 0 в противном случае. В современной теории нейронных сетей функция φ, преобразующая значение суммарного импульса в выходное значение нейрона, получила название активационной функции нейрона. Вообще говоря, модель МакКалока-Питтса может рассматриваться как пороговый линейный классификатор.

В ходе своей работы в начале 40-х годов Маккалок и Питтс заложили основы нейроматематики и сформулировали основные положения теории деятельности головного мозга. Помимо разработанной модели искусственного нейрона к достижениям ученых относится разработанная конструкция сети из таких элементов для выполнения некоторых арифметических и логических операций, а также сделанное важное предположение о том, что подобные сети способны обучаться и обобщать полученную информацию.

Минусом данной модели является использование пороговой функции, которая не предоставляет модели достаточную гибкость в обучении сети для конкретных задач. В предложенной схеме, даже если взвешенной сумме входных импульсов совсем немного не удается достичь величины порога, то сигнал на выходе нейрона сразу пропадает, что приводит к затуханию импульса на входах нейронов следующего слоя, связанных с данным нейроном. К тому же модель не учитывает многих особенностей реальных нейронов, например нелинейности суммирования входной информации.

1.2 Персептрон

Американский ученый Фрэнк Розенблатт стал человеком, воплотившим в 1958 году идею МакКалока-Питтса. в жизнь. Для IBM-794 было создано устройство МАРК-1, а также соответствующая ему математическая модель мозга [5]. Здесь под «моделью мозга» понимается любая теоретическая система, объясняющая физиологические функции мозга с помощью известных законов физики и математики, опираясь на известные факты нейрофизиологии и нейроанатомии. Эта математическая модель получила название персептрона, она представляет собой сеть, которая состоит из так называемых генераторов сигнала трех типов: сенсорных, ассоциативных и реагирующих элементов. В этих элементах содержатся функции, производящие выходной сигнал элемента. Эти функции зависят от

набора сигналов, поступающих на входы элемента от других элементов сети, либо же из внешней среды. Обычно, под персептроном Розенблатта понимают частный случай описанной сети – элементарный персептрон, упрощенный по набору параметров, схема которого представлена на рисунке Рисунок 3 .

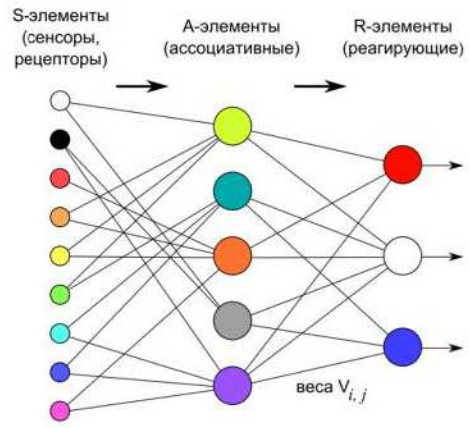

Рисунок 3 – Элементарный персептрон Розенблатта

Первый тип элементов персептрона – элементы типа S, или S-элементы. Набор S-элементов образует слой рецепторов, получающих сигналы из окружающей среды. Эти рецепторы имеют связи с A-элементами, которые могут быть тормозными либо возбуждающими. Каждый рецептор в фиксированный момент времени находится в одном из двух состояний – покоя или возбуждения.

A-элементы представляют собой искусственный (формальный) нейрон, или как еще говорят сумматор с порогом Θ. Так же A-элементы называют ассоциативными элементами персептрона, так как обычно одному A-элементу соответствует целый набор S-элементов. Таким образом, A-элемент переходит в состояние возбуждения в том случае, когда алгебраическая сумма сигналов, приходящая от S-элементов превосходит некоторый фиксированный порог. Сигналы, приходящие в A-элемент по возбуждающей связи считаются положительными, а по тормозной связи – отрицательными.

Сигналы от возбужденных A-элементов поступают в R-элементы, при чем сигнал от i-го ассоциативного элемента передается с коэффициентом – весом связи между A-элементом и R-элементом. R-элементы аналогично элементам типа подсчитывают взвешенную сумму входных сигналов и сравнивают ее с величиной порога Θ, выдавая «1», в случае, если вычисленная сумма превзошла значение порога и «0» иначе. Математически такую операцию можно записать в виде (2).

$$f(x) = sign\left(\sum_{i=1}^{n} \omega_i x_i - \Theta\right) \quad (2)$$

Классическим методом обучения персептрона считается обучение с коррекцией ошибки, который заключается в том, чтобы не изменять вес связи до тех пор, пока текущая «настройка» персептрона остается правильной. Если персептрон дает неправильную реакцию то вес связи изменяется на единицу со знаком, противоположным знаку ошибки.

В 1949 появляется книга «Организация сознания» написанная физиологом Дональдом Хеббом. В этой книге объясняется, как обучаются нейроны человеческого мозга. Основным смыслом идеи обучения Хебба является то, что если изначально наблюдается причинно-следственная связь между активациями пре- и постсинаптического нейрона, то эта связь обладает тенденцией к усилению, причем усиление связанности может происходить не только за счет изменения синапса, но и за счет изменения метаболических особенностей самих клеток. Из этого суждения вытекло два основных правила обучения Хэбба, которые можно сформировать следующим образом:

– если два нейрона по обе стороны соединения активизируются синхронно, то вес (значимость) этого соединения имеет склонность к увеличению;

– если два нейрона по обе стороны соединения активизируются асинхронно, то вес такой связи склонен к уменьшению, а значение связи очень может становиться крайне малым.

Розенблатт занимался классификацией различных алгоритмов обучения и называл их системами подкрепления. При описании разных видов алгоритмов обучения Розенблатт опирался на идеи Хэбба. Ученым были выделены следующие основные методы коррекции ошибок персептрона [6].

Метод коррекции ошибок без квантования предполагает вводить подкрепление, когда реакция персептрона на стимул S_i неправильная, при этом вес каждого активного А-элемента увеличивается на величину (3).

$$\eta = \rho_i \Delta x_i \tag{3}$$

Здесь Δx_i число единиц подкрепления, которое выбирается так, чтобы величина сигнала превысила порог Θ, а ρ_i определяется по формуле (4).

$$\rho_i = \begin{cases} +1, \text{если } S = S_i^+ \\ -1, \text{если } S = S_i^- \end{cases} \tag{4}$$

При этом S_i^+ и S_i^- – стимулы, принадлежащие положительному и отрицательному классу.

Метод коррекции ошибки с квантование отличается от данного, только фиксированной величиной $\Delta x_i = 1$.

Метод коррекции со случайным знаком подкрепления отличается тем, что знак подкрепления η в этом случае подбирается случайным образом и не зависит от реакции персептрона, но подкрепление по-прежнему используется только в случае, когда персептрон неверно реагирует на стимул.

Метод коррекции ошибок со случайными возмущениями предполагает, что величина и знак η выбирается для каждой связи отдельно и независимо с некоторым распределением вероятностей.

2 Метод обратного распространения ошибки

Для обучения многослойных персептронов невозможно использовать правило, предложенное Розенблаттом, так как для подсчета величины изменения весового коэффициента требуется знать верные выходные значения, которые известны только для выходного слоя сети.

Поэтому для обучения многослойного персептрона рядом ученых был предложен метод, основанный на градиенте функции ошибки по весовым коэффициентам модели. Такой метод предполагал распространение ошибки от выходов сети против направления естественного движения сигнала при ее обучении, вплоть до входных нейронов, благодаря чему данный метод получил название метода обратного распространения ошибки (error back propagation). На сегодняшний день это самый популярный метод обучения многослойных персептронов, позволяющий обучить все слои многослойной нейронной сети. Метод предполагает дифференцируемость функции активации.

Алгоритм обратного распространения ошибки – итерационный градиентный алгоритм обучения, который используется с целью минимизации функции ошибки, или как ее еще называют функции потерь. Согласно методу наименьших квадратов, минимизируемую целевую функцию ошибки нейронной сети можно представить в виде (5).

$$E(w) = \frac{1}{2}\sum_{j,k}(y_{j,k}^{(Q)} - d_{j,k})^2 \qquad (5)$$

Здесь $y_{j,k}^{(Q)}$ – реальное выходное состояние j-го нейрона выходного слоя сети, при передаче на ее входы k-го образа, а $d_{j,k}$ – требуемое выходное состояние этого нейрона. Суммирование происходит по всем нейронам выходного слоя и всем обрабатываемым сетью образам. Минимизация

методом градиентного спуска обеспечивает подстройку весовых коэффициентов, изменяя соответствующие веса системы на величину, вычисляемую по формуле (6).

$$\Delta w_{i,j}^{(q)} = -\eta \frac{\partial \mathrm{E}}{\partial w_{i,j}}, \tag{6}$$

где $w_{i,j}$ – весовой коэффициент связи, ссоединяющей i-ый нейрон слоя (q - 1) с j-ым нейроном слоя q, η –коэффициент скорости обучения из интервала (0,1). По правилу дифференцирования сложной функции частную производную функции потерь по весовому коэффициенту системы можно выразить формулой (7).

$$\frac{\partial \mathrm{E}}{\partial w_{i,j}} = \frac{\partial \mathrm{E}}{\partial y_i} \frac{\mathrm{d}y_j}{\mathrm{d}s_j} \frac{\partial s_j}{\partial w_{i,j}}, \tag{7}$$

где s_i – взвешенная сумма всех сигналов нейрона j, т.е. аргумент функции активации. Так как производная активационной функции должна быть определена на всей оси абсцисс, то функции с разного рода неоднородностями не подходят для рассматриваемых нейронных сетей. В них применяют такие гладки функции, как гиперболический тангенс или классический сигмоид с экспонентой и другие. Заметим, что в формуле (7) последний множитель принимает значение, равное выходу нейрона предыдущего слоя $y_j^{(q-1)}$. Первый множитель из (7) легко раскладывается в вид (8):

$$\frac{\partial \mathrm{E}}{\partial y_j} = \sum_r \frac{\partial \mathrm{E}}{\partial y_r} \frac{\mathrm{d}y_r}{\mathrm{d}s_r} \frac{\partial s_r}{\partial y_j} = \sum_r \frac{\partial \mathrm{E}}{\partial y_r} \frac{\mathrm{d}y_r}{\mathrm{d}s_r} w_{j,r}^{(q+1)} \tag{8}$$

Здесь суммирование по r выполняется среди нейронов слоя (q + 1). Далее введем новую переменную

$$\delta_j^{(q)} = \frac{\partial \mathrm{E}}{\partial y_j} \frac{\mathrm{d}y_j}{\mathrm{d}s_j} \tag{9}$$

и получим рекуррентную формулу для расчетов величин $\delta_j^{(q)}$ слоя q из величин $\delta_j^{(q+1)}$ старшего слоя (q + 1):

$$\delta_j^{(q)} = \left[\sum_r \delta_j^{(q+1)} w_{j,r}^{(q+1)} \right] \frac{\mathrm{d}y_j}{\mathrm{d}s_j}, \tag{10}$$

при чем, для выходного слоя имеем формулу (11)

$$\delta_j^{(Q)} = (y_j^{(Q)} - d_j) \frac{\mathrm{d}y_j}{\mathrm{d}s_j}, \tag{11}$$

После этого можно записать в раскрытом виде формулу (6):

$$\Delta w_{i,j}^{(q)} = -\eta \delta_j^{(q)} y_j^{(q-1)}. \tag{12}$$

Для придания коррекции весов некоторой инерционности, сглаживающей резкие скачки при перемещении по поверхности целевой функции, вид (12) можно дополнить значением изменения веса на предыдущей итерации:

$$\Delta w_{i,j}^{(q)}(t) = -\eta \left(\mu \Delta w_{i,j}^{(q)}(t-1) + (1-\mu) \delta_j^{(q)} y_j^{(q-1)} \right). \tag{13}$$

В последней формуле μ – коэффициент инерционности, t – номер итерации.

Полный алгоритм обучения нейронной сети с помощью метода обратного распространения ошибки выглядит следующим образом.

Шаг 1: на входы сети подается один из возможных образов и в режиме прямого распространения сигнала от входов к выходам, рассчитываются значения выходов.

Шаг 2: По формуле (11) рассчитать значение $\delta^{(Q)}$. По формулам (12) и (13) рассчитать изменения весов $\Delta w^{(Q)}$ выходного слоя Q.

Шаг 3: По формулам (10) и (12) (или (10) и (13)) вычислить соответственно $\delta^{(q)}$ и $\Delta w^{(q)}$ для всех остальных слоев q = (Q-1)..1.

Шаг 4: Скорректировать все веса нейронной сети, прибавлением вычисленных приращений весов.

Шаг 5: Если величина ошибки все еще существенна, то перейти на шаг 1, в противном случае алгоритм заканчивает свою работу.

На первом шаге сети попеременно в случайном порядке предъявляются все тренировочные образы, чтобы сеть обучалась, корректно, а не «запоминала» лишь один конкретный пример.

Из выражения (12) следует, что, когда выходное значение $y_j^{(q-1)}$ стремится к нулю, эффективность обучения сильно снижается. Таким образом при двоичных входных векторах половина весовых коэффициентов не будет корректироваться, поэтому область возможных значений выходов (0, 1) можно сдвинуть в пределы (-0,5, 0,5), что достигается простыми модификациями активационных функций.

3 Свёрточные нейронные сети

Свёрточная нейронная сеть (CHH) – специальная архитектура искусственных нейронных сетей, которая была предложена Яном Лекуном в 1988 году и предназначенная для эффективного распознавания образов.

Благодаря архитектурным особенностям такой сети удается гораздо точнее распознавать объекты на изображениях, так как, в отличие от многослойного персептрона, эта сеть учитывает расположение пикселов изображения в двухмерном пространстве. Многомерные персептроны принимают входную информацию в виде векторов. Если вытянуть входное изображение в длинный одномерный массив для его передачи в многослойный персептрон, то будет утрачена информация о близости пикселов в исходном изображении, что не позволяет таким сетям качественно анализировать изображения.

Помимо возможности достигать высокой точности распознавания образов на изображениях, в свёрточных нейронных сеть сильно уменьшается количество настраиваемых в процессе обучения весов. Изображения в компьютере представляется в виде матрицы пикселов, каждый пиксель характеризуется интенсивностями каналов изображения. Например, чёрно-белое изображение имеет один канал, а изображение в формате RGB – три канала. На рисунке Рисунок 4 представлено представление одноканального черно-белого изображения в памяти компьютера. В случае цветного изображения его компьютерное представление будет состоять из трех аналогичных представлений – по одному на каждый канал.

Изображение размером M x N в кодировке RGB характеризуется набором из M x N x 3 чисел. Если передавать такое изображение на вход полносвязной нейронной сети, то для каждого нейрона первого слоя потребуется определить M x N x 3 параметра (весов). Помимо входного слоя сети имеется набор скрытых слоев и выходной слой, количество нейронов в которых может превышать число нейронов на первом слое. Такое большое количество

параметров сильно замедляет процесс обучения нейронной сети и даже может делать его невозможным за приемлемое время ввиду ограниченности доступных вычислительных мощностей. Количество параметров свёрточной сети намного меньше за счет особенностей работы ее слоев, о чем речь пойдет позже.

Рисунок 4 – Изображение рукописной цифры и ее представление в памяти компьютера

Архитектура сверточной нейронной сети основана на идеи последовательного отбора признаков на изображении – от примитивных признаков, таких как разнонаправленные контуры и текстуры, до более абстрактных представлений, собираемых из наборов более примитивных. Похожим образом работает человеческий мозг при визуальной идентификации объектов. Таким образом, нейронная сеть в общем случае представляет собой набор чередующихся пар слоев, в которых первый слой отвечает за выделение в изображении некоторых признаков, а второй – за выделение среди этих признаков наиболее важных и выраженных.

Структура сети однонаправленная, то есть отсутствуют обратные связи от более глубоких слоев к менее глубоким. При обучении свёрточных нейронных сетей используются те же методы, что и при обучении многих других сетей, чаще всего используется одна из реализаций метода обратного

распространения ошибки. Функция активации выбирается для таких сетей экспериментально.

Как показала практика использования свёрточных нейронных сетей, они довольно устойчивы к небольшим смещениям изображения, изменениям масштаба и поворотам объектов на входных изображениях. На сегодняшний день архитектуры, основанные на свёрточных сетях, являются лучшим решением для задач распознавания образов.

3.1 Операция свёртки

Двумерная свертка является довольно простой операцией. Она характеризуется своим ядром – двумерной матрицей весовых коэффициентов. Операция выполняется над парой матриц A размера n_x x n_y и B размера m_x x m_y, в результате чего получается результирующая матрица C размера $(n_x - m_x + 1)$ x $(n_y - m_y + 1)$. Элементами результирующей матрицы C являются скалярные произведения подматриц размерами m_x x m_y матрицы A на матрицу B. Подматрица при этом определяется положением элемента в матрице C. Для расчета элементов матрицы C используют формулу (14).

$$C_{i,j} = \sum_{u=0}^{m_x-1} \sum_{v=0}^{m_y-1} A_{i+u,j+v} B_{u,v} \tag{14}$$

Другими словами, процесс свертки можно представить как перемещение матрицы B называемой ядром свертки по элементам матрицы A и нахождение суммы произведений элементов ядра на соответствующие им элементы подматрицы A, над которой в данный момент «нависает» ядро свертки. Этот процесс визуализирован на рисунке Рисунок 5 .

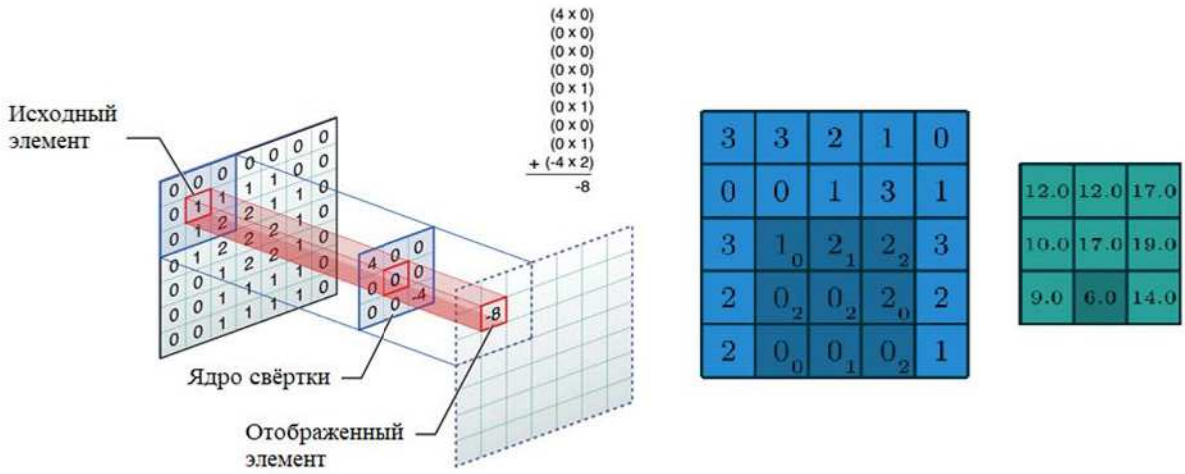

Рисунок 5 – Операция свертки

Логический смысл свертки заключается в том, что получаемый элемент тем больше, чем больше похожа (в смысле скалярного произведения) матрица B на подматрицу из матрицы A, на которую она наложена в данный момент.

Использование свертки позволяет заметно снизить количество параметров слоя сети. Например, имея 5x5 входных и 3x3 выходных характеристик, преобразование сверткой требует всего девяти параметров – элементов ядра. Если же реализовывать преобразование с помощью стандартного полносвязного слоя, то для этого потребуется определить весовую матрицу размера 25x9, то есть работать с 255 параметрами. Таким образом свёртывание является способом получения приближенной версии входных данных, что значительно сокращает количество требуемых расчетов.

Идея сверточной сети заключается в том, чтобы делать те же преобразования, что происходят в сетях прямого распространения, а именно находить суммы произведений входных сигналов на весовые коэффициенты для каждого нейрона слоя, затем применяя активационную функцию. Разница лишь в том, что сверточная сеть в качестве входных сигналов использует некоторый пучок сигналов, а не все возможные.

Пусть имеется входная матрица 4x4, которую требуется трансформировать в матрицу размерами 2x2. Если использовать для этой цели полносвязную нейронную сеть, то потребовалось бы вначале развернуть входную матрицу в вектор размерностью в 16 сигналов. Далее пропускать этот вектор через сеть с 16 выходными нейронами и 4 выходными. На рисунке Рисунок 6 изображена матрица W синаптических весов (весов связей) между нейронами входного и выходного слоев.

$$\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} & w_{1,4} & w_{1,5} & w_{1,6} & w_{1,7} & w_{1,8} & w_{1,9} & w_{1,10} & w_{1,11} & w_{1,12} & w_{1,13} & w_{1,14} & w_{1,15} & w_{1,16} \\ w_{2,1} & w_{2,2} & w_{2,3} & w_{2,4} & w_{2,5} & w_{2,6} & w_{2,7} & w_{2,8} & w_{2,9} & w_{2,10} & w_{2,11} & w_{2,12} & w_{2,13} & w_{2,14} & w_{2,15} & w_{2,16} \\ w_{3,1} & w_{3,2} & w_{3,3} & w_{3,4} & w_{3,5} & w_{3,6} & w_{3,7} & w_{3,8} & w_{3,9} & w_{3,10} & w_{3,11} & w_{3,12} & w_{3,13} & w_{3,14} & w_{3,15} & w_{3,16} \\ w_{4,1} & w_{4,2} & w_{4,3} & w_{4,4} & w_{4,5} & w_{4,6} & w_{4,7} & w_{4,8} & w_{4,9} & w_{4,10} & w_{4,11} & w_{4,12} & w_{4,13} & w_{4,14} & w_{4,15} & w_{4,16} \end{bmatrix}$$

Рисунок 6 – Матрица синаптических весов полносвязной сети с 16 входами, 4 выходами и отсутствием скрытых слоев

Если же использовать для описанной цели свёрточный слой с ядром размерности 3x3, то соответствующая матрица весов приняла бы вид, представленный на рисунке Рисунок 7 .

$$\begin{bmatrix} k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} & 0 \\ 0 & 0 & 0 & 0 & 0 & k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} \end{bmatrix}$$

Рисунок 7 – Псеведоматрица весов, сформированная перемещением ядра свертки

Таким образом, свёртка – линейное преобразование, хоть и отличающееся от привычных преобразований в «обычных» нейронных сетях. Для матрицы, состоящей из 64 элементов, мы определили всего 9 параметров, которые используются повторно при перемещении ядра свёртки над исходной матрицей. Выходное значение формируется, используя только те элементы, которые попадают в «поле видимости» ядра свертки в тот или

иной момент времени, а веса связей с другими элементами можно считать определенными и равными нулю. Можно сделать вывод, что полносвязные нейронные сети могут хорошо выполнять некоторые возложенные на них задачи, но у них есть проблемы с производительностью. При увеличении количества узлов сети количество связей растет очень быстро, что усложняет как прямой проход сигналов, так и обратное распространение ошибки при обучении сети. В этом плане сверточные сети на голову их опережают.

Размер свертки в сверточных слоях нейронной сети обычно выбирается равным 3x3 либо 5x5, так как больший размер может привести к потере некоторых значимых характеристик исходного изображения, что не позволит сети качественно распознавать некоторые классы изображений.

3.2 Другие операции

Можно обратить внимание, что при перемещении ядра свёртки его центр не может нависать над некоторыми граничными элементами исходной матрицы, в результате чего размер результата свертки меньше, чем исходный размер. В некоторых случаях это недопустимо и на выходе необходимо получать матрицу того же размера. Чтобы решить эту проблему применяют технику, называемую паддинг (англ. padding). Визуализация этой техники изображена на рисунке Рисунок 8 .

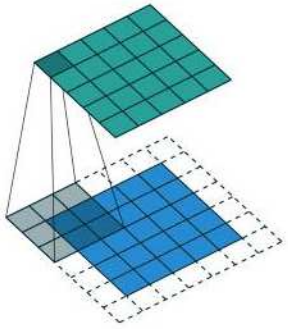

Рисунок 8 – Процесс применения свертки с паддингом

Ее идея заключается в том, чтобы умышленно увеличить исходную матрицу, дописав к ней дополнительные граничные строки и столбцы, заполнив их значениями, которые не повлияют на результат работы свертки, как правило, их заполняют нулями. Размер исходной матрицы увеличивается настолько, чтобы в результате свёртки получалось изображение того же размера, что и исходное.

Часто задачей свёрточного слоя сети является уменьшение размера результирующей матрицы. Это относится к основным идеям сверточных нейронных сетей – поступающие в систему импульсы редуцируются со скоростью, прямо пропорциональной количеству каналов, о которых речь пойдет позже. Для уменьшения размера результирующей матрицы можно использовать отдельный слой, называемый слоем субдискретизации (подвыборки) или слоем пулинга.

Пулинг – еще одна техника, основанная на скользящем окне, где вместо применения обучаемых весов используется статистическая функция некоторого вида по содержимому этого окна. Использование в сверточных нейронных сетях слоев пулинга дает следующие преимущества:

1) благодаря даунсемплингу происходит уменьшение количества параметров модели;

2) определение признаков на изображении становится более устойчивым к изменению ориентации или размера объекта.

Даунсемплингом в теории нейронных сетей называют сокращение размерности выходных данных после обработки слоем.

Чаще всего на сегодняшний момент в качестве статистической функции пулинга используют функцию максимума, возвращающую максимальное из значений, над которыми находится окно. Однако раньше в качестве пулинга довольно часто использовалась усредняющая подвыборка или даже L2-нормированная подвыборка, но как выяснилось на практике лучше работает максимизационная подвыборка [7]. В сверточных нейронных

сетях обычно используется пулинг со скользящим окном размера 2x2 и шагом 2. Пример работы такого пулинга представлен на рисунке Рисунок 9 .

Рисунок 9 – Принцип работы max-пулинга

Другим способом, позволяющим осуществлять даунсемплинг, является использование страйда в процессе свертки. Эта техника получила название страйдинга. Страйд со значением один означает, что используется описанная ранее операция свертки, в которой окно свертки двигается по исходной матрице с единичным шагом. При страйде со значением 2 ядро свертки смещается на два пиксела исходной матрицы, что дает уменьшение размера выходной матрицы примерно в два раза и т.д. В современных архитектурах нейронных сетей, таких как ResNet пришли к использованию страйдов, полностью исключив пулинговые слои. Необходимо заметить, что величина страйда не должна быть слишком большой, иначе операция свертки будет пропускать наличие признаков.

3.3 Свёрточный слой

Сверточный слой представляет собой набор карт признаков, иначе говоря, набор ядер свертки. Количество карт, которые еще называют фильтрами, определяется требованиями к задаче, например, если взять

слишком большой число карт, то повысится качество распознавания, но увеличится вычислительная сложность.

Один фильтр представляет собой набор ядер свертки, например, для цветного RGB изображения одному фильтру соответствует три ядра – по одному на каждый канал изображения. Для черно-белого изображения понятия фильтра и ядра сливаются в одно целое.

Каждый фильтр свёрточного слоя генерирует один единственный выходной канал. На вход слою передается набор каналов изображения. Каждый фильтр обладает набором ядер для анализа каждого канала изображения из набора. Описанная ранее операция свертки применяется разными ядрами фильтра над разными каналами входных данных. Затем получившиеся матрицы складываются поэлементно, таким образом вычисляется результат работы фильтра. Для окончания процесса к результату применяют смещение. В СНН смещение применяется для всех фильтров слоя и представляет собой поэлементное сложение элементов результирующей матрицы с некоторой константой, которая так же является параметром обучения. Таким образом, на выходе сверточного слоя генерируется количество каналов, равное количеству фильтров слоя.

3.4 Архитектура свёрточной нейронной сети

Свёрточная нейронная сеть состоит из разного вида слоев, таких как свёрточные слои, слои пулинга и полносвязные слои. Входной слой представляет собой исходное изображение. Это изображение проходит через ряд сверточных слоев, перемешанных со слоями пулинга, в результате чего размер изображения уменьшается с увеличением числа каналов. После нескольких прохождений свёртки изображения и уплотнения с помощью пулигна размер изображения вырождается в вектор, а большой набор каналов характеризует самые абстрактные визуальные признаки, найденные на изображении. Эти данные объединяются и передаются на выход

полносвязной нейронной сети, в которой может быть несколько слоев, но как правило не много. При этом полносвязные слои утрачивают пространственную структуру пикселей. Общая классическая схема архитектуры свёрточной сети представлена на Рисунок 10 .

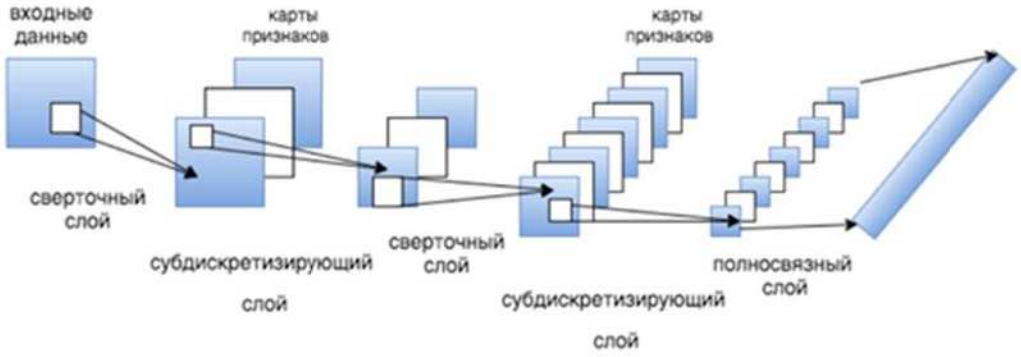

Рисунок 10 – Схема архитектуры сверточной нейронной сети

К преимуществам сверточной архитектуры нейронных сетей относят:

1) гораздо меньшее число настраиваемых весов, по сравнению с полносвязными сетями, связанное с использованием ядра свертки для всего изображения, благодаря чему не нужно для анализа разных фрагментов изображения отводить свои собственные веса;

2) возможность удобного распараллеливания вычислений, а значит и перенос нагрузки при обучении сети на графический процессор;

3) возможность обучения сети обычным методом обратного распространения ошибки, по аналогии с другими сетями;

4) относительная устойчивость алгоритма к сдвигу и повороту распознаваемого изображения.

Недостатком СНН является большое число варьируемых параметров. К таким параметрам относятся количество слоев, размерность ядра свертки для каждого из них и количество извлекаемых фильтров на слое, функция активации, размер страйда свертки и пулинга, размер ядра пулинга и его статистическая функция, использование падинга и параметры полносвязной

сети. Все эти параметры оказывают влияние на качество обучения модели и подбираются разработчиком в некотором роде экспериментальным образом. Существуют лишь некоторые общие рекомендации к организации подобных сетей, которые так же были получены эмпирически другими исследователями.

3.5 Выбор функции активации

Выбор функции активации нейронов является одним из этапов при разработке нейронной сети. Выбранная функция активации во многом определяет функциональные возможности нейронной сети и метод обучения этой сети. На сетях небольшой глубины алгоритм обратного распространения ошибки работает хорошо, но с увеличением глубины эффективность алгоритма заметно снижается. Одна из причин такого явления – затухание градиентов. По мере распространения ошибки от выходного слоя сети к выходному на каждом слое происходит умножение на производную функции активации. Если производная функции меньше единицы на всей области определения, то после прохождения алгоритмом нескольких слоев величина изменения весов будет крайне мала, что замедлит скорость обучения. Если же функция активации имеет неограниченную производную, то может произойти большое увеличение дельты весов, вычисленной на некотором слое, а это приводит к неустойчивости всего процесса обучения. Еще одной важной характеристикой функции активации является сложность вычисление ее производной. Чем легче вычисляется производная, тем меньше нагрузки будет получать процессор при обучении.

Одна из функций, используемых в качестве функции активации в нейронных сетях – сигмоидальная функция (15), график которой показан на рисунке Рисунок 11 .

$$f(x) = \frac{1}{1 + e^x} \tag{15}$$

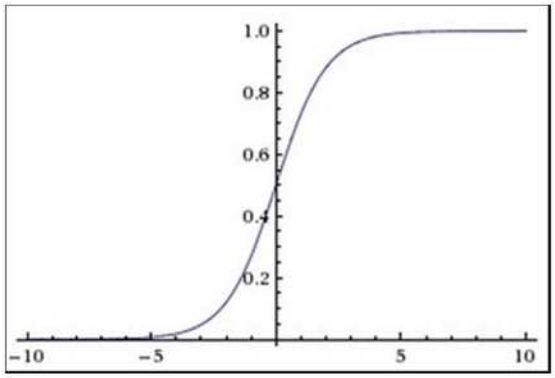

Рисунок 11 – График сигмоидальной функция активации

Недостатком этой функции является тот факт, что при насыщении функции со стороны 0 либо 1, градиент на этих участках становится близким к нулю. Из-за этой особенности необходимо следить за первоначальной инициализацией весов и не допускать слишком больших по модулю значений, в противном случае сеть будет обучаться плохо.

Гиперболический тангенс (16) – еще она функция, применяемая на практике в качестве функции активации.

$$f(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{16}$$

Производная данной функции вычисляется через ее значение, что снижает количество вычислений во время работы метода обратного распространения ошибки. Область значения в интервале (-1; 1) так же бывает удобной. Из недостатков функции для использования в качестве функции активации можно назвать риск затухания либо крутого увеличения градиента. График гиперболического тангенса представлен на Рисунок 12 .

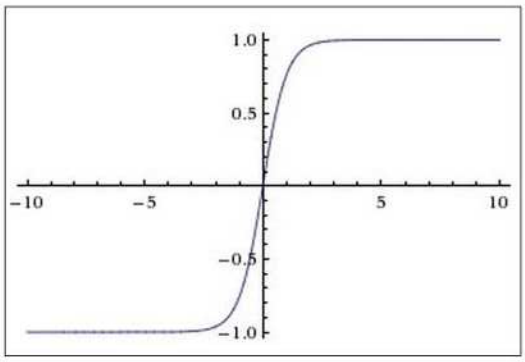

Рисунок 12 – График гиперболического тангенса

Часто используемой функции в сверточных нейронных сетях является функция ReLU (17)– с английского «rectified linear unit» – блок линейной ректификации. Эта функция хорошо применима в сверточных нейронных сетях, поскольку она отфильтровывает отрицательные значения выходов, которые можно считать шумами канала.

$$f(x) = \max(0, x) \tag{17}$$

К преимуществам ReLU можно также отнести равенство ее производной либо нулю, либо единице. Это не позволяет градиенту разрастаться либо затухать, а также прореживает весовые коэффициенты. Если сравнивать ее с гиперболическим тангенсом и сигмоидой, то их вычисление требует ресурсоемких операций, например, возведение в степень. Значение же функции ReLU может быть вычислено очень просто.

Таким образом, в настоящее время для сверточных слоев чаще всего используют функцию активации ReLU, сокращая количество шумов в канале и нагрузку на процессор при обучении.

3.6 Интерпретация откликов сети

Внутреннее состояние обученной нейронной сети характеризуется значениями подобранных в ходе обучения весовых коэффициентов. Узнать

по этим коэффициентам как сеть приходит к решениям практически невозможно. Существует целый раздел машинного обучения, направленный на разработку способов интерпретации откликов нейронных сетей. Одним из сильных инструментов в этой области является визуализация признаков, распознаваемых на изображениях [8]. Визуализация признаков позволяет понять, что сеть или ее части пытаются отыскать, используя те или иные фильтры, а так же отвечает на вопрос о том, каким должно быть исходное изображение, чтобы получить наибольший эффект от использования фильтра. Если посмотреть результат применения некоторого фильтра в свёрточной сети, то это можно считать визуализацией работы выбранного фильтра. На рисунке Рисунок 13 показана визуализация результатов применения фильтров первого сверточного слоя сети GoogLeNet [8]

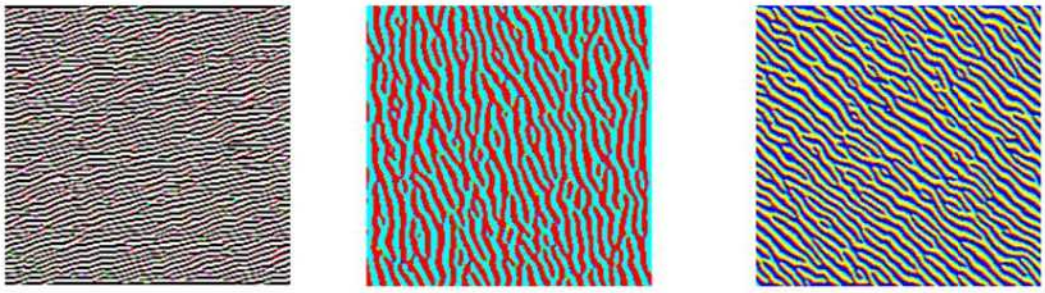

Рисунок 13 – Визуализация характеристик некоторых каналов первого слоя сети GoogLeNet

Как видно из этого рисунка фильтры первых слоев занимаются распознаванием на изображении разного вида границ. Заметим, что свернутые изображения по-прежнему остаются изображениями, каждая точка такого изображения определяется пучком точек исходного изображения из той же области и в дальнейшем эта точка будет использована для выявления характеристик более глубокого, абстрактного уровня.

В сверточных нейронных сетях применяется решение, которое заключается в уменьшении размера входной матрицы от слоя к слою, при

увеличении количества фильтров. Этот эффект как раз и достигается за счет пулинга и страйдов. Операция свёртки отображает пучок входных сигналов в один выходной сигнал. Ту часть матрицы исходных сигналов, которая сворачивается в один выходной сигнал, называют рецептивным полем. Оно характеризуется величиной, названной плотностью рецептивного поля – это количество информации, которое может поместиться в пределах рассматриваемого пучка сигналов. Суть свертывания в том, чтобы уместить в одном сигнале как можно больше информации об исходном изображении, то есть в увеличении плотности рецептивного поля. Чем глубже по слоям нейронной сети мы будем двигаться, тем большая плотность рецептивного поля будет достигаться. Ответ каждого сверточного слоя характеризует исходное изображение, представляя собой его сжатую версию. Благодаря уплотнению рецептивного поля у свёрточных нейронных сетей появляется возможность работы с более сложными характеристиками исходного изображение, нежели границы и примитивные текстуры, как это происходит на первых слоях. Используя даунсорсинг в виде пулигна или страйдирования, нейронная сеть формирует все более и более абстрактные детекторы сложных характеристик изображения по мерее ее продвижения по слоям. На рисунке Рисунок 14 показана визуализация фильтров 4 и 5 слоя сети GoogLeNet [8].

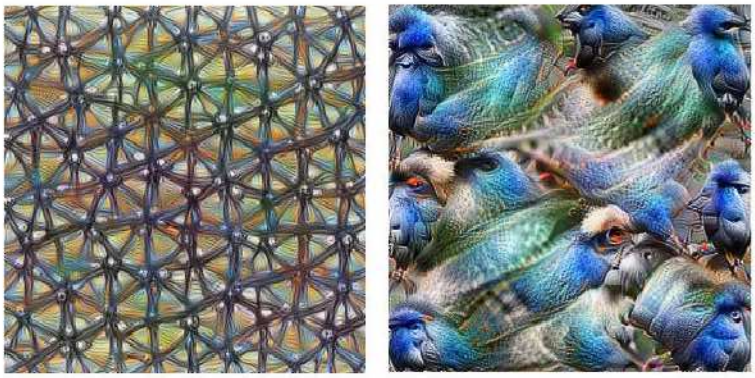

Рисунок 14 – Визуализация характеристик произвольных каналов 4 и 5 слоя сети GoogLeNet

Таким образом, первые слои нейронной сети активируются, при обнаружении элементов низкого уровня, таких как линии и границы. Слои более высоких уровней работают с более высокоуровневыми шаблонами и могут распознавать некоторые объекты. Последний пулинговый слой переводит небольшую оставшуюся матрицу сигналов в один выходной сигнал, для которого плотность рецептивного поля настолько велика, что этот сигнал характеризует все исходное изображение.

4 Реализация программной системы

4.1 Инструменты разработки

В рамках проекта в качестве инструмента для разработки и обучения модели нейронной сети был выбран интерпретируемый язык программирования Python, а конкретно его распространенная реализация CPython, написанная на C. Язык предлагает широкие возможности машинного обучения, например, предоставляет удобные зарекомендовавшие себя библиотеки TensorFlow и Keras. Так же при работе с нейронными сетями немаловажную роль имеет предобработка входных данных для обучения. Python предоставляет множество удобных библиотек для работы с данными, например, NumPy – открытая библиотека для работы удобной работы с многомерными матрицам, PIL (Pythin Image Library) поддерживающая работу с разными изображениями.

Для написания модели нейронной сети использовался Keras – это открытая нейросетевая библиотека, которая представляет собой надстройку над такими фреймворками, как Deeplearning4j, TensorFlow и Theano. Использование Keras дает компактный малословный код, описывающий многослойные сети разных архитектур.

Был выбран формат JPEG для хранения растровых изображений. С ним удобно работать ввиду того, что изображения, сохраненные в этом формате, имеют сравнительно небольшой размер, при этом качество изображения остается приемлемым.

В качестве сред разработки модели и мобильного приложения были выбраны редакторы кода из одного семейства от компании JetBrains. Для работы с нейронными сетями из языка Python использовалась среда разработки PyCharm, а для ведения проекта разработки под андроид была использована среда Android Studio. Удобство использования сред разработки

от JetBrains заключается в том, что стандартные настройки разных сред и доступ к их возможностям практически не отличается.

4.2 Процесс разработки

В качестве архитектуры нейронной сети была выбрана сверточная нейронная сеть. Это обусловлено опытом других исследователей в направлении распознавания визуальных образов, а также результатами мировых конкурсов в этой области и особенностями, дающими снижение количества обучаемых параметров сети в отличие от других архитектур.

Изображения растительности имеют множество мелких деталей, а значит и большое число признаков. Ввиду это так же нежелательно брать слишком маленький размер изображений, поэтому был выбран размер изображения 250x250 пикселей. Исходя из этих ограничений, экспериментальным образом были подобраны параметры сети. Она состоит из 4 сверточных слоев, содержащих соответственно 32, 32, 64 и 64 фильтра с тремя каналами. Размер ядра фильтра составляет 3x3. Размер ядра свертки принято выбирать размерами 3x3, 5x5 или 7x7. Чем больше размер фильтра, тем больше может уменьшаться размерность выходной матрицы, но при этом уменьшается способность сети распознавать мелки детали. Так как растения могут обладать множеством мелких деталей, был выбран размер ядра близкий к минимальному. По этой же причине размер страйда (шага окна свертки) был взят равным 1. Для операции подвыборки (пулинга) был выбран пулинг с функцией максимума и окном размером 2x2, чтобы не происходило резкого отбрасывания малозначащих сигналов, которые могут быть важны.

Размер выходного слоя полносвязной подсети, идущей после сверточных слоев равен количество распознаваемых растений. Для разработанной системы это значение равно трем. Для классификации

растений на большее число классов требуются бóльшие вычислительные мощности, а также изменений характеристик модели.

В качестве функции активации сверточных слоев была выбрана функция ReLU, зарекомендовавшая себя при работе со сверточными нейронными сетями. Выходной слой полносвязной сети использует сигмоидальную функцию, с целью получения оценок принадлежности изображения классу в вероятностной форме как число из интервала (0; 1).

Для обучения модели в качестве оптимизатора используется RMSProp – метод, в котором скорость обучения настраивается для каждого параметра. Этот метод показывает хорошие показатели скорости обучения в различных приложениях и не позволяет процессу обучения остановиться в небольших локальных минимумах. Так как оптимизатор RMSProp корректирует скорость обучения, то в качестве начального параметра скорости обучения было оставлено значение по умолчанию равное 0,01. Вообще увеличение данного параметра приводит к увеличению скорости обучения, но большое его значение может не давать сойтись процессу обучения в окрестности минимума. Однако алгоритмы обучения с адаптивной скоростью как в нашем случае позволяют не заботиться о величине данного параметра.

Обычно для обучения сети все доступные данные делят на три части. Для начала из всех данных выбирается некоторое количество контрольных примеров, на которых можно проверить работу сети после обучения. Оставшиеся данные делятся на две группы – обучающую и проверочную выборку. Обе этих группы используются сетью в процессе обучения. Обычно размер проверочных данных берут от 10 до 30% всего набора данных для обучения. В рамках данной работы было подобрано порядки 1,5 тысяч изображений для трех классов растений. Такое число данных позволило отобрать 20 процентов из них в проверочную выборку. Фрагмент обучающих данных приведен на рисунке Рисунок 15 .

Рисунок 15 – Фрагмент данных для обучения сети

Экспериментальным образом было установлено оптимальное количество эпох до достижения сетью эффекта переобучения, который проявляется в снижении величины ошибки сети на обучающих данных, но увеличение ошибки на тренировочном наборе. После прохождения 11 эпох ошибка сети на проверочных данных начинала возрастать, поэтому обучение было прервано после 11 эпохи. На рисунке Рисунок 16 приведен график достигаемой точности модели на обучающих и проверочных данных разных эпох обучения. На графике по вертикальной оси расположены значения точности от 0 до 1, а по горизонтальной оси отмечаются пройденные эпохи обучения. Модель показала итоговую точность на проверочных данных порядка 95%.

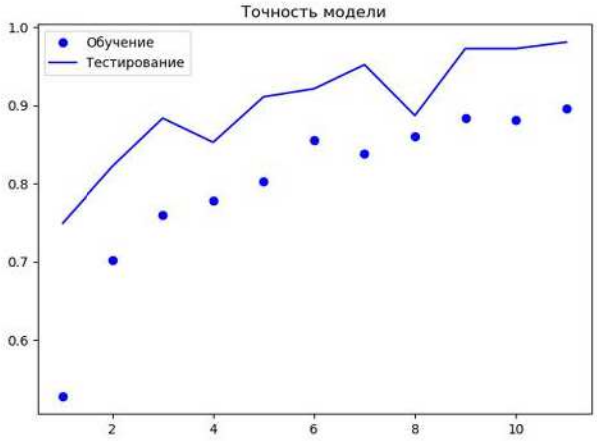

Рисунок 16 – График точности модели в процессе обучения

Для использования обученной модели keras локально в мобильном приложении ее требуется перевести в формат «TensorFlow Lite». Для этого был написан отдельны скрипт на языке Python, конвертирующий сохраненную модель в формате h5 в модель формата tflite.

Файл сконвертированной модели помещается в проект андроид приложения и используется для анализа изображений. При старте приложения отображается основой экран, на котором отображается изображение с камеры устройства. Приложение находится в непрерывном цикле анализа изображений. На анализ одного среза непрерывного сигнала с камеры уходит от 80 до 120 миллисекунд. В нижней части экрана выводятся результаты анализа изображений, которые динамически изменяются при движении камерой. Такое решение позволяет наводить камеру на разные части растения, что повышает шанс распознать объект. Основное окно приложения можно видеть на рисунке Рисунок 17 .

Рисунок 17 – Основное окно мобильного приложения

Результаты анализа изображения представляют собой вероятностные характеристики, а также существует множество сильно похожих видов

растений, что допускает шанс ошибки распознавания. Для того, чтобы оценку сети можно было проверить лично, в приложении предоставляется справка по распознаваемым видам ядовитых растений. Если пользователь коснется любой части экрана мобильного устройства, то для растения из начала списка (в идентификации которого сеть наиболее «уверена») откроется окно справки, которое можно видеть на рисунке Рисунок 18 .

Рисунок 18 – Окно справки мобильного приложения

Окно является скроллируемым по вертикали и содержит пару характерных изображений растения, по которым его можно опознать. Ниже под изображениями приводится текстовая справка по данному растению.

В приложении организована удобная система конфигурации через файлы. Список названий, описания растений и изображения представляются файлами, именуются определенным образом и располагаются в нужных папках проекта. Требуется только, чтобы количество записей в списке названий распознаваемых растений совпадало с числом выходных нейронов в прилагаемой модели. Таким образом, никак не требуется изменять код мобильного приложения при изменении модели нейронной сети.

ЗАКЛЮЧЕНИЕ

Цель работы – разработка мобильного приложения для распознавания ядовитых растений, включающая разработку модели сверточной сети и ее обучение – достигнута.

В рамках данной работы была рассмотрена история развития искусственных нейронных сетей, подробно изучено устройство сверточных сетей, разработана и обучена модель сверточной сети для распознавания растительных образов и написано мобильное приложение, работающее с облегченной моделью «TensorFlow Lite» без доступа в сеть. Разработанное мобильное приложение «Ядовитые растения Кавказа» используя камеру устройства постоянно считывает изображения и дает оценку обнаружения на них некоторых ядовитых растений. Так же приложение предоставляет справку о растении, определенном на изображении с наибольшей вероятностью.

Разработанная нейронная сеть обучалась на подготовленном наборе данных, а затем была протестирована на исключенной из общего набора обучающих данных выборке.

Результаты исследования доказали удачность подобранных параметров сети и успешность сверточных нейронных сетей в анализе сложных растительных образов на изображениях.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 "Сильный" искусственный интеллект – наследник человечества. Часть 1. URL: https://scientificrussia.ru/articles/silnyj-iskusstvennyj-intellekt-naslednik-chelovechestva (дата обращения 11.04.2020)

2 Экспертные системы: учебно-методическое пособие / А. Л. Каткова; Шадр. гос. пед. ин-т. – Шадринск, 2011. – 92с.

3 Нейронные сети. Большая Российская энциклопедия. URL : https://bigenc.ru/technology_and_technique/text/4114009 (дата обращения 24.05.2020)

4 Модель МакКаллока-Питтса. URL: http://www.machinelearning.ru/wiki/index.php?title=Модель_МакКаллока-Питтса (дата обращения 25.05.2020)

5 Персептрон URL: http://www.machinelearning.ru/wiki/index.php?title=Персептрон (дата обращения 24.05.2020)

6 Розенблатт Ф. Принципы нейродинамики: Перцептроны и теория механизмов мозга. Principles of Neurodynamic: Perceptrons and the Theory of Brain Mechanisms. М.: Мир, 1965.

7 Сверточные нейронные сети: взгляд изнутри. URL : http://ru.datasides.com/code/cnn-convolutional-neural-networks/ (дата обращения 31.05.2020)

8 Feature Visualization. How neural networks build up their understanding of image. URL: https://distill.pub/2017/feature-visualization/ (дата обращения 31.05.2020)

9 Hopfield J. J., «Neural networks and physical systems with emergent collective computational abilities», Proceedings of National Academy of Sciences,vol. 79 no. 8 pp. 2554–2558, April 1982

10 Rashid T., Make Your Own Neural Network, 1st edition // Tariq Rashid 2017