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

Кафедра информационных технологий

КУРСОВАЯ РАБОТА

РАСПОЗНАВАНИЕ ОБЪЕКТОВ КАРТ С ПОМОЩЬЮ НЕЙРОННОЙ СЕТИ

Работу выполнил_____И. А. Рыков
(подпись, дата)
Факультет компьютерных технологий и прикладной математики 3 курс

Направление 01.03.02 – «Прикладная математика и информатика»

Научный руководитель доц.,
канд. физ.-мат. наук _____ В. В. Подколзин
(подпись, дата)

Нормоконтролер ст. преп. _____А. В. Харченко
(подпись, дата)

Краснодар 2018

РЕФЕРАТ

Курсовая работа 39 с., 4 ч., 18 рис., 13 источников.

НЕЙРОННАЯ СЕТЬ, НЕЙРОСЕТЬ, ПЕРЦЕПТРОН, ОБУЧЕНИЕ, УЧИТЕЛЬ, ОШИБКА, ИЗОБРАЖЕНИЕ, КАРТА, АНАЛИЗ, СПУТНИК, СНИМОК, ПОВЕРХНОСТЬ.

Цель работы – разработка программы, позволяющей разрешить задачу идентификации и классификации изображений.

Объектом исследования являются нейронные сети, в частности, свёрточные нейронные сети.

В результате проведенной работы разработано компьютерное приложение, реализующее решение задачи классификации изображения географической области на спутниковых снимках земной поверхности посредством свёрточной нейронной сети.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире системы искусственного интеллекта внедряются повсеместно, во всех сферах нашей жизни: в науке и технике, мультимедиа, экономике, рекламе и маркетинге, здравоохранении. В связи с колоссальным ростом объема всевозможных данных нейронные сети становятся незаменимым инструментом для их анализа.

Цель данного проекта состоит в разработке программного приложения, включающей создание и обучение сверточной нейронной сети для классификации объектов карт. Объект исследования – спутниковый снимок земной поверхности. Необходимо классифицировать объект на снимке: водоём, населённый пункт, горный массив.

Таким образом, в дальнейшем предполагается использование нескольких сетей для распознавания типа и названия населенного пункта или водоёма.

1 Нейронные сети и их классификация

Искусственная нейронная сеть (ИНС) — математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма [1].

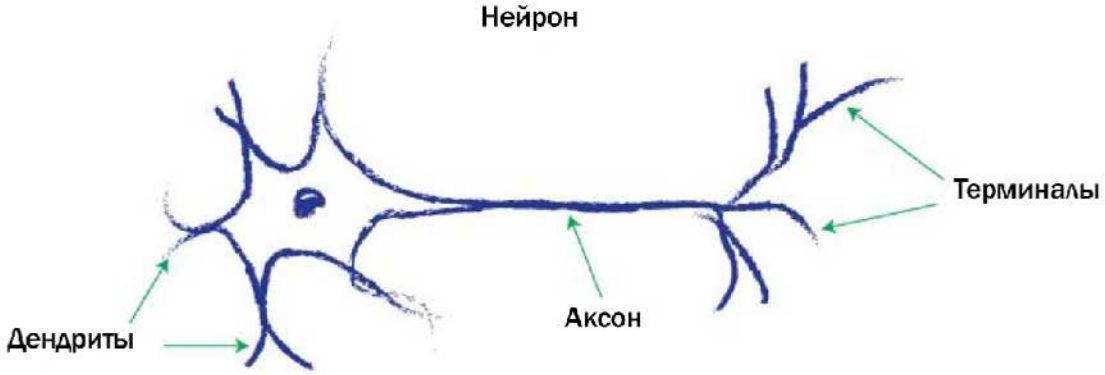


Рисунок 1.1 – Биологическое представление нейрона

О нейросетях стало известно в 50-х годах прошлого века. В период с 1943 по 1950 год были представлены миру первые две основополагающие ученые работы. Одна из них, статья 1943 года двух выдающихся ученых - Уорена Маккалока и Уолтера Питтса освещала математическую модель нейронной сети, а в 1949 году канадский нейропсихолог Дональд Хебб выпустил книгу "Организация поведения", в которой было подробное описание процесса самообучения искусственной нейронной сети.

В 1957 году Фрэнком Розенблаттом был предложен перцептрон - математическая (компьютерная) модель обработки информации человеческим мозгом. Данная разработка уже в те годы умела прогнозировать погоду и распознавать образы [2].

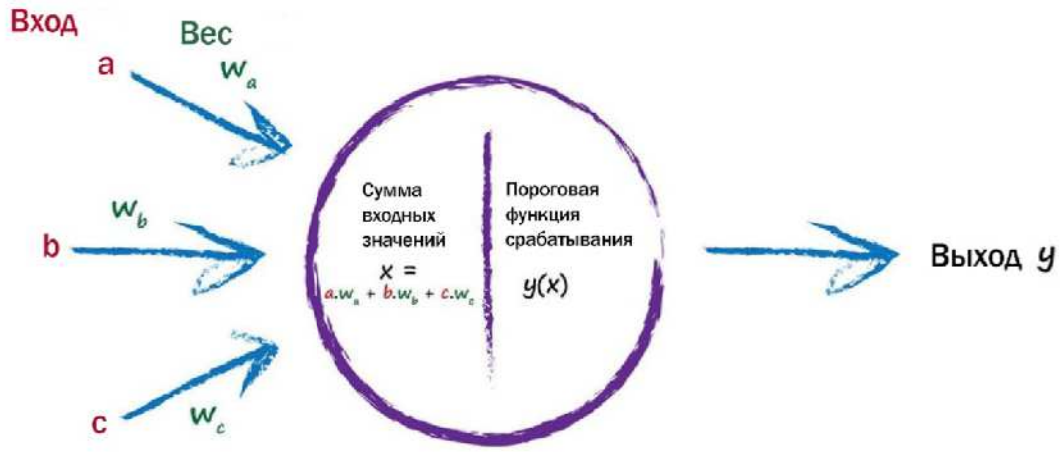


Рисунок 1.2 – Классическая модель искусственного нейрона

Возможность обучения — одно из главных преимуществ нейронных сетей перед традиционными алгоритмами. Технически обучение заключается в нахождении коэффициентов связей между нейронами. В процессе обучения нейронная сеть способна выявлять сложные зависимости между входными данными и выходными, а также выполнять обобщение. Это значит, что в случае успешного обучения сеть сможет вернуть верный результат на основании данных, которые отсутствовали в обучающей выборке, а также неполных и/или «зашумленных», частично искажённых данных [1].

Классификация искусственных нейронных сетей по применяемой структурной модели:

1) перцептрон - нейронная сеть прямого распространения, то есть сигнал поступает на вход и движется только в прямом направлении;

2) рекуррентные нейронные сети - нейронные сети, где присутствует обратная связь, подразумевающая связь от логически более удаленного элемента к менее удаленному;

3) сверточные нейронные сети - нейронные сети прямого распространения, предназначенные в основном для эффективной классификации и обработки изображений.

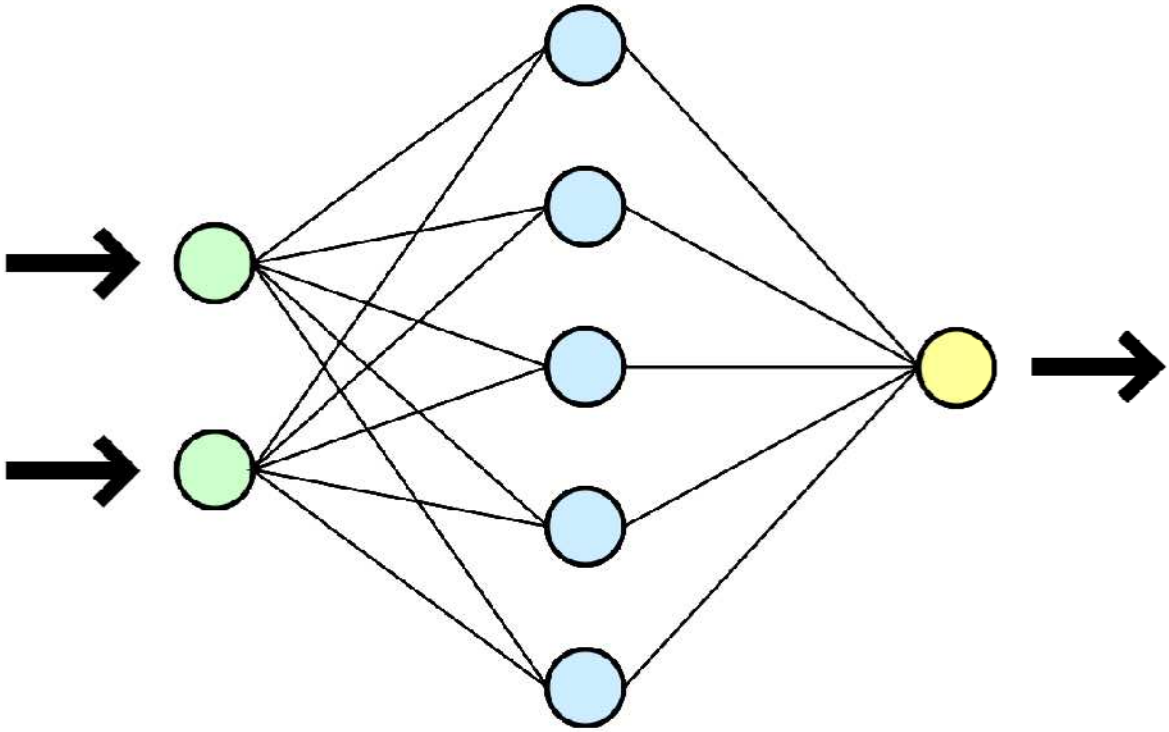


Рисунок 1.3 - Схема простой нейросети. Зелёным цветом обозначены входные нейроны, голубым — скрытые нейроны, жёлтым — выходной нейрон [1]

1.1 Перцептрон

Перцептрон, или персептрон — математическая или компьютерная модель восприятия информации мозгом (кибернетическая модель мозга). Перцептрон - одна из первых моделей нейронных сетей. Несмотря на свою простоту, перцептрон способен учиться и решать достаточно сложные задачи. Основная математическая задача, с которой он справляется — это линейное разделение произвольных нелинейных множеств, так называемое обеспечение линейной сепарабельности.

Персептрон состоит из трех типов элементов, а именно: сигналы, поступающие от датчиков, передаются в ассоциативные элементы, а затем в реагирующие. Таким образом, перцептроны позволяют создать набор «ассоциаций» между входящими стимулами и необходимой реакцией на

выходе. В биологическом плане это соответствует превращению, например, зрительной информации в физиологический ответ двигательных нейронов.

Согласно современной терминологии, перцептроны могут быть классифицированы как искусственные нейронные сети:

– с одним скрытым слоем;

– с пороговой передаточной функцией;

– с прямым распространением сигнала [3].

Элементарный перцептрон состоит из элементов трех типов: S-элементов, A-элементов и одного R-элемента. S-элементы — это слой сенсоров, или рецепторов. В физическом воплощении они отвечают, например, светочувствительным клеткам сетчатки глаза или фоторезисторам матрицы камеры. Каждый рецептор может находиться в одном из двух состояний — покоя или возбуждения, и только в последнем случае он передает единичный сигнал к следующему слою, ассоциативным элементам.

A-элементы называются ассоциативными, потому что каждому такому элементу, как правило, соответствует целый набор (ассоциация) S-элементов. A-элемент активизируется, как только количество сигналов от S-элементов на его входе превышает определенную величину θ.

Сигналы от возбужденных A-элементов, в свою очередь, передаются в сумматор R, причем сигнал от i-го ассоциативного элемента передается с коэффициентом. Этот коэффициент называется весом AR связи.

Так же, как и A-элементы, R-элемент подсчитывает сумму значений входных сигналов, умноженных на веса (линейную форму). R-элемент, а вместе с ним и элементарный перцептрон, выдает «1», если линейная форма превышает порог θ, иначе на выходе будет «0». Математически, функцию, реализующую R-элемент, можно записать так:

$$f(x) = sign(\sum_{i-1}^{n} w_i x_i - \theta) \quad (1)$$

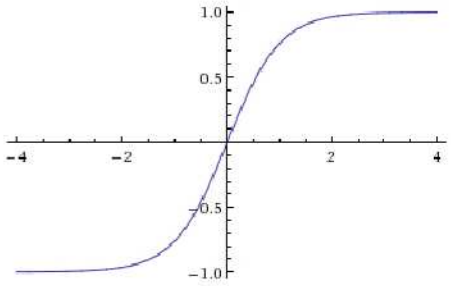
Если перцептрон получает на вход не дискретные, а непрерывные значения, то модель перцептрона можно представить в виде формулы (2):

$$y_i = f\left(\sum_{i=1}^{n} w_i x_i\right), \quad (2)$$

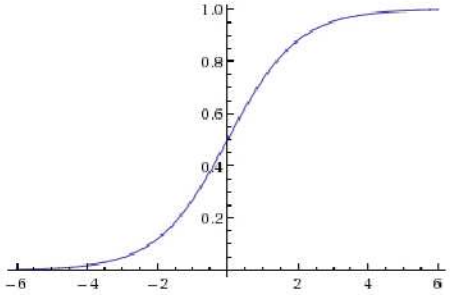
где *f(x)* – функция активации нейрона, например, сигмоидная функция (Рисунок 1.4);

w_i - вес синаптической связи нейрона;

x_i - входное значение нейрона;



$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



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

Рисунок 1.4 – Примеры сигмоидных активационных функций

Логическая схема перцептрона, где $S_1 \ldots S_8$ – слой рецепторов, $A_1 \ldots A_4$ - ассоциативные элементы, R_1 и R_2 – сумматоры, имеет вид:

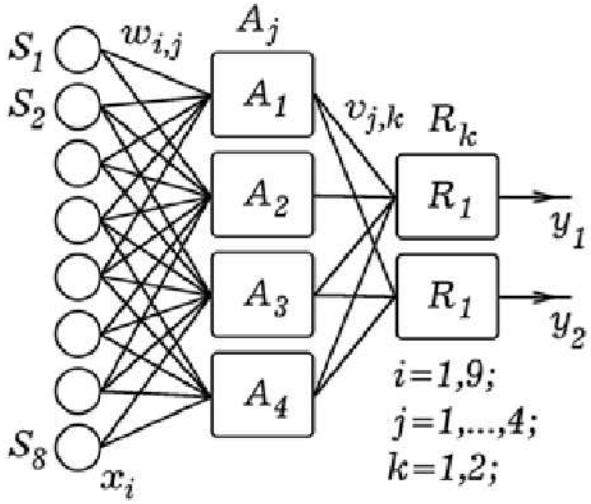


Рисунок 1.5 – Логическая схема элементарного перцептрона

1.2 Алгоритмы обучения нейронной сети

Важным свойством любой нейронной сети является способность к обучению. Процесс обучения является процедурой настройки весов и порогов с целью уменьшения разности между желаемыми (целевыми) и получаемыми векторами на выходе. Розенблатт классифицировал различные алгоритмы обучения перцептрона, называя их системами подкрепления.

Система подкрепления — это любой набор правил, на основании которых можно изменять с течением времени матрицу взаимодействия (или состояние памяти) перцептрона.

Описывая эти системы подкрепления и уточняя возможные их виды, Розенблатт основывался на идеях Д. Хебба об обучении, предложенных им в 1949 году, которые можно перефразировать в следующее правило, состоящее из двух частей:

Если два нейрона по обе стороны синапса (соединения) активизируются одновременно (то есть синхронно), то прочность этого соединения возрастает.
Если два нейрона по обе стороны синапса активизируются асинхронно, то такой синапс ослабляется или вообще отмирает.

Выделяют несколько методов обучения нейронной сети:

- обучение с учителем;
- обучение без учителя;
- метод обратного распространения ошибки [3].

1.2.1 Обучение с учителем

Классический метод обучения перцептрона — это метод коррекции ошибки. Он представляет собой такой вид обучения с учителем, при котором вес связи не изменяется до тех пор, пока текущая реакция перцептрона остается правильной. При появлении неправильной реакции вес изменяется на единицу, а знак (плюс/минус) определяется противоположным от знака ошибки.

Допустим, мы хотим обучить перцептрон разделять два класса объектов так, чтобы при предъявлении объектов первого класса выход перцептрона был положителен (плюс 1), а при предъявлении объектов второго класса — отрицательным (минус 1) [3].

Перцептроном будем называть устройство, вычисляющее следующую систему функций (3):

$$\psi = \left[\sum_{i=1}^{n} w_{ij} x_i > \theta \right], \tag{3}$$

где $j=1,...,n$,

w_i - веса перцептрона;

x_i - значения входных сигналов;

θ - порог срабатывания. Квадратные скобки означают переход от булевых значений к числовым.

Обучение персептрона состоит в подстройке весовых коэффициентов. Пусть имеется набор пар векторов $(x^\alpha, y^\alpha), \alpha = 1, \ldots, p$, называемый обучающей выборкой. Будем называть нейронную сеть обученной на данной обучающей выборке, если при подаче на входы сети каждого вектора x^α на выходах всякий раз получается соответствующий вектор y^α.

Предложенный Розенблаттом метод обучения состоит в итерационной подстройке матрицы весов, последовательно уменьшающей ошибку в выходных векторах. Алгоритм включает несколько шагов:

Шаг 0: начальные значения всех весов нейронов W(t = 0) полагаются случайными.

Шаг 1: сети предъявляется входной образ x^α в результате формируется выходной образ $\tilde{y}^\alpha \neq y^\alpha$.

Шаг 2: вычисляется вектор ошибки $\delta^\alpha = (y^\alpha - \tilde{y}^\alpha)$, совершаемой сетью на выходе. Дальнейшая идея состоит в том, что изменение вектора весовых коэффициентов в области малых ошибок должно быть пропорционально ошибке на выходе и равно нулю, если ошибка равна нулю.

Шаг 3: Вектор весов модифицируется по формуле (4):

$$W(t+\Delta T) = W(t) + \eta x^{n} \cdot (\delta^{n})^{T}, \quad (4)$$

где $0 < \eta < 1$ - темп обучения.

Шаг 4: шаги 1—3 повторяются для всех обучающих векторов. Один цикл последовательного предъявления всей выборки называется эпохой. Обучение завершается по истечении нескольких эпох: а) когда итерации сойдутся, т.е. вектор весов перестает изменяться, или б) когда полная, просуммированная по всем векторам абсолютная ошибка станет меньше некоторого малого значения [5].

1.2.2 Обучение без учителя

Рассмотренный выше алгоритм обучения нейронной сети подразумевает наличие некоего внешнего звена, предоставляющего сети наличие не только входных данных, но и эталонных выходных значений для обучающей выборки.

Главным преимуществом метода обучения без учителя является «самостоятельность» сети. Процесс обучения, как и в случае обучения с учителем заключается в подстройке весов синапсов (связей). Некоторые алгоритмы меняют структуру сети: количество нейронов и их связей, такие преобразования называются самоорганизацией сети.

Очевидно, подстройка весов связей между нейронами может производиться только на основании информации, доступной в нейроне: его состояниях и уже имеющихся весовых коэффициентов.

Д. Хеббом был разработан сигнальный метод обучения, в котором изменение весов синаптических связей происходит по следующему правилу (5):

$$w_{ij}(t) = w_{ij}(t-1) - \alpha \cdot y_i^{(n-1)} \cdot y_j^{(n)}, \quad (5)$$

где $y_i^{(n-1)}$ – выходное значение нейрона *i* слоя *n-1*;

$y_j^{(n)}$ – выходное значение нейрона *j* слоя *n*;

$w_{ij}(t)$ и $w_{ij}(t-1)$ – весовые коэффициенты связи, соединяющей эти нейроны, на итерациях *t* и *t-1*;

α - коэффициент скорости обучения.

Существует также и дифференциальный метод обучения Хебба (6):

$$w_{ij}(t) = w_{ij}(t-1) - \alpha \cdot \left[y_i^{(n-1)}(t) - y_i^{(n-1)}(t-1) \right] \cdot \left[y_j^{(n)}(t) - y_j^{(n)}(t-1) \right], \quad (6)$$

где $y_i^{(n-1)}(t)$ и $y_i^{(n-1)}(t-1)$ – выходное значение нейрона *i* слоя *n-1* соответственно на итерациях *t* и *t-1*; $y_j^{(n)}(t)$ и $y_j^{(n)}(t-1)$ – то же самое для нейрона *j* слоя *n*. Как видно из формулы (6), сильнее всего обучаются связи, соединяющие те нейроны, выходы которых наиболее быстро изменились в сторону увеличения.

Полный алгоритм обучения с применением приведенных формул (5) и (6) выглядит следующим образом:

Шаг 0: на стадии инициализации всем весовым коэффициентам присваиваются небольшие случайные значения.

Шаг 1: на входы сети подается входной образ, сигналы передаются согласно принципам классического прямого распространения, то есть для каждого нейрона вычисляется сумма его входных значений, к которой затем применяется функция активации нейрона. В результате формируется его выходное значение.

Шаг 2: на основании полученных выходных значений нейронов по формуле (5) или (6) производится изменение весовых коэффициентов.

Шаг 3: цикл возобновляется с первого шага, пока выходные значения сети не стабилизируются с заданной точностью. Применение этого нового способа определения завершения обучения, отличного от используемого для сети обратного распространения, обусловлено тем, что подстраиваемые значения весовых коэффициентов синапсов практически не ограничены. На втором шаге цикла попеременно предъявляются все образы из входного набора.

Следует отметить, что вид откликов на каждый класс входных образов неизвестен заранее и представляет собой произвольное сочетание состояний нейронов, обусловленное выбором случайных весовых значений на этапе

инициализации. Вместе с тем, сеть способна обобщать схожие образы, относя их к одному классу.

Другой алгоритм обучения без учителя – алгоритм Кохонена – предусматривает подстройку синапсов на основании их значений от предыдущей итерации (7):

$$w_{ij}(t) = w_{ij}(t-1) - \alpha \cdot \left[y_i^{(n-1)}(t) - w_{ij}(t-1) \right]. \quad (7)$$

Из приведенной выше формулы видно, что обучение сводится к минимизации разницы между входными сигналами нейрона, поступающими с выходов нейронов предыдущего слоя $y_i^{(n-1)}$, и весовыми коэффициентами его синапсов.

Полный алгоритм обучения имеет примерно такую же структуру, как в методе Хебба, но на втором шаге из всего слоя выбирается нейрон, значения синапсов которого максимально походят на входной образ, и подстройка весов по формуле (7) проводится только для него. Эта, так называемая, аккредитация может сопровождаться затормаживанием всех остальных нейронов слоя и введением выбранного нейрона в насыщение. Выбор такого нейрона может осуществляться, например, расчетом скалярного произведения вектора весовых коэффициентов с вектором входных значений. Максимальное произведение дает выигравший нейрон [6].

1.2.3 Метод обратного распространения ошибки

Для обучения многослойных сетей рядом учёных, в том числе Д. Румельхартом, был предложен градиентный алгоритм обучения с учителем, проводящий сигнал ошибки, вычисленный выходами перцептрона, к его входам, слой за слоем. Сейчас это самый популярный метод обучения многослойных перцептронов. Его преимущество в том, что он может обучить все слои нейронной сети, и его легко просчитать локально. Однако этот метод является очень долгим, к тому же, для его применения нужно, чтобы

передаточная функция нейронов (функция срабатывания) была дифференцируемой. При этом в перцептронах пришлось отказаться от бинарного сигнала, и пользоваться на входе непрерывными значениями.

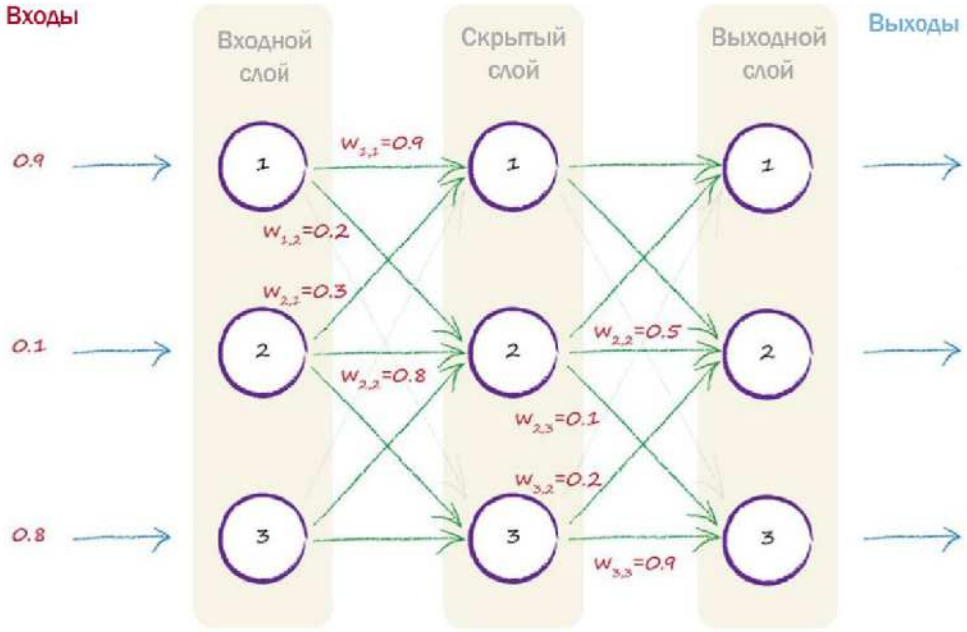


Рисунок 1.6 – Нейронная сеть с одним скрытым слоем

Для обучения многослойного перцептрона нельзя применить правило Розентблатта, так как для применения метода обучения с учителем требуются верные выходные значения, которые есть только для выходного слоя. Основная идея метода обратного распространения лежит в нахождении оценки ошибки для нейронов скрытых слоёв [7].

Чем больше значение синаптической связи между нейроном скрытого слоя и выходным нейроном, тем сильнее ошибка первого влияет на ошибку второго. Следовательно, оценку ошибки элементов скрытых слоев можно получить как взвешенную сумму ошибок последующих слоев. При обучении информация распространяется от низших слоев иерархии к высшим, а оценки ошибок, совершаемые сетью - в обратном направлении, что и отражено в названии метода.

Общая структура алгоритма сходна с описанными выше, но в данном случае усложняются формулы подстройки весов. Алгоритм метода обратного распространения ошибки имеет вид:

Шаг 0: начальные значения всех весов нейронов W(t = 0) полагаются случайными.

Шаг 1: сети предъявляется входной образ x^0, при этом формируется выходной образ y^0. При этом нейроны последовательно от слоя к слою функционируют по следующим формулам (8, 9):

$$x_j = \sum_i w_{ij} x_i^0 ; y_j = f(x_j) \quad (8) \text{ - скрытый слой;}$$

$$x_k = \sum_j v_{jk} y_j ; y_k = f(x_k) \quad (9) \text{ - выходной слой,}$$

где *f(x)* – сигмоидная функция, определяемая по формуле (10):

$$f(x) = \frac{1}{1 + e^{-1}} \tag{10}$$

Шаг 2: функционал квадратичной ошибки сети для данного входного образа имеет вид (11):

$$E = \frac{1}{2} \sum_k (y_k - Y_k^0)^2 \tag{11}$$

Данный функционал подлежит минимизации. Классический градиентный метод оптимизации состоит в итерационном уточнении аргумента согласно формуле (12):

$$v_{jk}(t-1) = v_{jk}(t) \quad h \frac{\partial E}{\partial v_{jk}}, \tag{12}$$

где h – параметр темпа обучения, выбираемый достаточно малым для сходимости алгоритма.

Функция ошибки в явном виде не содержит зависимости от веса v_{jk}, поэтому воспользуемся формулами неявного дифференцирования сложной функции (13-15):

$$\frac{\partial E}{\partial y_k} - \delta_k - (y_k - Y_k^\alpha) \tag{13}$$

$$\frac{\partial E}{\partial x_k} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial x_k} = \delta_k \cdot y_k(1 - y_k) \tag{14}$$

$$\frac{\partial E}{\partial v_{jk}} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial x_k} \cdot \frac{\partial x_k}{\partial v_{jk}} = \delta_k \cdot y_k(1 - y_k) \cdot y_j \tag{15}$$

Здесь учтено полезное свойство сигмоидной функции *f(x)*: ее производная выражается только через само значение функции, *f'(x)=f(1-f)*. Таким образом, все необходимые величины для подстройки весов выходного слоя v получены.

Шаг 3: на этом шаге выполняется подстройка весов скрытого слоя. Градиентный метод по-прежнему дает (16):

$$W_{ij}(t+1) = W_{ij}(t) - h \cdot \frac{\partial E}{\partial W_{ij}} \tag{16}$$

Вычисления производных выполняются по тем же формулам, за исключением некоторого усложнения формулы для ошибки δ_j (17-19):

$$\frac{\partial E}{\partial x_k} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial x_k} = \delta_k \cdot y_k(1 - y_k) \tag{17}$$

$$\frac{\partial E}{\partial y_j} = \delta_j = \sum_k \frac{\partial E}{\partial x_k} \cdot \frac{\partial x_k}{\partial y_j} = \sum_k \delta_k \cdot y_k(1 - y_k) \cdot v_{jk} \tag{18}$$

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial x_j} \cdot \frac{\partial x_j}{\partial W_{ij}} = \delta_j \cdot y_j(1-y_j) \cdot X_i^\alpha =$$
$$\left[\sum_k \delta_k \cdot y_k(1 - y_k) \cdot v_{jk}\right] \cdot [y_j(1 - y_j) \cdot X_i^\alpha] \tag{19}$$

При вычислении здесь δ_j и был применен принцип обратного распространения ошибки: частные производные берутся только по переменным последующего слоя. По полученным формулам модифицируются веса нейронов скрытого слоя. Если в нейронной сети имеется несколько скрытых слоев, процедура обратного распространения применяется последовательно для каждого из них, начиная со слоя, предшествующего выходному, и далее до

слоя, следующего за входным. При этом формулы сохраняют свой вид с заменой элементов выходного слоя на элементы соответствующего скрытого слоя.

Шаг 4: шаги 1-3 повторяются для всех обучающих векторов. Обучение завершается по достижении малой полной ошибки или максимально допустимого числа итераций, как и в методе обучения Розенблатта [7].

1.3 Рекуррентные нейронные сети

Рекуррентные нейронные сети (англ. Recurrent neural network; RNN) — вид нейронных сетей, в которых имеется обратная связь. При этом под обратной связью подразумевается связь от логически более удалённого элемента к менее удалённому. Наличие обратных связей позволяет запоминать и воспроизводить целые последовательности реакций на один стимул. С точки зрения программирования в таких сетях появляется аналог циклического выполнения, а с точки зрения систем — такая сеть эквивалентна конечному автомату. Такие особенности потенциально предоставляют множество возможностей для моделирования биологических нейронных сетей. Однако большинство возможностей на данный момент плохо изучены в связи с возможностью построения разнообразных архитектур и сложностью их анализа [7].

Если предполагается использование нейронной сети, например, для обработки текста или звука, то есть в общем случае – любой условно бесконечной последовательности, в которой важно не только содержание, но и порядок, в котором следует информация, то для этого следует прибегнуть к рекуррентной нейронной сети.

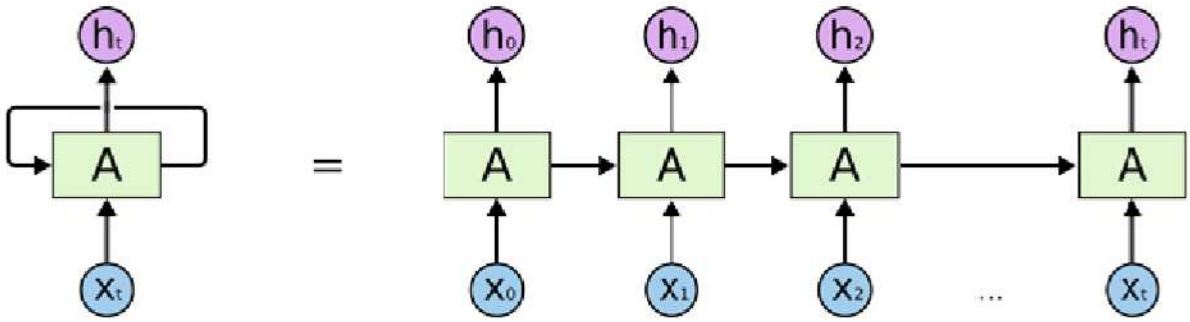


Рисунок 1.7 - Схема однослойной рекуррентной нейронной сети: на каждом цикле работы внутренний слой нейронов получает набор входных данных X и информацию о предыдущем состоянии внутреннего слоя A, на основании чего генерирует ответ h [8]

Противоположность рекуррентных сетей – сети прямого распространения, где информация передается в одном направлении от нейрона к нейрону в сторону выхода. В рекуррентных нейросетях нейроны обмениваются информацией между собой: например, вдобавок к новому фрагменту входных данных нейрон также получает некоторую информацию о предыдущем состоянии сети. Таким образом, в сети реализуется «память», что принципиально меняет характер ее работы и позволяет анализировать любые последовательности данных, в которых важно, в каком порядке идут значения — от звукозаписей до котировок акций.

Если нейронные сети прямого распространения представляют собой некоторую функцию, то рекуррентные нейронные сети можно назвать программой. В самом деле, память рекуррентных нейросетей делает их Тьюринг-полными: при правильном задании весов сеть может успешно эмулировать работу компьютерных программ [8].

Простая рекуррентная нейронная сеть или сеть Элмана состоит из трех слоёв - входного (распределительного) слоя, скрытого и выходного (обрабатывающих) слоёв. При этом скрытый слой имеет обратную связь сам на себя. На рисунке представлена схема нейронной сети Элмана:

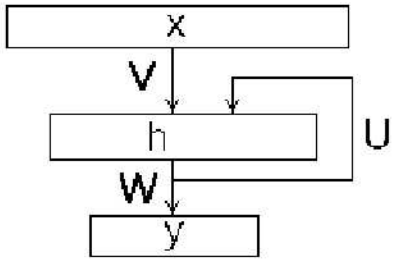


Рисунок 1.8 – Схема нейронной сети Элмана

В отличие от сети прямого распространения, входной образ рекуррентной сети это не один вектор, но последовательность векторов $\{X_1 ... X_n\}$, векторы входного образа в заданном порядке подаются на вход, при этом новое состояние скрытого слоя зависит от его предыдущих состояний. Сеть Элмана можно описать следующими соотношениями (20, 21):

$$h(t) = f(V \cdot x(t) + U \cdot h(t-1) + b_h) \tag{20}$$

$$y(t) = g(W \cdot h(t) + b_y), \tag{21}$$

где *x(t)* - выходной вектор номера *t;*

h(t) - состояние скрытого слоя для входа *x(t) (h(0)=0)*;

y(t) - выход сети для входа *x(t)*;

U - матрица весов распределительного слоя;

W - квадратная матрица весов обратных связей скрытого слоя;

b(h)- вектор сдвигов скрытого слоя;

V - матрица весов выходного слоя;

b(y) - вектор сдвигов выходного слоя;

f - функция активации скрытого слоя;

g - функция активации выходного слоя [9].

Помимо рекуррентной сети Элмана существуют и другие разновидности нейронных сетей с обратной связью.

Нейронная сеть Хопфилда - полносвязная нейронная сеть с симметричной матрицей связей. В процессе работы динамика таких сетей сходится к одному из положений равновесия. Эти положения равновесия определяются заранее в процессе обучения, они являются локальными

минимумами функционала, называемого энергией сети (в простейшем случае — локальными минимумами отрицательно определённой квадратичной формы на n-мерном кубе) [10].

Нейронная сеть Джорждана - вид нейронных сетей, который получается из многослойного перцептрона, если на его вход подать, помимо входного вектора, выходной с задержкой на один или несколько тактов [11].

1.3.1 Метод обучения рекуррентной сети

Для обучения сети Элмана применяются те же градиентные методы, что и для сетей прямого распространения, но с определёнными модификациями для корректного вычисления градиента функции ошибки. Он вычисляется с помощью модифицированного метода обратного распространения, который носит название backpropagation through time (метод обратного распространения с разворачиванием сети во времени, BPTT). Идея метода - развернуть последовательность, превратив рекуррентную сеть в "обычную". Как и в методе обратного распространения для сетей прямого распространения, процесс вычисления градиента (изменения весов) происходит в три следующих этапа:

а) прямой проход - вычисляются состояния слоёв;

б) обратный проход - вычисляется ошибка слоёв;

в) вычисление изменения весов, на основе данных полученных на первом и втором этапах [9].

Алгоритм BPTT следующий:

Шаг 1 - прямой проход: для каждого вектора последовательности: {x(1)…x(n)}:

вычисляются состояния скрытого слоя {s(1)…s(n)} и выходы скрытого слоя {h(1)…h(n)} по формулам (22, 23):

$$s(t) = Vx(t) + U \cdot h(t-1) + a \tag{22}$$

$$h(t) = f(s(t)) \tag{23}$$

и вычисляется выход сети *y* (24):

$$y(n) = g(W \cdot h(n) + b) \tag{24}$$

Шаг 2 – обратный проход: вычисляется ошибка выходного слоя

$$\delta_o = y - d, \tag{25}$$

вычисляется ошибка скрытого слоя в конечном состоянии

$$\delta_h(n) = W^T \cdot \delta_o \odot f'(s(n)), \tag{26}$$

вычисляются ошибки скрытого слоя в промежуточных состояниях

$$\delta_h(t) = U^T \cdot \delta_{h(t)}(t+1) \odot f'(s(n)), t = (1, \ldots, n). \tag{27}$$

Шаг 3 – вычисление изменения весов (28-32):

$$\Delta W = \delta_o \cdot (h(n))^T \tag{28}$$

$$\Delta b_U = \sum \delta_o \tag{29}$$

$$\Delta V = \sum_t \delta_h(t) \cdot (x(t))^T \tag{30}$$

$$\Delta U = \sum_t \delta_h(t) \cdot (h(t-1))^T \tag{31}$$

$$\Delta b_h = \sum \sum_t \delta_h(t) \tag{32}$$

2 Свёрточная нейронная сеть

Свёрточная нейронная сеть (англ. convolutional neural network, CNN) — специальная архитектура искусственных нейронных сетей, предложенная Яном Лекуном в 1988 году и нацеленная на эффективное распознавание изображений, входит в состав технологий глубокого машинного обучения. Использует некоторые особенности зрительной коры головного мозга, в которой были открыты так называемые простые клетки, реагирующие на прямые линии под разными углами, и сложные клетки, реакция которых связана с активацией определённого набора простых клеток. Таким образом, идея свёрточных нейронных сетей заключается в чередовании свёрточных слоев и субдискретизирующих слоев (слоев подвыборки). Структура сети — однонаправленная (без обратных связей), принципиально многослойная. Для обучения используются стандартные методы, чаще всего метод обратного распространения ошибки. Функция активации нейронов (передаточная функция) — любая, по выбору исследователя [12].

Задача классификации изображений — это приём начального изображения и вывод его класса (кошка, собака и т.д.) или группы вероятных классов, которая лучше всего характеризует изображение. Для людей это один из первых навыков, который они начинают осваивать с рождения [13].

Рисунок 2.1 – Слева – образ, воспринимаемый человеком, справа – образ, представленный в памяти компьютера

Компьютер принимает на вход массив пикселей. Например, размер массива может быть 32x32x3 (где «3» — это количество каналов RGB). Допустим, цветное изображение в формате JPG, и его размер 480x480. Соответствующий массив будет 480x480x3. Каждому из этих чисел присваивается значение от 0 до 255, которое описывает интенсивность пикселя в этой точке.

Сеть может выполнять классификацию изображений через поиск характеристик базового уровня, например, границ и искривлений, а затем с помощью построения более абстрактных концепций через группы свёрточных слоев. Это общее описание работы сверточной нейронной сети.

В перцептроне, который представляет собой полносвязную нейронную сеть, каждый нейрон связан со всеми нейронами предыдущего слоя, причем каждая связь имеет свой персональный весовой коэффициент. В свёрточной нейронной сети в операции свёртки используется лишь ограниченная матрица весов небольшого размера, которую «двигают» по всему обрабатываемому слою (в самом начале — непосредственно по входному изображению), формируя после каждого сдвига сигнал активации для нейрона следующего слоя с аналогичной позицией. То есть для различных нейронов выходного слоя используются одна и та же матрица весов, которую также называют ядром свёртки. Её интерпретируют как графическое кодирование какого-либо признака, например, наличие наклонной линии под определенным углом. Тогда следующий слой, получившийся в результате операции свёртки такой матрицей весов, показывает наличие данного признака в обрабатываемом слое и её координаты, формируя так называемую карту признаков [13].

Естественно, в свёрточной нейронной сети набор весов не один, а целая гамма, кодирующая элементы изображения (например, линии и дуги под разными углами). При этом такие ядра свертки не закладываются исследователем заранее, а формируются самостоятельно путём обучения сети классическим методом обратного распространения ошибки. Проход каждым набором весов формирует свой собственный экземпляр карты признаков, делая

нейронную сеть многоканальной (много независимых карт признаков на одном слое). Также следует отметить, что при переборе слоя матрицей весов её передвигают обычно не на полный шаг (размер этой матрицы), а на небольшое расстояние. Так, например, при размерности матрицы весов 5×5 её сдвигают на один или два нейрона (пикселя) вместо пяти, чтобы не «перешагнуть» искомый признак.

Операция субдискретизации (операция подвыборки или операция объединения), выполняет уменьшение размерности сформированных карт признаков. В данной архитектуре сети считается, что информация о факте наличия искомого признака важнее точного знания его координат, поэтому из нескольких соседних нейронов карты признаков выбирается максимальный и принимается за один нейрон уплотнённой карты признаков меньшей размерности. За счёт данной операции, помимо ускорения дальнейших вычислений, сеть становится более инвариантной к масштабу входного изображения [12].

Преимущества:

– один из лучших алгоритмов по распознаванию и классификации изображений;

– по сравнению с полносвязной нейронной сетью (типа перцептрона) — гораздо меньшее количество настраиваемых весов, так как одно ядро весов используется целиком для всего изображения, вместо того, чтобы делать для каждого пикселя входного изображения свои персональные весовые коэффициенты. Это подталкивает нейросеть при обучении к обобщению демонстрируемой информации, а не попиксельному запоминанию каждой показанной картинки в мириадах весовых коэффициентов, как это делает перцептрон;

– удобное распараллеливание вычислений, а, следовательно, возможность реализации алгоритмов работы и обучения сети на графических процессорах;

– относительная устойчивость к повороту и сдвигу распознаваемого изображения;

– обучение при помощи классического метода обратного распространения ошибки.

К недостатку данной сети можно отнести слишком много варьируемых параметров сети, непонятно, для какой задачи и вычислительной мощности какие нужны настройки. Так, к варьируемым параметрам можно отнести: количество слоев, размерность ядра свёртки для каждого из слоёв, количество ядер для каждого из слоёв, шаг сдвига ядра при обработке слоя, необходимость слоев субдискретизации, степень уменьшения ими размерности, функция по уменьшению размерности (выбор максимума, среднего и т. п.), передаточная функция нейронов, наличие и параметры выходной полносвязной нейросети на выходе свёрточной [12].

2.1 Архитектура свёрточной нейронной сети

Слой свёртки — это основной блок свёрточной нейронной сети. Слой свёртки включает в себя для каждого канала свой фильтр, ядро свёртки которого обрабатывает предыдущий слой по фрагментам (суммируя результаты матричного произведения для каждого фрагмента). Весовые коэффициенты ядра свёртки (небольшой матрицы) неизвестны и устанавливаются в процессе обучения.

Особенностью свёрточного слоя является сравнительно небольшое количество параметров, устанавливаемое при обучении. Так например, если исходное изображение имеет размерность 100×100 пикселей по трём каналам (это значит 30000 входных нейронов), а свёрточный слой использует фильтры с ядром 3x3 пикселя с выходом на 6 каналов, тогда в процессе обучения определяется только 9 весов ядра, однако по всем сочетаниям каналов, то есть 9×3×6 = 162, в таком случае данный слой требует нахождения только 162

параметров, что существенно меньше количества искомых параметров полносвязной нейронной сети.

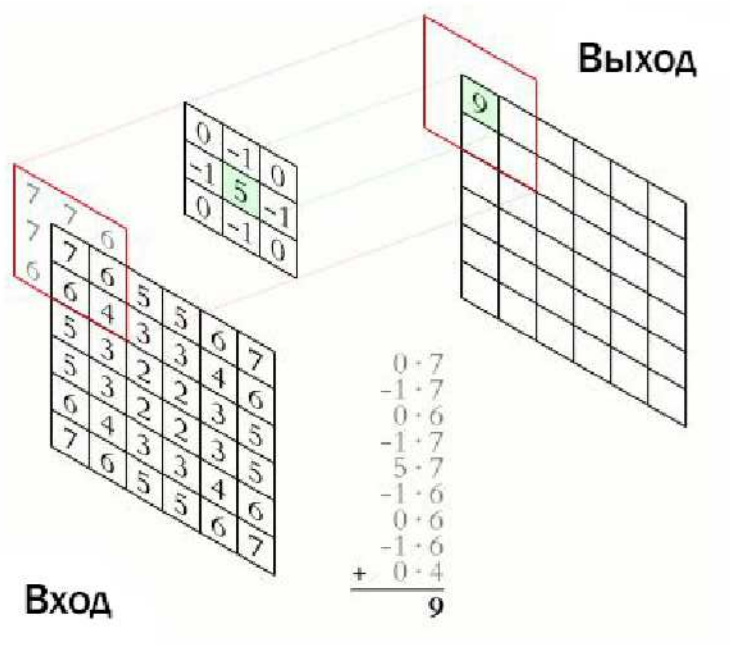


Рисунок 2.2 – Визуальное представление процесса свертки

ReLU — это сокращение от английского англ. rectified linear unit (ReLU), и означает блок линейной ректификации. Слой ReLU — не что иное как функция активации после свёрточного слоя, однако для активации выбирается вместо классических функций типа гиперболического тангенса:

$$f(x) = tanh(x) \tag{33}$$

или сигмоиды:

$$f(x) = \frac{}{ + e^{-1}} \tag{34}$$

ненасыщаемая функция:

$$f(x) = max(0. x). \tag{35}$$

Такая функция показывает хорошие результаты при обучении нейронных сетей и отвечает за отсечение ненужных деталей в канале (при отрицательном выходе).

Слой пулинга (иначе подвыборки, субдискретизации) представляет собой нелинейное уплотнение карты признаков, при этом группа пикселей (обычно размера 2×2) уплотняется до одного пикселя, проходя нелинейное преобразование. Наиболее употребительна при этом функция максимума. Преобразования затрагивают непересекающиеся прямоугольники или квадраты, каждый из которых ужимается в один пиксель, при этом выбирается пиксель, имеющий максимальное значение. Операция пулинга позволяет существенно уменьшить пространственный объём изображения. Пулинг интерпретируется так. Если на предыдущей операции свёртки уже были выявлены некоторые признаки, то для дальнейшей обработки настолько подробное изображение уже не нужно, и оно уплотняется до менее подробного. К тому же фильтрация уже ненужных деталей помогает не переобучаться. Слой пулинга, как правило, вставляется после слоя свёртки перед слоем следующей свёртки.

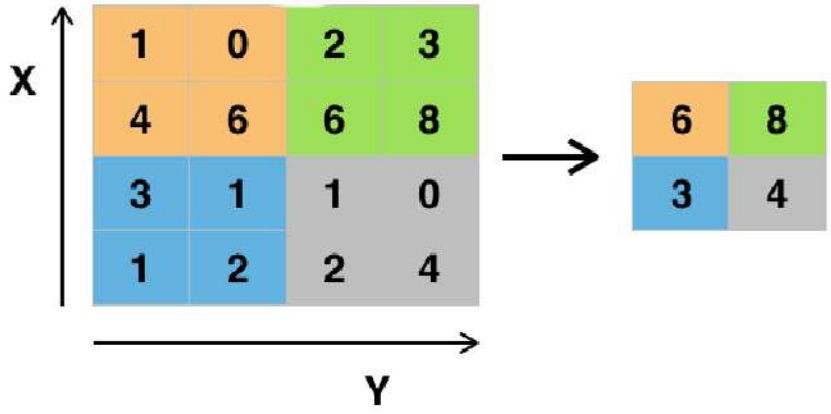


Рисунок 2.3 – Визуализация процесса пулинга (подвыборки или субдискретизации)

Кроме пулинга с функцией максимума можно использовать и другие функции — например, среднего значения или L2-нормирования. Однако практика показала преимущества именно пулинга с функцией максимума, который включается в типовые системы.

После нескольких прохождений свёртки изображения и уплотнения с помощью пулинга на каждом следующем слое увеличивается число каналов и уменьшается размерность изображения в каждом канале. В конце концов остаётся большой набор каналов, хранящих небольшое число данных (даже один параметр), которые интерпретируются как самые абстрактные понятия, выявленные из исходного изображения.

Эти данные объединяются и передаются на обычную полносвязную нейронную сеть, которая тоже может состоять из нескольких слоёв. При этом полносвязные слои уже утрачивают пространственную структуру пикселей и обладают сравнительно небольшой размерностью (по отношению к количеству пикселей исходного изображения) [12].

Схема архитектуры сверточной нейронной сети:

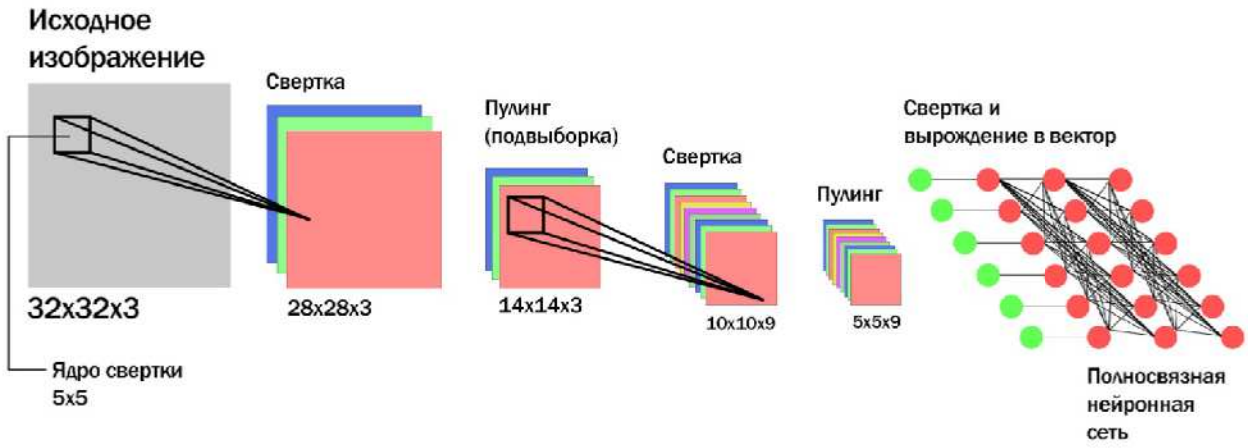


Рисунок 2.4 – Классическая схема архитектуры сверточной нейронной сети

3 Постановка задачи

В рамках данной работы рассматривается задача классификации изображений с помощью сверточной нейронной сети.

Пусть имеется спутниковый снимок земной поверхности.

Произвольно выбирается участок изображения, представляющий собой некоторый географический объект. Необходимо разработать и обучить нейронную сеть, которая позволит определить категорию данного объекта из трёх: водоём, населенный пункт, горный массив.

3.1 Описание метода решения поставленной задачи

Для разрешения задачи классификации изображения географического объекта должна быть построена свёрточная нейронная сеть. Чтобы реализовать решение поставленной проблемы, необходимо выполнить следующие действия:

создание и обучение свёрточной нейронной сети:

– разработать архитектуру сети в соответствии с требованиями поставленной задачи;

– подготовить обучающую выборку данных;

– обучить и протестировать нейронную сеть;

подготовка входного изображения:

– загрузить исходное изображение – спутниковый снимок некоторой области земной поверхности;

– выбрать произвольный участок данного изображения. Этот участок отправить на вход обученной нейронной сети.

4 Выбор инструментария для реализации проекта

В рамках проекта в качестве основного инструмента разработки приложения используется интерпретируемый язык программирования Python, так как для него существуют библиотеки, удобные для быстрой и простой разработки, а именно: библиотека NumPy с открытым исходным кодом, содержащая реализации вычислительных алгоритмов в виде операторов и функций, оптимизированных для работы с многомерными массивами, библиотека для работы с растровой графикой – PIL (Python Image Library), которая поддерживает чтение и запись растровых изображений форматов BMP, JPEG, GIF, PNG, TIFF. Для визуализации работы программы и вывода изображения на экран выбрана библиотека PyOpenGL, предоставляющая высокоуровневый интерфейс для работы с графическим API OpenGL. Для разработки свёрточной нейронной сети применяется фреймворк TensorFlow, а так же библиотека Keras, являющаяся надстройкой над данным фреймворком для оптимизации и упрощения работы.

Используется распространенная и эталонная реализация языка Python – CPython. CPython является интерпретатором байт-кода, написан на C.

Для хранения изображений исходного снимка и входного изображения сети используется расширение JPEG – формат хранения растровых изображений, преимуществом которого являются простота работы с ним и сравнительно небольшой размер.

В качестве среды разработки выступает среда JetBrains PyCharm, которая предоставляет широкий инструментарий для работы с кодом, тестирования и исправления ошибок.

4.1 Разработка программы, реализующей решение поставленной задачи

Программный код реализует следующий алгоритм:

Шаг 1: загружается спутниковый снимок земной поверхности.

Шаг 2: курсором мыши выбирается квадратной участок произвольного размера, который необходимо классифицировать как некоторый географический объект. Максимальный размер выбираемой области ограничен квадратом размера 200x200 пикселей.

Шаг 3: выбранный участок формирует изображение выбранного размера пикселя и сохраняется на диск под именем *input.jpg*.

Шаг 4: полученное изображение масштабируется до размера 64x64 пикселя и подается на вход обученной нейронной сети, которая возвращает ответ в виде вектора классов изображения, элементами которого являются вероятностные значения принадлежности данного географического объекта определенной категории: водоём, населенный пункт, горный массив. Наибольшее значение определяет класс изображения.

Архитектура нейронной сети состоит из трёх сверточных слоёв, содержащих 32, 32, 64 трёхканальных фильтра размера 3x3 соответственно, трёх слоёв субдискретизации, входного слоя полносвязной нейронной сети, формирующей окончательный результат, состоящего из 64 нейронов, выходного слоя из трех нейронов, количество которых соответствует количеству различимых нейронной сетью классов. В качестве функции активации на выходном слое полносвязной сети используется сигмоидная функция, в остальных слоях применяется функция линейной ректификации ReLU.

Обучающая выборка для данной сети сформирована из 360 скриншотов размера 64x64 пикселя (по 120 изображений каждого из трёх классов) географических объектов спутниковых снимков Земли, предоставленных сервисом Google Maps.

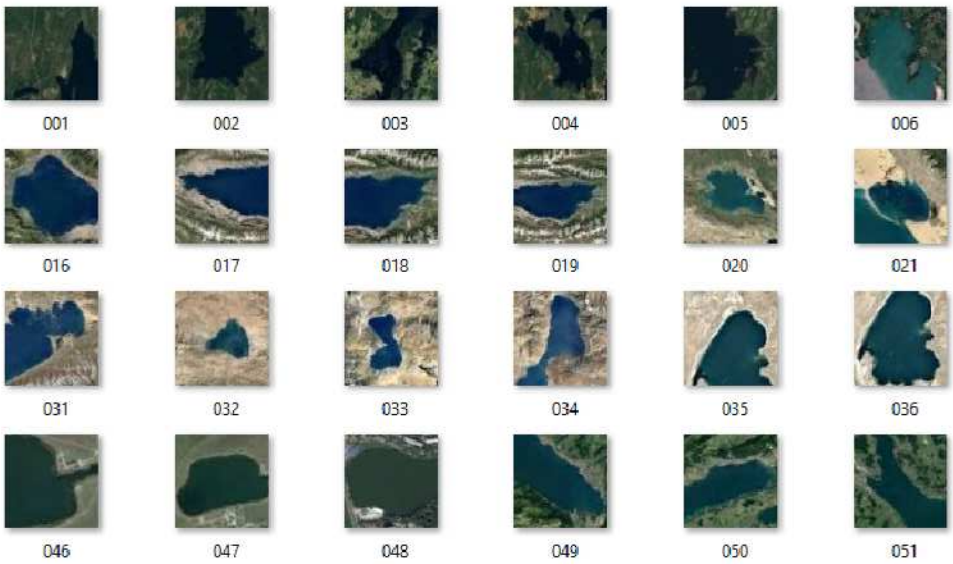


Рисунок 4.1 – Фрагмент набора данных для обучения нейронной сети на примере изображений водоёмов

Обучение сети производится с учителем методом обратного распространения ошибки. Эмпирическим путем определено число эпох обучения равное 18 и коэффициент скорости обучения равный 0,01.

Программа состоит из трех модулей: core, cnn_main, cnn_train. Модуль core содержит в себе функции описания интерфейса и загрузки входных данных:

– reshape() – функция, которая выполняет инициализацию сцены для последующей визуализации;

– display() – функция, отвечающая за визуализацию;

– mouse() – функция обработки нажатий клавиш мыши;

– mouseMove() – обработчик движений мыши; возвращает координаты указателя мыши;

– keys_pressed() – функция, отвечающая за обработку нажатий клавиш клавиатуры.

Модуль cnn_main содержит в себе функцию загрузки изображения и загрузки обученной модели свёрточной нейронной сети:

– load_image(img_path, img_width, img_height) – производит загрузку входного изображения, формируя матрицу значений пикселей, в качестве параметров принимает путь к расположению файла изображения, размеры изображения по горизонтали и вертикали;

– classify_object(img_width, img_height) – непосредственно производит загрузку обученной модели нейронной сети, выполняет анализ и возвращает ответ в удобочитаемом формате.

Модуль cnn_train включает в себя описание архитектуры нейронной сети, начальных значений, функцию загрузки обучающих данных и функцию обучения сети. Данный модуль производит сохранение обученной модели сети в формате HDF.

Диаграмма взаимодействия модулей программы выглядит следующим образом:

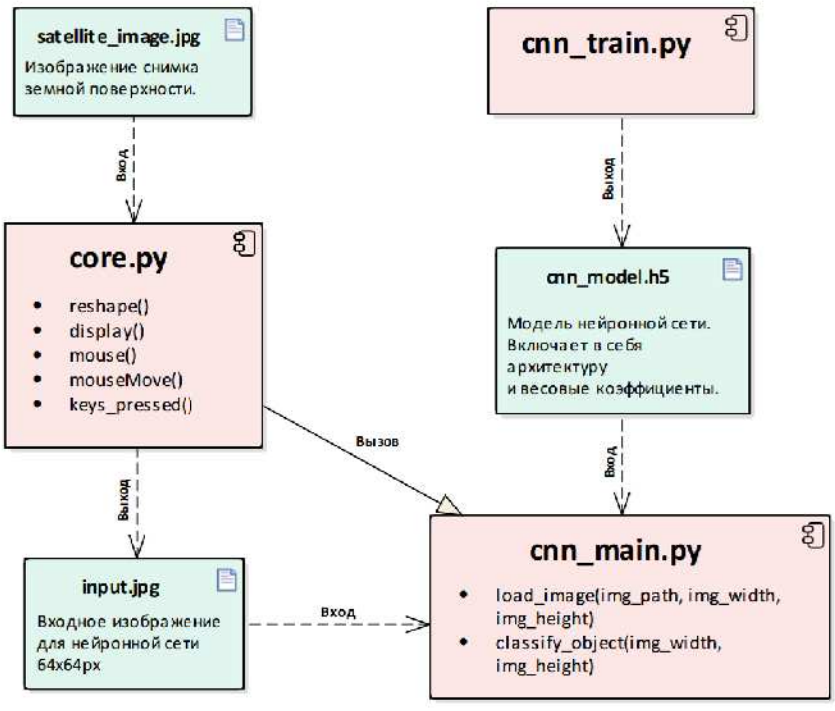


Рисунок 4.2 – Компонентная модель программы

4.2 Демонстрация выполнения разработанного приложения

На вход поступает цветное изображение снимка земной поверхности произвольного масштаба и размерности:

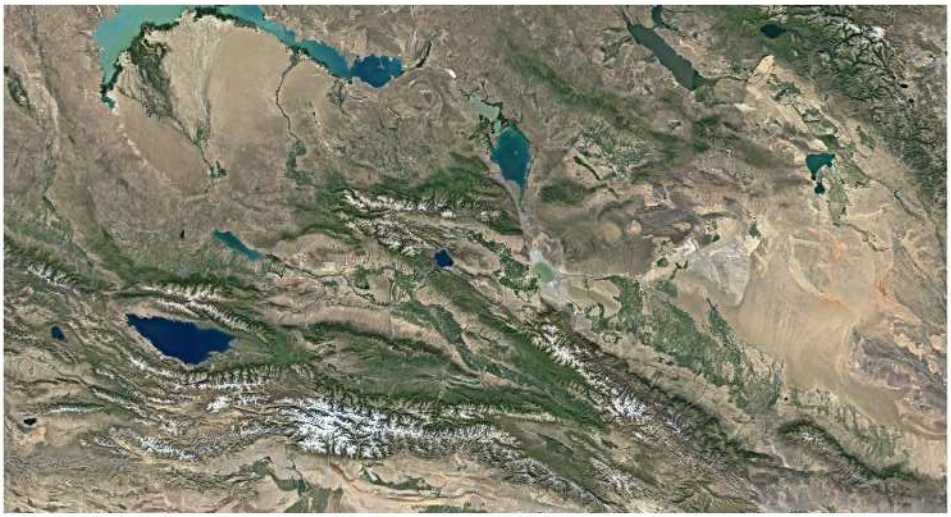


Рисунок 4.3 – Исходное изображение

Затем выбирается квадратный участок произвольного размера, например, область размера 64x64 пикселя, содержащую водоём (помечена красным цветом):

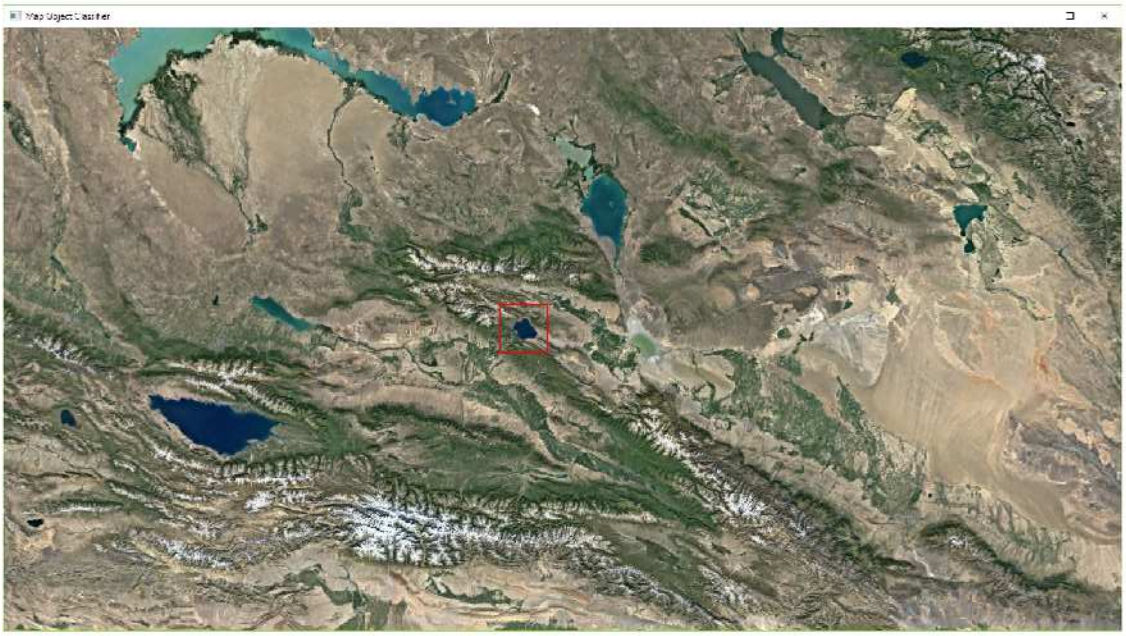


Рисунок 4.4 – Выбор произвольного участка на карте (отмечен красным)

Выбранный участок представляет собой трёхканальное цветное изображение размера 64x64 пикселя и является входным для нейронной сети:

Рисунок 4.5 – Входное изображение свёрточной нейронной сети

Нейронная сеть возвращает ответ в виде вектора вероятностных значений принадлежности объекта одному из трёх классов.

```
Водоём:                                    99.5401%
Населённый пункт:  0.0%
Горная местность:  0.0%

Класс: Водоём
```

Рисунок 4.6 – Результат выполнения программы

Для демонстрации работы алгоритма выбирается участок, принадлежащий данному снимку поверхности, однако он может быть и внешним произвольным изображением географического объекта.

При обучении сети достигнута точность 92%, что является приемлемым, но не хорошим показателем для выбранной архитектуры. Это является результатом небольшого размера обучающей выборки, содержащей 360 примеров. Для более качественного обучения необходим массив данных, включающий, по крайней мере, несколько тысяч изображений.

ЗАКЛЮЧЕНИЕ

В рамках данной работы изучены основные технологии, используемые в искусственных нейронных сетях, архитектура сетей и методы их обучения. Рассмотрены свёрточные нейронные сети, которые позволяют решать задачи идентификации и классификации изображений.

В работе над проектом разработано программное приложение, поддерживающее свёрточные нейронные сети, позволяющее классифицировать объекты географических карт - спутниковых снимков земной поверхности. Созданная в рамках проекта модель нейронной была обучена на предварительно подготовленном наборе обучающих данных и сети протестирована на специально подготовленной тестовой выборке.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Искусственная нейронная сеть. URL: https://ru.wikipedia.org/wiki/Искусственная_нейронная_сеть (дата обращения 10.12.2017)

2 Искусственные нейронные сети (ИНС) - что такое нейросети, как они работают, преимущества и недостатки искусственных нейронов, где используются нейросети. URL: http://www.stevsky.ru/kompiuteri/iskusstvennie-neyronnie-seti-ins-chto-takoe-neyroseti-kak-oni-rabotaiut-preimuschestva-i-nedostatki-iskusstvennich-neyronov-gde-ispolzuiutsya-neyroseti (дата обращения 02.12.2017)

3 Перцептрон. URL: https://ru.wikipedia.org/wiki/Перцептрон (дата обращения 17.11.2017)

4 Перцептрон. URL: http://info-farm.ru/alphabet_index/p/perceptron.html (дата обращения 22.11.2017)

5 Персептроны. Обучение персептронов. URL: http://algmet.simulacrum.me/tai/l3/index.htm (дата обращения 22.11.2017)

6 Нейронные сети: обучение без учителя. URL: http://www.codenet.ru/progr/alg/ai/htm/gl3_4.php (дата обращения 09.12.2017)

7 Обучение методом обратного распространения ошибок. URL: https://studfiles.net/preview/986654/page:11 (дата обращения 09.12.2017)

8 Азбука ИИ. URL: https://nplus1.ru/material/2016/11/04/recurrent-networks (дата обращения 09.12.2017)

9 О рекуррентных нейронных сетях. URL: http://mechanoid.kiev.ua/neural-net-rnn.html (дата обращения 07.12.2017)

10 Нейронная сеть Хопфилда. URL: https://ru.wikipedia.org/wiki/Нейронная_Сеть_Хопфилда (дата обращения 07.12.2017)

11 Нейронная сеть Джордана. URL: https://ru.wikipedia.org/wiki/Нейронная_Сеть_Джордана (дата обращения 09.12.2017)

12 Сверточная нейронная сеть. URL: https://ru.wikipedia.org/Сверточная_нейронная_сеть (дата обращения 10.12.2017)

13 Что такое сверточная нейронная сеть. URL: https://habrahabr.ru/post/309508/ (дата обращения 10.12.2017)