МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

имени академика С.П. КОРОЛЕВА »

Солдатова О. П.

Нейронные сети и машинное обучение

Методические указания для выполнения лабораторных работ и практических заданий

Самара

2024

Оглавление

[ВВЕДЕНИЕ 3](#_Toc145697909)

[Постановка задачи классификации 6](#_Toc145697910)

[Решение задачи классификации 7](#_Toc145697911)

[Метрики оценки качества классификации 10](#_Toc145697912)

[Постановка задачи прогнозирования 13](#_Toc145697913)

[Решение задачи прогнозирования 15](#_Toc145697914)

[Метрики оценки качества прогнозирования 18](#_Toc145697915)

[Задания для практической работы №1 20](#_Toc145697916)

[Задания для практической работы №2 22](#_Toc145697917)

[Задания для лабораторной работы №1 26](#_Toc145697918)

[Задание №1 – максимальная оценка 10 баллов 26](#_Toc145697919)

[Задание №2 – максимальная оценка 15 баллов 28](#_Toc145697920)

[Задание №3 – максимальная оценка 15 баллов 30](#_Toc145697921)

[Задание №4 - максимальная оценка 15 баллов 31](#_Toc145697922)

[Задание №5 - максимальная оценка 15 баллов 33](#_Toc145697923)

[Задание №6 - максимальная оценка 15 баллов 35](#_Toc145697924)

[Задание №7 - максимальная оценка 20 баллов 37](#_Toc145697925)

[Задание №8 - максимальная оценка 25 баллов 39](#_Toc145697926)

[Задание №9 - максимальная оценка 25 баллов 41](#_Toc145697927)

[Задание №10 - максимальная оценка 30 баллов 43](#_Toc145697928)

[Задание №11 - максимальная оценка 30 баллов 45](#_Toc145697929)

[Задание №12 - максимальная оценка 15 баллов 47](#_Toc145697930)

[Задание №13- максимальная оценка 15 баллов 49](#_Toc145697931)

[Задание №14 - максимальная оценка 20 баллов 51](#_Toc145697932)

[Задание №15 - максимальная оценка 20 баллов 53](#_Toc145697933)

[Задание №16 - максимальная оценка 25 баллов 55](#_Toc145697934)

[Задание №17 - максимальная оценка 25 баллов 57](#_Toc145697935)

[Задание №18- максимальная оценка 30 баллов 59](#_Toc145697936)

[Задание №19 - максимальная оценка 30 баллов 61](#_Toc145697937)

[Задание №20 - максимальная оценка 30 баллов 63](#_Toc145697938)

[Задания для лабораторной работы №2 65](#_Toc145697939)

[Примеры оформления отчётов для лабораторных работ 69](#_Toc145697940)

ВВЕДЕНИЕ

Целью выполнения практических заданий и лабораторных работ по курсу «Нейронные сети и машинное обучение» является получение практических навыков разработки моделей машинного обучения и методов анализа данных для решения задач распознавания, диагностики и прогнозирования с целью поддержки процесса принятия решений в интеллектуальных системах.

По курсу «Нейронные сети и машинное обучение» магистранты выполняют 2 практических задания и 2 лабораторные работы.

Практическое задание 1. Тема работы: «Исследование эффективности моделей машинного обучения для решения задачи классификации на примере решения задачи классификации трёх различных сортов итальянских вин».

В качестве моделей машинного обучения, в зависимости от варианта, необходимо реализовать две из следующих пяти моделей: логистическая регрессия, наивный байесовский классификатор, метод «k-ближайших соседей», линейный дискриминант Фишера, случайный лес с использованием библиотеки Scikit-learn и языка программирования Python.

Исходные данные являются специальными данными библиотеки Scikit-learn.

На выполнение работы отводится 8 часов.

Практическое задание 2. Тема работы: «Исследование эффективности алгоритмов обучения и топологии нейронной сети на примере решения задачи классификации изображений в градации серого цвета на наборе изображений рукописных цифр MNIST»

В качестве модели машинного обучения необходимо реализовать модель многослойного персептрона с использованием библиотек Keras и Tensorflow и языка Python.

Исходные данные доступны по ссылке: https://www.kaggle.com/datasets/hojjatk/mnist-datase.

На выполнение работы отводится 8 часов.

ЛР 1. Тема работы: «Решение задачи классификации и прогнозирования при помощи моделей «мелких» нейронных сетей».

В качестве модели машинного обучения необходимо реализовать одну из моделей нейронной сетей в соответствии с заданием без использования специализированных библиотек для разработки нейронных сетей. Язык программирования выбирает магистрант.

Ссылки на примеры исходных данных приведены в заданиях на выполнение лабораторной работы №1.

На выполнение работы отводится 8 часов.

ЛР 2. Тема работы: «Решение задачи классификации изображений на примере решения задачи распознавания дефектов поверхности металлов на изображениях в оттенках серого цвета (определение одного из 6 типов дефектов)».

В качестве модели машинного обучения необходимо реализовать модель свёрточной нейронной сети с использованием библиотек Keras и Tensorflow и языка Python.

Исходные данные доступны по ссылке: https://www.kaggle.com/datasets/fantacher/neu-metal-surface-defects-data.

На выполнение работы отводится 8 часов.

Практические задания и лабораторные работы выполняются индивидуально.

Номер вариантов заданий на практические задания 1 и 2, а также на лабораторную работу 2 соответствуют номеру фамилии магистранта в списке группы.

Задания на 1 лабораторную работу магистрант выбирает по согласованию с преподавателем. Степень сложности разработки той или иной нейросетевой модели отражается в максимальном количестве баллов, которые может получить магистрант за выполнение лабораторной работы.

Результаты исследований оформляются в виде отчёта по лабораторной работе. Примеры оформления отчётов приведены в приложении к данным методическим указаниям.

Исходные данные для обучения и тестирования разработанных программ для лабораторной работы 1 рекомендуется брать из UCI Machine Learning Repository или с сайта Kaggle.

Постановка задачи классификации

При решении задачи классификации необходимо отнести имеющиеся статические образцы к определенным классам. Возможно несколько способов представления данных. Наиболее распространенным является способ, при котором образец представляется вектором. Компоненты этого вектора представляют собой различные характеристики образца, которые влияют на принятие решения о том, к какому классу можно отнести данный образец.

Таким образом, на основании некоторой информации о примере, необходимо определить, к какому классу его можно отнести. Классификатор относит объект к одному из классов в соответствии с определенным разбиением N-мерного пространства, которое называется пространством входов, и размерность этого пространства является числом компонент вектора.

В реальных задачах часто возникает ситуация, когда число образцов ограничено, что затрудняет определение сложности задачи. Можно выделить три основных уровня сложности. Первый – когда классы можно разделить прямыми линиями (или гиперплоскостями) – так называемая линейная разделимость. Во втором случае классы невозможно разделить линиями (плоскостями), но их можно отделить с помощью более сложного деления – нелинейная разделимость. В третьем случае классы пересекаются и можно говорить только о вероятностной разделимости. Варианты разделения классов представлены на рисунке 1.



Рисунок 1 – Варианты разделения классов

Решение задачи классификации

Можно сказать, что задача классификации заключается в идентификации объекта и отнесении его к одному из нескольких множеств. Для решения задачи классификации с линейной и нелинейной разделимостью классов используются классические модели нейронных сетей: многослойный персептрон и рекуррентные сети на его основе, радиально-базисные сети, сети Кохонена, гибридные сети, рекуррентные самоорганизующиеся сети.

Если же предположить, что множества объектов накладываются друг на друга, то задача классификации становится более общей и предполагает, что объект характеризуется степенью принадлежности к тому или иному множеству, то есть имеет место задача классификации с вероятностной разделимостью классов. При решении задачи классификации в такой постановке часто используются методы нечетких множеств, а также нечеткие нейронные сети: нечёткий многослойный персептрон, сеть Ванга-Менделя, сеть TSK.

Для решения задачи классификации изображений используются свёрточные сети.

Применительно к нейронным сетям задачу классификации можно поставить следующим образом: пусть имеется N множеств D1,D2…Dn признаков объектов. Сеть обучается на парах векторов **X** и **Y**, где:

**X** = (x1,x2…xm) – входной вектор признаков;

**Y** = (y1,y2…yn) = C(**X**) – выходной вектор, классифицирующий вектор X. При этом возможно несколько случаев:

1. **Y** = k, классификатор имеет скалярный характер. k – порядковый номер множества, к которому относится **X**.
2. **Y** = (y1,y2 …yk … yn). При этом только yk=1, остальные компоненты вектора равны 0. Таким образом работает звено Кохонена
3. **Y** = (y1,y2 … yn). При этом каждая компонента yk характеризует степень принадлежности к множеству Dk.

В режиме нормального функционирования сеть по входному вектору X выдает вектор Z по правилам, аналогичным описанным для векторов Y. Точность решения определяется статистикой: сколько раз вектор Z правильно классифицировал объект с признаками X, соотнося его с той или иной группой Dk. Для пункта 3 возможно вычисление погрешности, при наличии функции-скаляризатора степени принадлежности вектора X к множествам Dk. Задача может быть дополнена введением «шума», однако смысл от этого не изменится. Шум лишь изменит границы областей D1…Dn.

С другой стороны, сами области D1…Dn можно рассматривать как набор векторов эталонных признаков объекта E1,E2 … En для отнесения к тому, или иному классу. При этом каждый из векторов может задаваться с той или иной степенью шума, что в конечном итоге и сформирует\* области Dk. Последнее определение имеет смысл использовать для генерации выборок.

При решении задачи классификации входные элементы сети используются в качестве приёмников классификационных признаков. Таким образом, число входов сети однозначно определяется числом характеристик объекта классификации в решаемой задаче.

В задачах классификации число классов, как и число признаков, также известно априори. Однако для переноса значения класса принадлежности на структуру нейронной сети возможно несколько подходов. Один из них заключается в присваивании классам принадлежности уникальных числовых значений (в самом простом случае — порядковых номеров) и использования единственного выхода сети.

Другой подход — использование числа выходов, соответствующего числу классов принадлежности в решаемой задаче. В этом случае, для каждого из выходов возможно всего два эталонных значения — относится или не относится набор входных признаков соответствующему классу. Логично в качестве таких значений использовать числа 0 и 1, однако возможны и любые другие значения.

Преимущество первого подхода — в его простоте. Каким бы ни было число классов в решаемой задаче, структура сети остаётся неизменной и содержит лишь один выход. Это также положительно влияет на скорость обучения. Однако для единственного выходного значения может оказаться проблематичным толкование принадлежности к конкретному классу в задачах, где набор признаков может содержать в себе черты нескольких классов. Данное ограничение можно считать основным недостатком этого метода.

Вариант с множеством выходных значений этого недостатка лишён — на каждом выходе получаем значение принадлежности к каждому из возможных классов. Поэтому такой подход является наиболее гибким.

Обучающие выборки должны включать от 20 до 50 обучающих примеров для каждого класса. Тестирующие выборки должны составлять примерно 15-20% от объёма всех данных.

Подача на вход сети одной обучающей выборки – это одна эпоха обучения. Процесс обучения сети можно многократно повторять, используя одну и ту же обучающую выборку, число эпох, необходимых для обучения нейронной сети зависит от конкретной задачи и нейросетевой модели и может варьироваться от нескольких десятков до нескольких десятков тысяч.

При обучении рекуррентных самоорганизующихся сетей в качестве обучающей выборки выступает набор обучающих образцов, по одному на каждый класс. Обучающая выборка подаётся однократно, и расчёт весов также производится один раз, после чего сеть является обученной. Тестирующую выборку следует создавать по нескольку образцов для каждого класса.

Для определения качества обучения и тестирования сети необходимо использовать метрики оценки качества. Метрики для определения качества классификации приведены в следующем разделе.

Метрики оценки качества классификации

Пусть, существует «основной класс». Обычно, это класс, при обнаружении которого, предпринимается какое-либо действие; например, при постановке диагноза основным классом будет «болен», а вторичным классом «здоров».

Измерения ошибки, как оценки «вероятности выдать неверный ответ», может быть не всегда достаточно:

10% ошибки при постановке диагноза может означать, как и то что, 10 % больных будут признаны здоровыми, так и то, что 10% здоровых больными.

При неравнозначности ошибок для разных классов вводят понятие ошибки первого и второго рода и замеряют их по отдельности.

Ошибка первого рода равна вероятности принятия основного класса за вторичный, то есть это вероятность «промаха», когда искомый объект будет пропущен.

Ошибка второго рода равна вероятности принятия вторичного класса за основной, то есть это вероятность «ложной тревоги».

Пусть есть два класса и алгоритм, который предсказывает принадлежность объекта одному из классов, тогда матрица ошибок классификации описывается таблицей 1.

Таблица 1 – Матрица ошибок классификации

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | True Positive (TP) | False Positive (FP) |
|  | False Negative (FN) | True Negative (TN) |

Здесь представляет собой ответ алгоритма на объекте, а является истинным классом объекта.

Доля правильных ответов алгоритма среди общего количества ответов часто задается в виде меры аккуратности:

Особенность данной метрики заключается в том, что она не учитывает случай с несбалансированными данными в выборке. Если распределение тестов в выборке будет смещено в сторону какого-либо класса, в рамках этих классов алгоритм будет принимать большую часть правильных решений, а на некоторых он не сможет классифицировать и трети образцов. Доля правильных ответов не рассчитывается с учетом знаний о сбалансированности классов и будет демонстрировать не соответствующее фактической ситуации высокое значение. В таком случае следует либо использовать сбалансированную выборку данных, либо изменить подход к оценке качества классификации.

Для оценки в пределах класса используются такие метрики, как точность и полнота. Точность показывает долю объектов, верно рассчитанную алгоритмом классификации среди всех, рассчитанных им как положительные:

Полнота показывает способность алгоритма находить данный класс вообще, и описывается формулой:

Точность и полнота применимы даже в условиях несбалансированных выборок. На основании этих оценок можно рассчитать их взвешенное среднее.

Recall демонстрирует способность алгоритма обнаруживать данный класс вообще, а precision — способность отличать этот класс от других классов.

FPR – показывает уровень неверных позитивных ответов:

TPR – показывает уровень верных позитивных ответов:

F-мера — среднее гармоническое precision и recall.

β в данном случае определяет вес точности в метрике.

F-мера достигает максимума при полноте и точности, равными единице, и близка к нулю, если один из аргументов близок к нулю.

В случае задач с несбалансированными классами, которые превалируют в реальной практике, часто приходится прибегать к техникам искусственной модификации выборок для выравнивания соотношения классов.

Постановка задачи прогнозирования

Прогнозирование – это процесс выявления общих закономерностей развития, установления тенденций этого развития, вероятностных, количественных и качественных сдвигов, определение наиболее эффективных направлений воздействия на систему.

Прогноз представляет собой суждение о состоянии какого-либо объекта, процесса или явления к определенному моменту времени в будущем. Любой прогноз имеет вероятностный характер, т.е. точное предсказание значений предсказываемых показателей невозможно. Прогноз может быть представлен в виде точечной или интервальной оценки, нескольких вариантов развития событий, но в любом случае необходимо оценить точность прогноза или, иначе, ожидаемую ошибку прогнозирования.

Традиционно выделяют краткосрочные, среднесрочные и долгосрочные прогнозы. В зависимости от конкретного объекта прогнозирования длительности прогнозов для каждого типа могут различаться, однако, обычно краткосрочным является прогноз на 1-3 шага, среднесрочным – на 4-6 шагов, от 7 до 10 шагов прогнозы считаются долгосрочными. Величина шага прогноза соответствует шагу дискретизации прогнозируемого временного ряда (например, 1 день).

Прогнозирование – это предсказание будущих событий. Целью прогнозирования является уменьшение риска при принятии решений. В большинстве случаев прогноз получается ошибочным, причем ошибка зависит от прогнозирующей системы и методов прогнозирования. Для уменьшения ошибки следует увеличивать количество отсчётов временного ряда, предоставляемых для прогноза. Основной проблемой прогнозирования является выявление неточности прогноза. Обычно, решение, принимаемое на основании прогноза должно учитывать ошибку, о которой сообщает система прогнозирования. Таким образом, система прогнозирования должна обеспечить определение прогноза и ошибки прогнозирования.

Наиболее распространенными задачами прогнозирования, на сегодняшний день являются задачи метеопрогноза и задачи прогнозирования в экономике и на предприятиях, а в частности финансовое планирование, планирование технологического процесса, планирование фондового рынка и т.п.

Большинство задач прогнозирования можно свести к предсказанию временного ряда. Предсказание временного ряда сводится к типовой задаче аппроксимации функции многих переменных по заданному набору примеров – с помощью процедуры погружения ряда в многомерное пространство. Например, http://articles.mql4.com/c/articles/2008/01/d_4.png-мерное лаговое пространство ряда состоит из http://articles.mql4.com/c/articles/2008/01/d_5.pngзначений ряда в последовательные моменты времени:

.

Для динамических систем доказана следующая теорема Такенса: Если временной ряд порождается динамической системой, т.е. значения есть произвольная функция состояния такой системы, существует такая глубина погружения (примерно равная эффективному числу степеней свободы данной динамической системы), которая обеспечивает однозначное предсказание следующего значения временного ряда. Таким образом, выбрав достаточно большое , можно гарантировать однозначную зависимость будущего значения ряда от его предыдущих значений:

(2)

т.е. предсказание временного ряда сводится к задаче интерполяции функции многих переменных.

Решение задачи прогнозирования

Способности нейронной сети к прогнозированию напрямую следуют из ее способности к обобщению и выделению скрытых зависимостей между входными и выходными данными. После обучения сеть способна предсказать будущее значение некой последовательности на основе нескольких предыдущих значений и/или каких-то существующих в настоящий момент факторов.

Прогнозирование временных рядов с помощью нейронных сетей объединяет достоинства экстраполяционного и адаптивного прогнозирования.

Экстраполяционное прогнозирование сводится к типовой задаче нейроанализа – аппроксимации функции многих переменных по заданному набору примеров – с помощью процедуры погружения ряда в многомерное пространство, т.е. предсказание временного ряда сводится к задаче интерполяции функции многих переменных. Нейросеть далее можно использовать для восстановления этой неизвестной функции по набору примеров, заданных историей данного временного ряда.

Адаптивные методы предполагают, что параметры модели временного ряда пересчитываются при поступлении каждого нового значения, благодаря чему модель адаптируется к исходным данным. Этот процесс аналогичен обучению нейронной сети в режиме онлайн, но нейронная сеть представляет собой куда более сложную модель, чем те, что обычно используются в адаптивном прогнозировании.

Для прогнозирования на нейронных сетях, как правило, необходимо не менее 100 наблюдений для создания приемлемой модели. Зачастую в экономике такое количество исторических данных недоступно, что ограничивает применение нейронных сетей для прогнозирования экономической динамики. Однако возможно построение удовлетворительной модели на нейронных сетях даже в условиях нехватки данных, многократно повторяя обучения нейронной сети на одних и тех же обучающих данных. Кроме того, модель может уточняться по мере того, как свежие данные становится доступными.

Обобщающие способности нейронной сети позволяют предсказывать будущую ситуацию, даже если она не повторяет в точности какой-либо исторический момент. В экономике нередки случаи коренного изменения свойств временного ряда, когда исторические данные несопоставимы с новыми (например, в результате проведения реформ, изменения методики сбора данных, территориальных преобразований). При этом ни одна из традиционных экстраполяционных или адаптивных моделей не может использовать ретроданные, в то время как нейронная сеть зачастую может быть переучена и накопленная ранее информация не будет полностью потеряна.

Также при использовании традиционных экстраполяционных методов возникает неформализованная задача выбора вида экстраполяционной функции. За счет этого вносится дополнительная априорная информация и при правильном выборе вида модели повышается качество прогноза. Однако неверный выбор приводит к формированию в корне ошибочного прогноза даже при высокой точности описания исходных данных.

При использовании нейронной сети такой задачи не возникает, она заменяется задачей формирования архитектуры сети. Эта задача, хотя и не является строго формализованной, все же имеет несколько известных методик решения, в то время как выбор вида экстраполяционной модели целиком основывается на опыте эксперта или сводится к простому перебору всех доступных моделей.

Для обучения нейронной сети необходимо разделить выборку на обучающую и тестирующую, причем в обучающую должно попасть не менее 80% векторов. При обучении подсчитываются погрешности обучения и тестирования сети.

Для получения пар входных и выходных векторов к исходному ряду значений применяется так называемый метод «скользящих окон». Метод скользящих окон подразумевает использование двух окон: входного и выходного с заданными размерами и соответственно. Размер первого окна определяется числом значений величины, на основе которых строится прогноз, а размер второго — количеством прогнозируемых значений, то есть тем, на сколько шагов вперёд происходит прогноз. На уровне нейронной сети размер первого окна определяет число входов сети, а размер второго окна — число выходов. Перемещаясь по ряду значений с некоторым шагом, окна и формируют пары из вектора исходных значений и вектора эталонных спрогнозированных значений соответственно. Полученная на каждом шаге пара используется как элемент набора обучающих или тестовых данных.

Для решения задачи прогнозирования можно использовать как обычные нейронные сети, так и нечёткие нейронные сети.

Метрики оценки качества прогнозирования

Для определения качества обучения и тестирования сети необходимо подсчитывать погрешности обучения и тестирования соответственно.

В качестве погрешности обучения и тестирования можно рассчитывать MSE, MAE, RMSE, MAPE: –фактическое значение временного ряда, – прогнозное значение.

Средняя квадратичная ошибка MSE, один из наиболее распространённых показателей для оценки регрессионных моделей, MSE измеряет среднеквадратичную ошибку прогнозов. Для каждой точки вычисляется квадратная разница между прогнозами и целью, а затем усредняются эти значения

.

Чем выше это значение, тем хуже модель. Ошибка никогда не бывает отрицательной, для идеальной модели это будет ноль.

Среднеквадратическая ошибка RMSE, это квадратный корень из MSE.

RMSE служит для объединения величин ошибок в прогнозах для различных точек данных в единую меру прогноза. RMSE — это мера точности, позволяющая сравнивать ошибки прогнозирования различных моделей для конкретного набора данных, а не между наборами данных, поскольку она зависит от масштаба. RMSE всегда неотрицательна, и значение 0 указывает на идеальное соответствие данным.

Средняя абсолютная ошибка MAE это линейная оценка, показывающая что все индивидуальные различия взвешены одинаково в среднем

.

Данная оценка является более робастной чем MSE.

Средняя абсолютная ошибка MAPE – это та же ошибка MAE выраженная в процентах. Данная оценка применяется для временных рядов, фактические значения которых значительно больше 1. Для каждого объекта абсолютная ошибка делится на целевое значение, что дает относительную ошибку.

Считается, что точность модели хорошая, если среднее значение относительной погрешности не превышает 5%, удовлетворительная, если среднее значение относительной погрешности не превышает 15%, и неудовлетворительная, если среднее значение относительной погрешности больше 15%.

Задания для практической работы №1

Целью данной лабораторной работы является демонстрация способности моделей машинного обучения решать задачи классификации. Необходимо реализовать модель на языке Python в среде Jupyter Notebook с использованием библиотек Numpy, Pandas, Matplotlib, Sclearn.

Используемые модели машинного обучения приведены в таблице 1.

Порядок выполнения работы:

1. Считать файл с данными для классификации итальянских вин.
2. Построить диаграмму рассеивания для заданных классов вин.
3. Построить классификаторы для заданной пары классов итальянских вин в соответствии с вариантом задания. Обучить и протестировать классификаторы.
4. Рассчитать метрики accuracy, precision, recall, TPN, FPN, F1.
5. Построить ROC кривые и посчитать площадь AUC под кривой.
6. Рассчитать и построить графическое изображение для Confusion matrics.
7. Сравнить метрики и ROC-кривые. Сделать вывод о выборе лучшего классификатора по критерию F-меры и площади AUC. В качестве ответа в текстовое поле написать наилучший классификатор по каждому из критериев.
8. Продемонстрировать работу моделей машинного обучения преподавателю.
9. Составить отчёт, который должен содержать постановку задачи, текст программы, результаты работы программы по пунктам 2-8 и сдать его преподавателю.

Варианты заданий приведены в таблице 1.

Таблица 1 – Варианты заданий для выполнения практической работы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Класс 1 | Класс 2 | Классификатор 1 | Классификатор 2 |
| 1 | 0 | 1 | К-ближайших соседей (К=3) | Случайный лес |
| 2 | 0 | 2 | К-ближайших соседей (К=3) | Случайный лес |
| 3 | 1 | 2 | К-ближайших соседей (К=3) | Случайный лес |
| 4 | 0 | 1 | Логистическая регрессия | Наивный байесовский классификатор |
| 5 | 0 | 2 | Логистическая регрессия | Наивный байесовский классификатор |
| 6 | 1 | 2 | Логистическая регрессия | Наивный байесовский классификатор |
| 7 | 0 | 1 | Наивный байесовский классификатор | Линейный дискриминант Фишера |
| 8 | 0 | 2 | Наивный байесовский классификатор | Линейный дискриминант Фишера |
| 9 | 1 | 2 | Наивный байесовский классификатор | Линейный дискриминант Фишера |
| 10 | 0 | 1 | Линейный дискриминант Фишера | Логистическая регрессия |
| 11 | 0 | 2 | Линейный дискриминант Фишера | Логистическая регрессия |
| 12 | 1 | 2 | Линейный дискриминант Фишера | Логистическая регрессия |
| 13 | 0 | 1 | Случайный лес | Логистическая регрессия |
| 14 | 0 | 2 | Случайный лес | Логистическая регрессия |
| 15 | 1 | 2 | Случайный лес | Логистическая регрессия |
| 16 | 0 | 1 | Наивный байесовский классификатор | К-ближайших соседей (К=3) |
| 17 | 0 | 2 | Наивный байесовский классификатор | К-ближайших соседей (К=3) |
| 18 | 1 | 2 | Наивный байесовский классификатор | К-ближайших соседей (К=3) |
| 19 | 0 | 1 | Линейный дискриминант Фишера | К-ближайших соседей (К=3) |
| 20 | 0 | 2 | Линейный дискриминант Фишера | К-ближайших соседей (К=3) |
| 21 | 1 | 2 | Линейный дискриминант Фишера | К-ближайших соседей (К=3) |
| 22 | 0 | 1 | Логистическая регрессия | К-ближайших соседей (К=3) |
| 23 | 0 | 2 | Логистическая регрессия | К-ближайших соседей (К=3) |
| 24 | 1 | 2 | Логистическая регрессия | К-ближайших соседей (К=3) |
| 25 | 0 | 1 | Наивный байесовский классификатор | Случайный лес |
| 26 | 0 | 2 | Наивный байесовский классификатор | Случайный лес |
| 27 | 1 | 2 | Наивный байесовский классификатор | Случайный лес |
| 28 | 0 | 1 | Линейный дискриминант Фишера | Случайный лес |
| 29 | 0 | 2 | Линейный дискриминант Фишера | Случайный лес |
| 30 | 1 | 2 | Линейный дискриминант Фишера | Случайный лес |

Задания для практической работы №2

Целью данной лабораторной работы является демонстрация способности нейронной сети решать задачи классификации. Необходимо реализовать модель на языке Python в среде Jupyter Notebook с использованием библиотек Numpy, Pandas, Matplotlib, Sclearn, Keras и Tensorflow.

Используемая модель: многослойный персептрон.

Порядок выполнения работы:

1. Считать файл с данными набора MNIST.
2. Написать программу, имитирующую работу многослойного персептрона с настраиваемым числом скрытых слоёв от 1 до 2 в соответствии с вариантом.
3. Провести обучение сети с использованием заданных алгоритмов обучения SGD, SGD с импульсом (моментом) Нестерова, RMSProp , Adam , Adamax, Nadam , Adafactor, [AdamW](https://keras.io/api/optimizers/adamw/).
4. Провести тестирование сети, обученной при помощи всех алгоритмов.
5. Продемонстрировать работу многослойного персептрона преподавателю.
6. Исследовать зависимость функции ошибки при обучении и классификации, а также метрику качества accuracy от числа скрытых нейронов в 1 скрытом слое для заданных алгоритмов.
7. Исследовать зависимость функции ошибки при обучении и тестировании, а также метрики качества accuracy от числа эпох обучения для заданных алгоритмов.
8. Исследовать зависимость функции ошибки при обучении и тестировании, а также метрики качества accuracy от уровня исключения нейронов (dropout изменять в диапазоне 0.2-0.5) для заданных алгоритмов.
9. Построить графики функции ошибки при обучении и тестировании а также метрики качества accuracy для всех исследований.
10. Рассчитать метрики precision, recall, TPN, FPN, F1.
11. Рассчитать и построить графическое изображение для Confusion matrics.
12. Сделать вывод о лучшем алгоритме обучения и лучших параметрах топологии сети и обучения.
13. Составить отчёт, который должен содержать постановку задачи, текст программы, результаты исследований по пунктам 5-13 и сдать его преподавателю.

Варианты заданий приведены в таблице 2.

Таблица 2 – Варианты заданий для практической работы 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| \*\*№ | Алгоритм 1 | Алгоритм 2 | Число скрытых слоёв топология 1 | Число скрытых слоёв топология 2 | Функция активации | Функция ошибки |
| 1 | SGD | Nadam | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 2 | SGD с импульсом Нестерова | Adam | 1 | 2 | ELU | categorical\_crossentropy |
| 3 | Adafactor | RMSProp | 1 | 2 | ReLU | mean\_squared\_error |
| 4 | [AdamW](https://keras.io/api/optimizers/adamw/) | Adamax | 1 | 2 | LReLU | mean\_absolute\_error |
| 5 | SGD | RMSProp | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 6 | SGD с импульсом Нестерова | Adamax | 1 | 2 | ELU | categorical\_crossentropy |
| 7 | Adafactor | Nadam | 1 | 2 | ReLU | mean\_squared\_error |
| 8 | [AdamW](https://keras.io/api/optimizers/adamw/) | Adam | 1 | 2 | LReLU | mean\_absolute\_error |
| 9 | SGD | Adam | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 10 | SGD с импульсом Нестерова | Nadam | 1 | 2 | ELU | categorical\_crossentropy |
| 11 | Adafactor | Adamax | 1 | 2 | ReLU | mean\_squared\_error |
| 12 | [AdamW](https://keras.io/api/optimizers/adamw/) | RMSProp | 2 | 2 | LReLU | mean\_absolute\_error |
| 13 | SGD | Adamax | 1 | 2 | ELU | categorical\_crossentropy |
| 14 | SGD с импульсом Нестерова | RMSProp | 1 | 2 | ReLU | mean\_squared\_error |
| 15 | Adafactor | [AdamW](https://keras.io/api/optimizers/adamw/) | 1 | 2 | LReLU | mean\_absolute\_error |
| 16 | [AdamW](https://keras.io/api/optimizers/adamw/) | Adafactor | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 17 | SGD | Adafactor | 1 | 2 | ReLU | mean\_squared\_error |

Продолжение таблицы 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 18 | SGD с импульсом Нестерова | Adafactor | 1 | 2 | LReLU | mean\_absolute\_error |
| 19 | RMSProp | Nadam | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 20 | Adam | Nadam | 1 | 2 | ELU | categorical\_crossentropy |
| 21 | SGD | [AdamW](https://keras.io/api/optimizers/adamw/) | 1 | 2 | LReLU | categorical\_crossentropy |
| 22 | SGD с импульсом Нестерова | [AdamW](https://keras.io/api/optimizers/adamw/) | 1 | 2 | Логистическая сигмоида | mean\_squared\_error |
| 23 | Adam | [AdamW](https://keras.io/api/optimizers/adamw/) | 1 | 2 | ELU | mean\_absolute\_error |
| 24 | Nadam | RMSProp | 2 | 2 | ReLU | sparse categorical crossentropy |
| 25 | RMSProp | Adam | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 26 | Nadam | Adamax | 1 | 2 | ELU | categorical\_crossentropy |
| 27 | Adafactor | SGD | 1 | 2 | Логистическая сигмоида | mean\_squared\_error |
| 28 | Nadam | [AdamW](https://keras.io/api/optimizers/adamw/) | 1 | 2 | LReLU | mean\_absolute\_error |
| 29 | RMSProp | SGD с импульсом Нестерова | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 30 | RMSProp | Adamax | 1 | 2 | ELU | categorical\_crossentropy |
| 31 | Adam | Adafactor | 1 | 2 | ReLU | mean\_squared\_error |
| 32 | [AdamW](https://keras.io/api/optimizers/adamw/) | SGD с импульсом Нестерова | 1 | 2 | LReLU | mean\_absolute\_error |
| 33 | Adam | SGD | 1 | 2 | Логистическая сигмоида | sparse categorical crossentropy |
| 34 | Nadam | Adam | 1 | 2 | ELU | categorical\_crossentropy |
| 35 | Adam | RMSProp | 1 | 2 | ReLU | mean\_squared\_error |
| 36 | RMSProp | [AdamW](https://keras.io/api/optimizers/adamw/) | 1 | 2 | LReLU | mean\_absolute\_error |

Примеры оформления отчётов для лабораторных работ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»

(Самарский университет)

Институт информатики и кибернетики

Кафедра информационных систем и технологий

Отчет по лабораторной работе №2

по курсу

«Интеллектуальные системы управления»

Выполнил:

студент группы

6132

Иванов И.И.

Проверил:

к.т.н., доцент Солдатова О.П.

Самара 2023

Целью данной лабораторной работы является демонстрация способности нейронной сети решать задачи классификации. Реализовать модель на языке Python в среде Jupyter Notebook с использованием библиотек Numpy, Pandas, Matplotlib, Sclearn, Keras и Tensorflow.

Используемая модель: многослойный персептрон. Вариант №5.

Порядок выполнения работы:

1. Считать файл с данными для классификации итальянских вин.
2. Написать программу, имитирующую работу многослойного персептрона с настраиваемым числом скрытых слоёв от 1 до 3 в соответствии с вариантом.
3. Провести обучение сети с использованием заданных алгоритмов обучения SGD, SGD с импульсом (моментом) Нестерова, Adagrad , Adadelta, RMSProp , Adam , Adamas, Nadam .
4. Провести тестирование сети, обученной при помощи заданных алгоритмов.
5. Продемонстрировать работу многослойного персептрона преподавателю.
6. Исследовать зависимость функции ошибки при обучении и тестировании, а также метрику качества accuracy от объёма обучающей выборки для всех алгоритмов.
7. Исследовать зависимость функции ошибки при обучении и тестировании, а также метрики качества accuracy от числа эпох обучения для заданных алгоритмов.
8. Построить графики функции ошибки при обучении и тестировании, а также метрики качества accuracy для обучения и тестирования.
9. Рассчитать метрики precision, recall, TPN, FPN, F1.
10. Построить ROC кривые и посчитать площадь AUC под кривой.
11. Сделать вывод о лучшем алгоритме обучения и лучших параметрах топологии сети и обучения.
12. Рассчитать и построить графическое изображение для Confusion matrics.
13. Составить отчёт, который должен содержать постановку задачи, текст программы, результаты исследований по пунктам 5-12 и сдать его преподавателю.

Описание сети

В данной работе для решения задачи классификации используется сеть на основе многослойного персептрона.

Многослойный персептрон состоит из множества входных узлов, которые образуют входной слой; одного или нескольких скрытых слоев вычислительных нейронов и одного выходного слоя. Входной сигнал распространяется по сети в прямом направлении от слоя к слою. Многослойные персептроны успешно применяются для решения разнообразных сложных задач. При этом обучение с учителем выполняется с помощью такого популярного алгоритма, как алгоритм обратного распространения ошибки.

Многослойный персептрон имеет три отличительных признака:

Каждый нейрон имеет нелинейную функцию активации. Данная функция должна быть гладкой (то есть всюду дифференцируемой). Самой популярной гладкой функцией активации является сигмоидальная функция.

Сеть содержит один или несколько слоев скрытых нейронов. Эти нейроны позволяют сети обучаться решению сложных задач, последовательно извлекая наиболее важные признаки из входного вектора.

Сеть обладает высокой степенью связности, реализуемой посредством синаптических соединений.

Структура многослойного персептрона с одним скрытым слоем изображена на рисунке 2.

Файл с данными для классификации итальянских вин

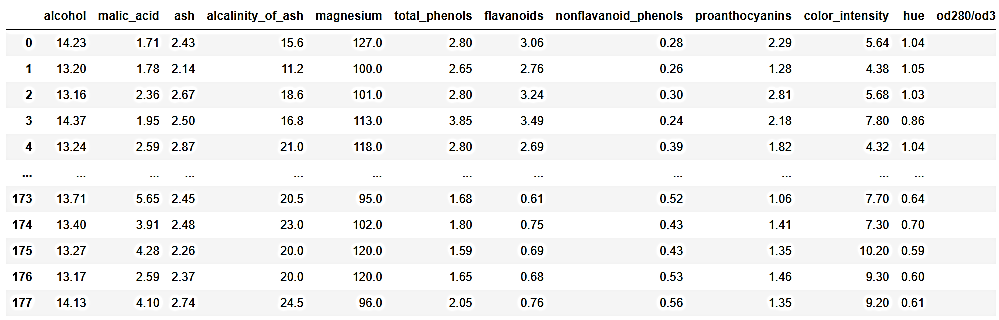
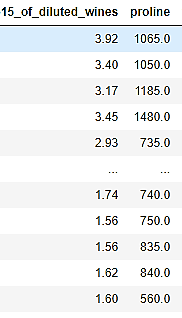
 

Рисунок 1 – Пример данных датасета.

Многослойный персептрон

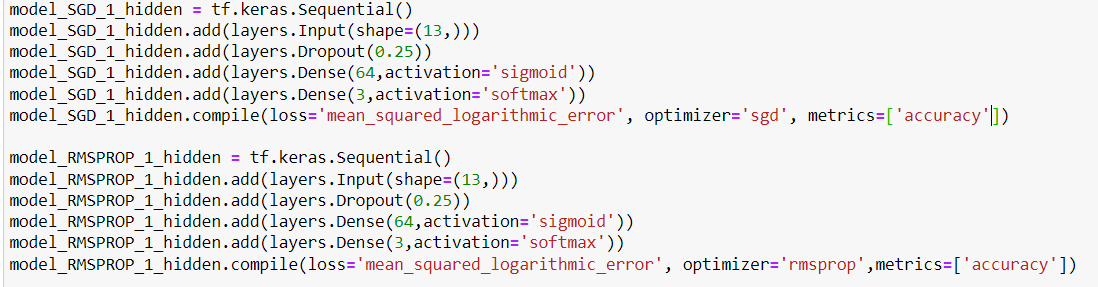


Рисунок 2 – Структура многослойного персептрона для алгоритмов SGD и RMSPROP

Структура многослойного персептрона с тремя скрытыми слоями изображена на рисунке 3

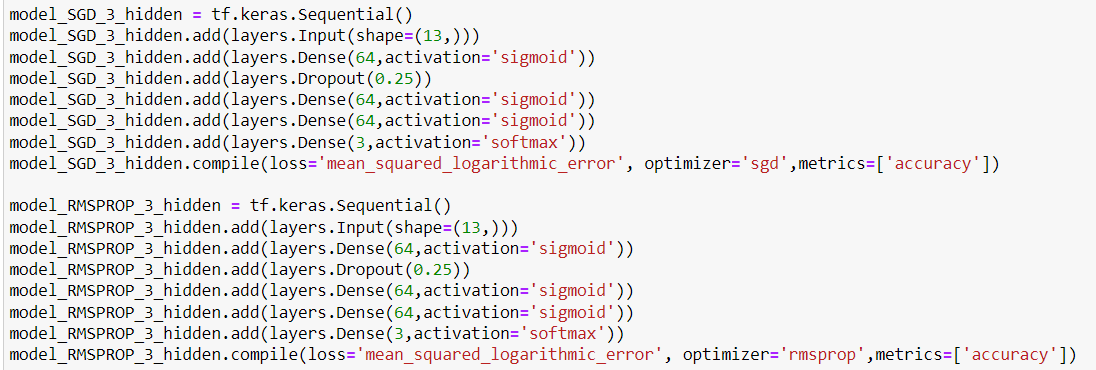


Рисунок 3 – Структура многослойного персептрона для алгоритмов SGD и RMSPROP.

В программе реализованы 2 модели многослойного персептрона.

Первая модель включает входной слой, 1 скрытый полносвязный слой. В качестве функции активации каждого слоя используется «sigmoid», для последнего слоя выбрана функция «softmax». Выбраны такие параметры: размер батча – 1, количество эпох обучения – 50, доля обучающей выборки из общей выборки = 20%. В качестве метрики погрешности прогнозирования взято accuracy, precision, recall, TPN, FPN, F1.

Вторая модель включает входной слой, 3 скрытых полносвязных слоя. В качестве функции активации каждого слоя используется «sigmoid», для последнего слоя выбрана функция «softmax». Выбраны такие параметры: размер батча – 1, количество эпох обучения – 50, доля обучающей выборки из общей выборки = 20%. В качестве метрики погрешности прогнозирования взято accuracy, precision, recall, TPN, FPN, F1.

При обучение использованы алгоритмы SGD, RMSPROP.

Описание предметной области

В данной работе использовался набор данных для классификации вин с 3 классами.

Пример данных на рисунке 3.

Исследование модели1 и модели2 многослойного персептрона

Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качества accuracy от числа эпох обучения для всех алгоритмов.

На рисунках 4 – 7 изображены графики зависимости погрешности и метрики качества accuracy от числа эпох.

Метрика accuracy при тестировании работы нейросети для алгоритма SGD с 1 слоем составила около 100% и алгоритма SGD с 3 слоями около 90%, RMSPROP с 1 слоем составила 100% и алгоритма RMSPROP с 3 слоями около 99%.

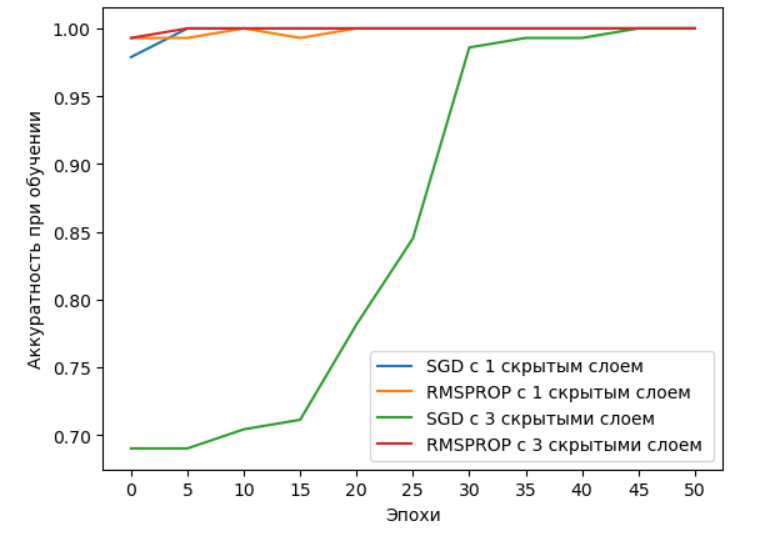


Рисунок 4 - Зависимость метрики accuracy при обучении по алгоритмам от количества эпох обучения.

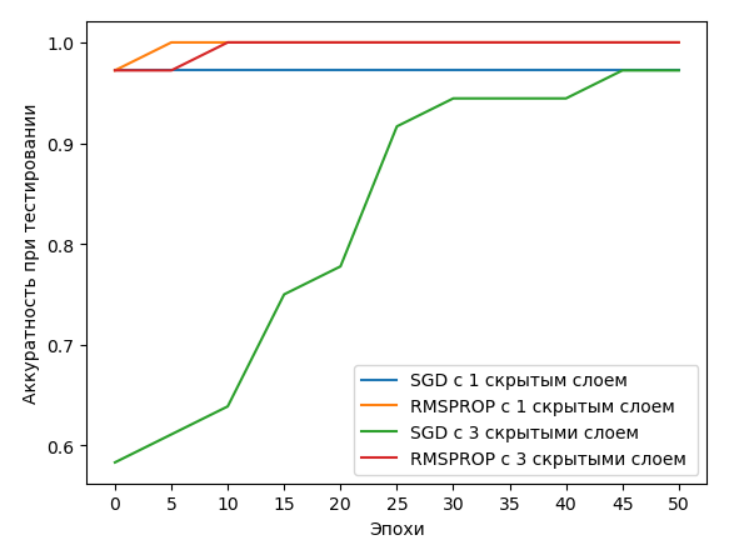
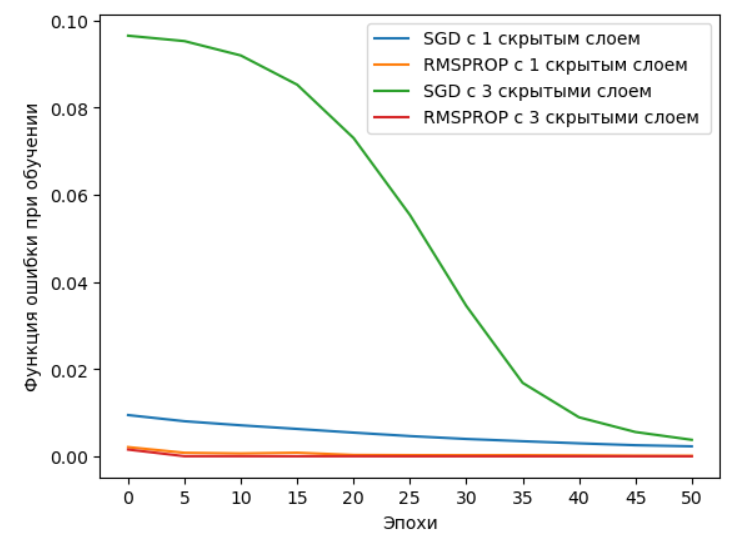
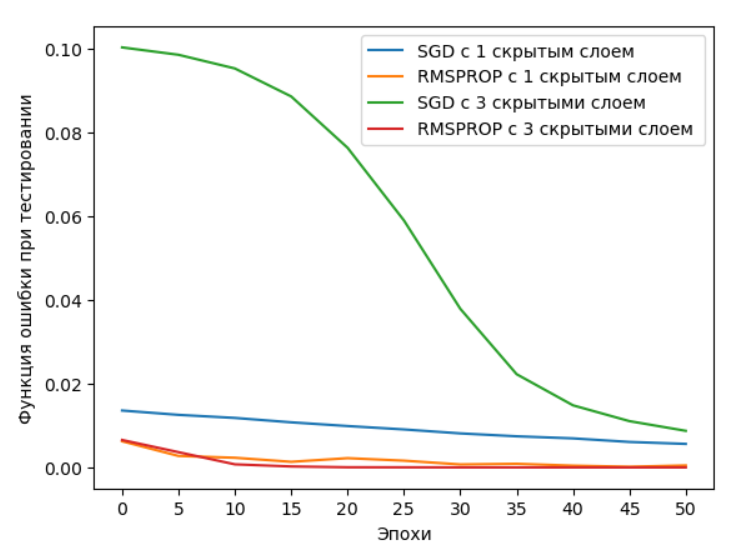


Рисунок 5 – Зависимость метрики accuracy при классификации по алгоритмам от количества эпох обучения.

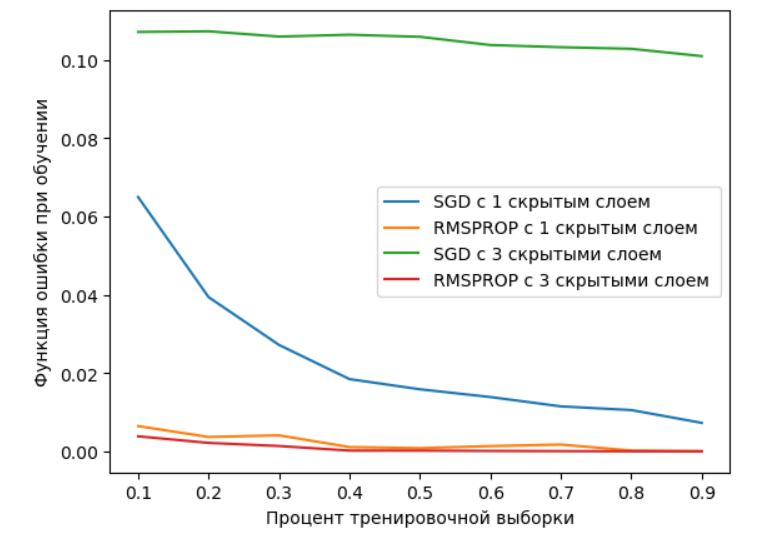
Рисунок 6 – Зависимость погрешности при обучении от количества эпох.

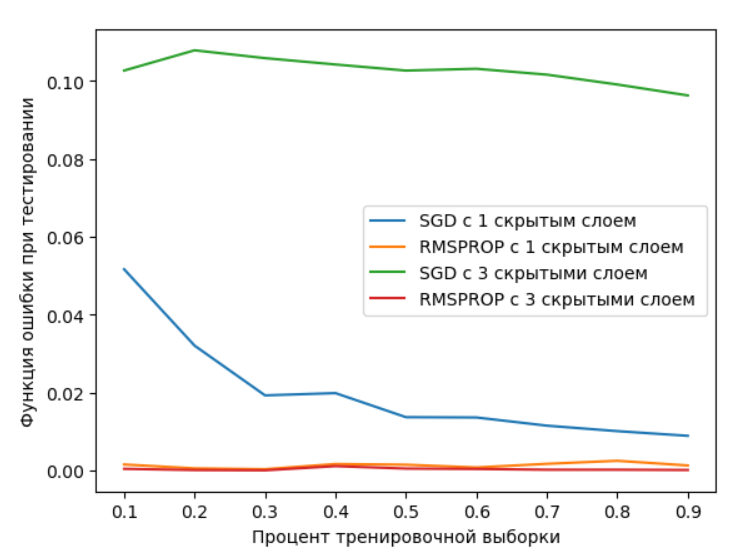
Рисунок 7 – Зависимость погрешности при классификации от количества эпох.

Исследование функции погрешности при обучении и классификации, а также метрики качества accuracy для всех исследований.

Дополнительно было проведено исследование погрешности прогнозирования от способа разделения обучающей выборки на две части: обучающую и тестирующую.

Результаты экспериментов приведены ниже. По оси X находится доля выборки, используемая для обучения, от 10% до 90%, по оси Y – погрешность прогнозирования.

Рисунок 8 – Зависимость погрешности при обучении от разделения обучающей выборки на 2 части: обучающую и тестирующую.

Рисунок 9 – Зависимость погрешности при классификации от разделения обучающей выборки на 2 части: обучающую и тестирующую.

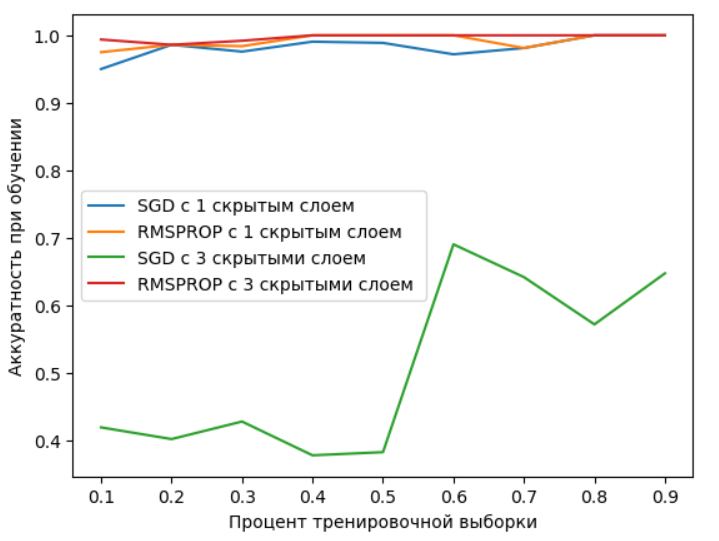


Рисунок 10 – Зависимость метрики accuracy при обучении от разделения обучающей выборки на 2 части: обучающую и тестирующую.

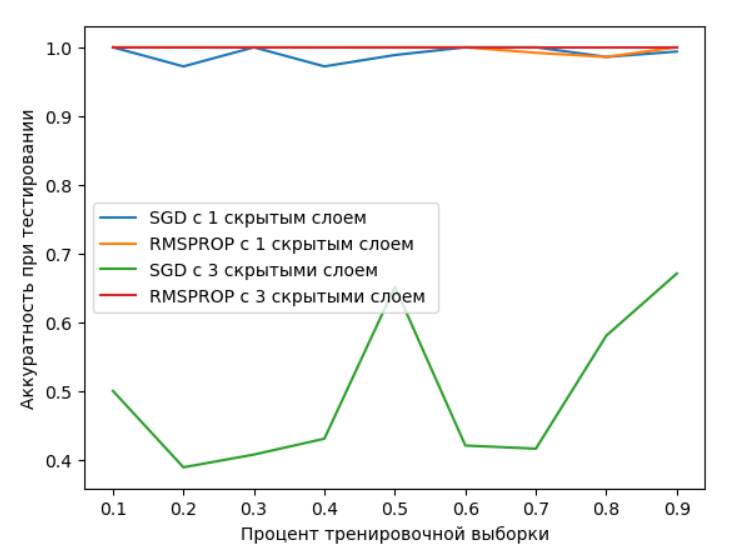


Рисунок 11 - Зависимость метрики accuracy при классификации от разделения обучающей выборки на 2 части: обучающую и тестирующую.

ROC-кривые и дополнительные метрики

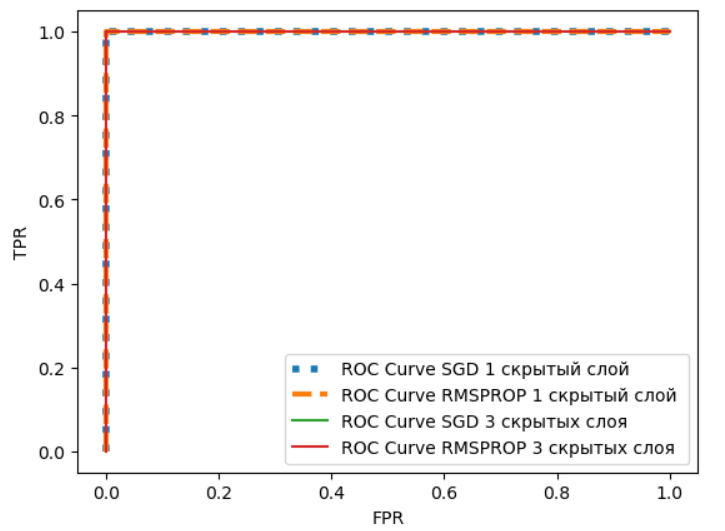


Рисунок 12 – Зависимость TPR от FPR

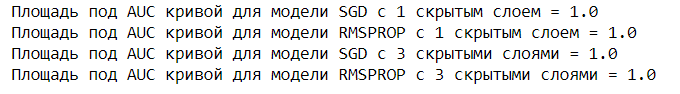
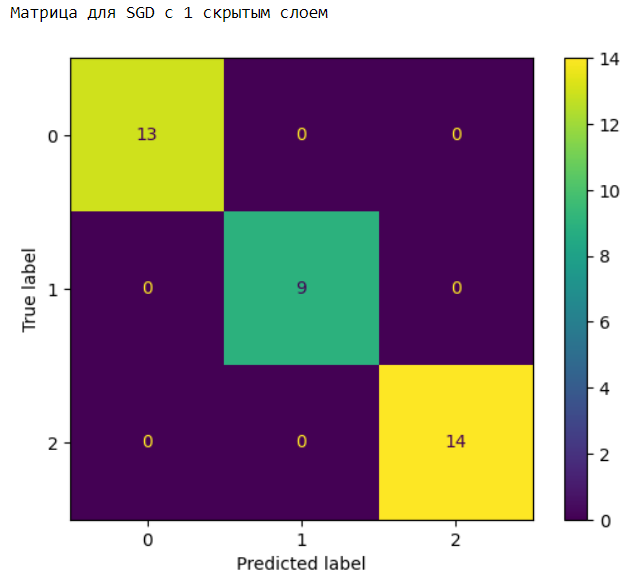
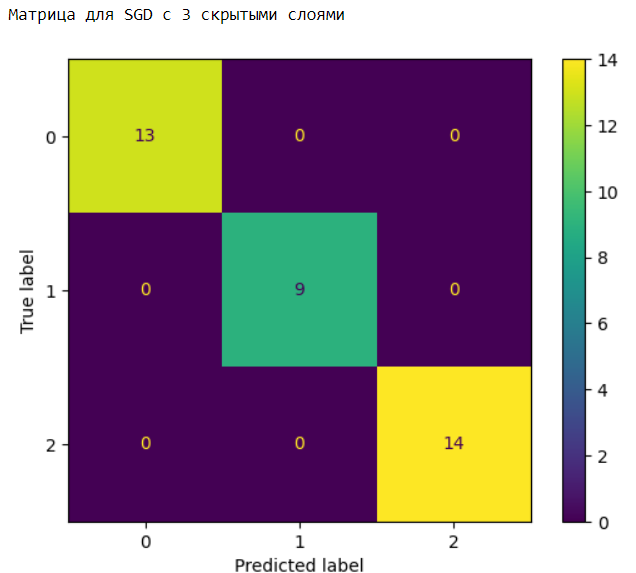
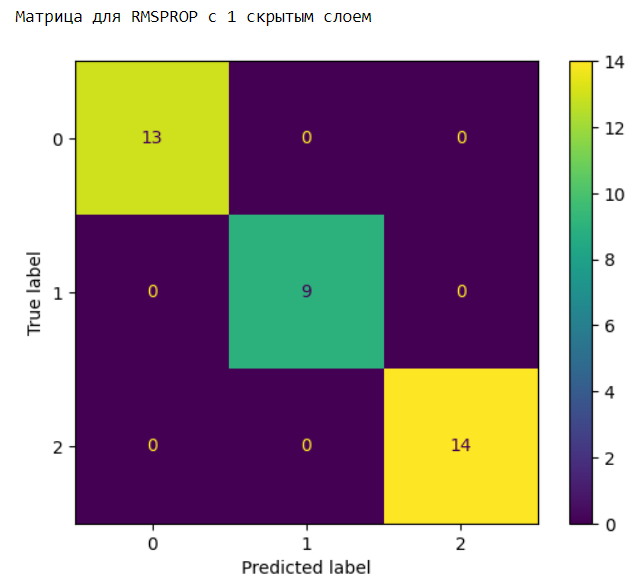
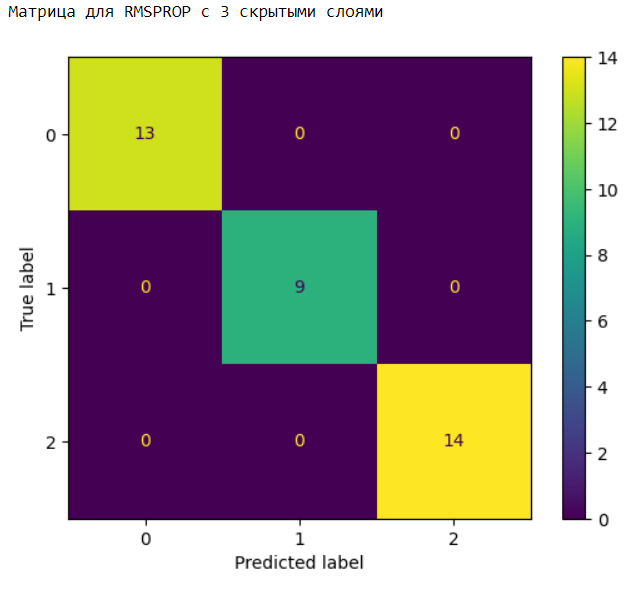


Рисунок 13 – Площади под ROC-кривыми

Рисунок 14 – Confusion matrix для SGD с 1 слоем

Рисунок 15 – Confusion matrix для SGD с 3 слоями

Рисунок 16 – Confusion matrix для RMSPROP с 1 слоем

Рисунок 17 – Confusion matrix для RMSPROP с 3 слоями

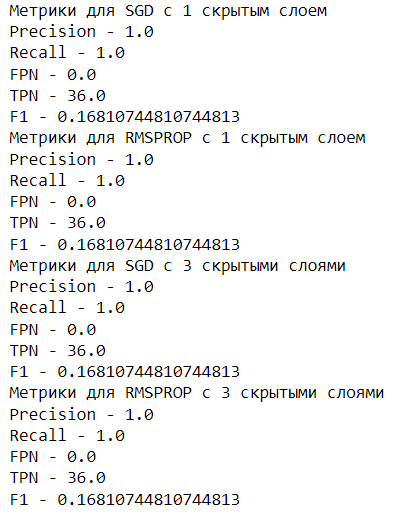


Рисунок 18 – Метрики Precision, Recall, FPN, TPN, F1.

Зависимость функции ошибки и метрики accuracy от размера Dropout

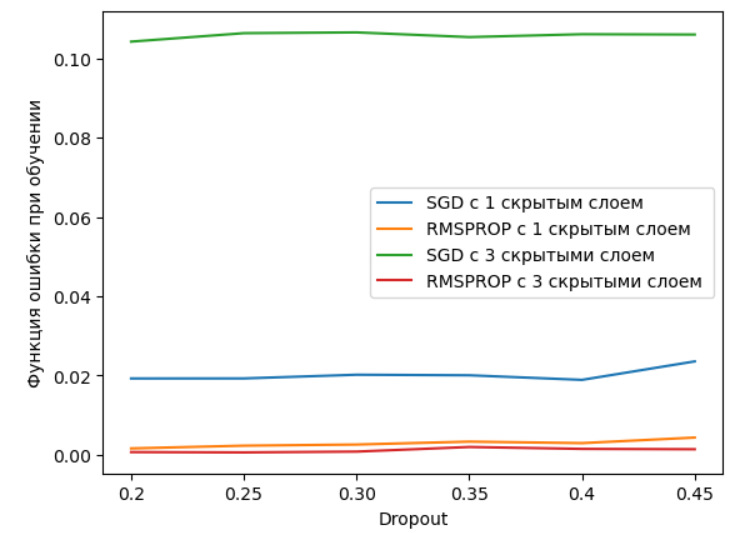


Рисунок 19 – Функция ошибки от Dropout при обучении

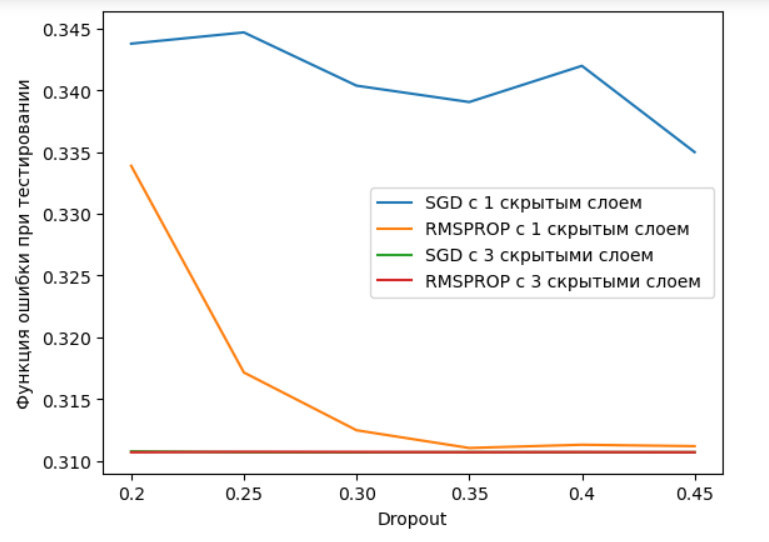
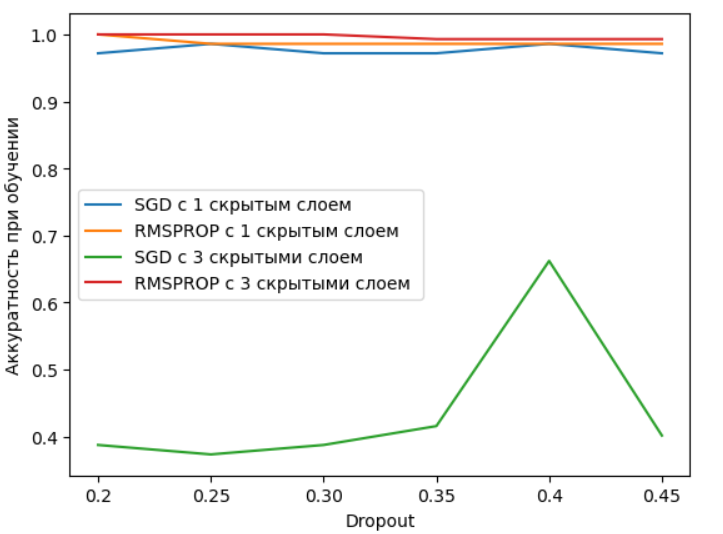


Рисунок 20 – Функция ошибки от Dropout при классификации

Рисунок 21 – Метрика accuracy от Dropout при обучении

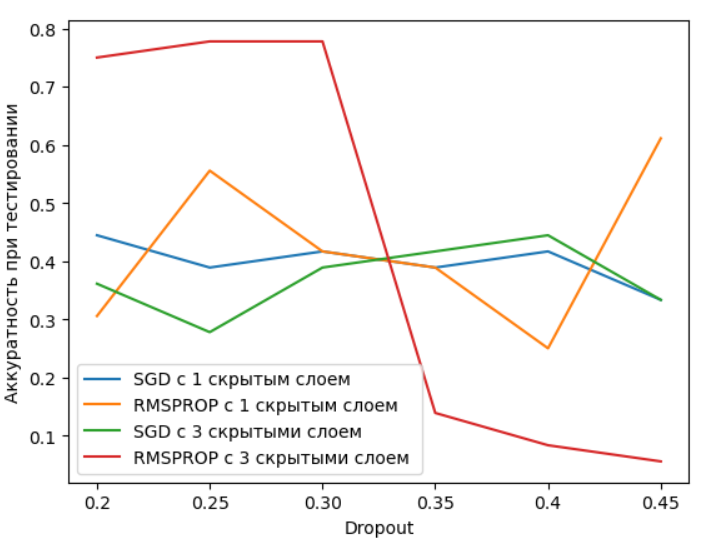


Рисунок 22 – Метрика accuracy от Dropout при классификации

Сравнительный анализ лучших параметров двух моделей многослойного персептрона

Выводы

По результатам исследований можно сделать вывод, что многослойный персептрон хорошо подходит для задач бинарной классификации и классификации с малым количеством классов. Для правильного выбора параметров сети важно проводить исследования и выбирать их оптимальные соотношения. Самым эффективным алгоритмом обучения на выбранной структуре сети является RMSPROP, оптимальным разделением является разделение: 80% на обучение, 20% на проверку. Результаты исследований приведены в таблице 1.

Оптимальными значениями Dropout является 0,35.

Таблица 1 – Результаты исследований двух моделей многослойного персептрона

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Алгоритм обучения 1 - SGD | | | | | |
|  | Оптимальный объём обучающей выборки | Оптимальное число эпох обучения | Максимальное значение accuracy при обучении | Максимальное значение accuracy при тестировании | Минимальное значение функции ошибки при обучении | Минимальное значение функции ошибки при тестировании |
| Модель 1  1 слой | 20% | 5 | 1 | 0.97 | 0.01 | 0.01 |
| Модель 2  3 слоя | 55% | 45 | 1 | 0.97 | 0.11 | 0.012 |
|  | Алгоритм обучения 2 - RMSPROP | | | | | |
|  | Оптимальный объём обучающей выборки | Оптимальное число эпох обучения | Максимальное значение accuracy при обучении | Максимальное значение accuracy при тестировании | Минимальное значение функции ошибки при обучении | Минимальное значение функции ошибки при тестировании |
| Модель 1  1 слой | 20% | 10 | 1 | 1 | 0.0001 | 0.0001 |
| Модель 2  3 слоя | 20% | 10 | 1 | 1 | 0.00001 | 0.00001 |



ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

Отчет по лабораторной работе №4  
по курсу  
«Интеллектуальные системы управления»

Выполнил: Иванов И.И., группа 6132

Проверил:  
к.т.н., доцент Солдатова О.П.

САМАРА 2023

Цель работы

Целью данной лабораторной работы является демонстрация способности нейронной сети решать задачи классификации при неполных или недостоверных данных. Сеть необходимо обучить классификации по пяти классам по 10 ‑ 20 количественным или качественным признакам.

Используемая модель

Многослойный персептрон

Модель многослойного персептрона представлена на рисунке 1.

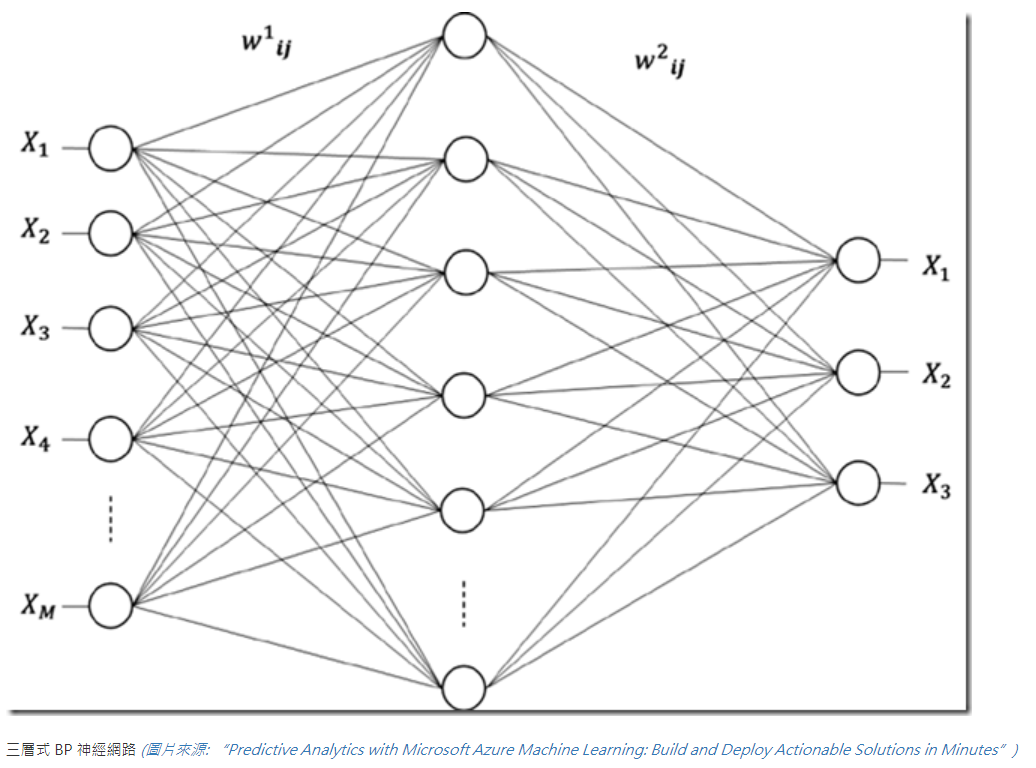


Рисунок 1 – Модель многослойного персептрона

Порядок выполнения работы

Данная лабораторная работа выполняется в несколько этапов:

1. Необходимо выбрать предметную область, отобрать числовые характеристики объектов и задать их диапазоны. Примеры типов предметных областей для классификации перечислены ниже.
2. Сгенерировать обучающую выборку размерностью от 30 до 50 примеров для каждого класса.
3. Написать программу, имитирующую работу многослойного персептрона и провести обучение сети по методу наискорейшего спуска с использованием алгоритма обратного распространения ошибки. Выбрать начальное значение числа скрытых слоёв, равным единице, число нейронов в скрытом слое – равным полусумме входов и выходов сети.
4. Провести обучение многослойного персептрона по алгоритму наискорейшего спуска с учётом моментов (1 занятие).
5. Продемонстрировать работу многослойного персептрона преподавателю (2 занятие).
6. Исследовать зависимость погрешности обучения для каждого алгоритма от значения коэффициента обучения.
7. Исследовать зависимость погрешности обучения и классификации для каждого алгоритма от объёма обучающей выборки.
8. Исследовать зависимость погрешности обучения и классификации для каждого алгоритма от числа нейронов скрытого слоя.
9. Исследовать зависимость погрешности классификации от числа итераций обучения.
10. Составить отчёт, который должен содержать постановку задачи, обучающие выборки, результаты исследований по пунктам 6 - 9 и сдать его преподавателю (3 занятие).

Предметная область

В качестве предметной области была выбрана классификация графических образов букв.Таким образом, нейронная сеть должна уметь распознавать и классифицировать образы черных латинских букв на белом фоне.

В качестве обучаемой выборки были выбраны 26 печатный латинских букв и 26 арабских латинских букв.

Пример латинских букв печатных и арабских печатных представлены на рисунке 2 и 3 соответственно.



Рисунок 2 – Печатная латинская буква



Рисунок 3 – Арабская печатная латинская буква

Обучающие выборки

В качестве обучающей выборки были заготовлены файлы .txt со списком пикселей латинских печатных и арабских букв. Перед обучением загружаются эти файлы и на основе данных выборок производится обучение нейронной сети.

Обучение нейронной сети

В качестве функции активации была выбрана сигмоидальная функция или сигмоид [2]:

|  |  |
| --- | --- |
|  | (1) |

В качестве алгоритма обучения были использованы методы обратного распространения ошибки (BackPROP), RPROP и iRPROP+.

1. Метод обратного распространения ошибки

Обратное распространение ошибки – это метод вычисления градиента, который используется при обновлении весов многослойного персептрона.

Цели обратного распространения просты:

* Отрегулировать каждый вес пропорционально тому, насколько он способствует общей ошибке;
* В результате итеративного уменьшения ошибки каждого веса получить ряд весов, которые дают хорошие результаты распознавания.

В основе алгоритма обратного распространения ошибки лежит применение метода градиентного спуска для поиска глобального минимума целевой функции [2]:

|  |  |
| --- | --- |
|  | (2) |

где – yj – эталонное значение нейрона в выходном слое,

dj – текущее значение нейрона в выходном слое.

Формула взвешенной суммы выглядит следующим образом [2]:

|  |  |
| --- | --- |
|  | (3) |

где – веса между нейронами,

– значения нейронов.

Формула активационного состояния нейрона выглядит следующим образом [2]:

|  |  |
| --- | --- |
|  | (4) |

Формула подсчета ошибок скрытого слоя выглядит следующим образом [2]:

|  |  |
| --- | --- |
|  | (5) |

Формула пересчета весов выглядит следующим образом [2]:

|  |  |
| --- | --- |
|  | (6) |

где – коэффициент скорости обучения.

Алгоритм обучения нейронной сети методом обратного распространения ошибки можно разделить на 2 этапа:

1. Расчёт выходных сигналов для заданного тренировочного примера;
2. Сравнение рассчитанных выходных сигналов с эталоном и обновление весовых коэффициентов связей между узлами на основе найденных отличий.

Тренировочный пример – это последовательность данных, которыми оперирует нейронная сеть.

1. Алгоритмы обучения RPROP и iRPROP+

В отличие от стандартного алгоритма BackPROP, RPROP использует только знаки частных производных для подстройки весовых коэффициентов. Алгоритм использует так называемое «обучение по эпохам», когда коррекция весов происходит после предъявления сети всех примеров из обучающей выборки.

|  |  |
| --- | --- |
|  | (7) |

Для определения величины коррекции используется следующее правило [3]:

Если на текущем шаге частная производная по соответствующему весу wij поменяла свой знак, то это говорит о том, что последнее изменение было большим, и алгоритм проскочил локальный минимум, и, следовательно, величину изменения необходимо уменьшить на η– и вернуть предыдущее значение весового коэффициента: другими словами, необходимо произвести «откат» [3]:

|  |  |
| --- | --- |
|  | (8) |

Если знак частной производной не изменился, то нужно увеличить величину коррекции на η+ для достижения более быстрой сходимости. Зафиксировав множители η− и η+, можно отказаться от глобальных параметров настройки нейронной сети, что также можно рассматривать как преимущество рассматриваемого алгоритма перед стандартным алгоритмом Backprop.

Рекомендованные значения для η− = 0.5, η+ = 1.2, но нет никаких ограничений на использование других значений для этих параметров.

Для того, чтобы не допустить слишком больших или малых значений весов, величину коррекции ограничивают сверху максимальным Δmax и снизу минимальным Δmin значениями величины коррекции, которые по умолчанию, соответственно, устанавливаются равными 50 и 1.0E-6.

Начальные значения для всех Δij устанавливаются равными 0.1. Опять же, это следует рассматривать лишь как рекомендацию, и в практической реализации можно задать другое значение для инициализации.

Для вычисления значения коррекции весов используется следующее правило [3]:

|  |  |
| --- | --- |
|  | (9) |

Если производная положительна, т.е. ошибка возрастает, то весовой коэффициент уменьшается на величину коррекции, в противном случае – увеличивается.

Затем подстраиваются веса [3]:

|  |  |
| --- | --- |
|  | (10) |

Сразу после появления, алгоритм RProp завоевал популярность благодаря простоте и высокой скорости и эффективности обучения. Однако, нужно отметить, что данный алгоритм использует только информацию о локальном градиенте ошибки (точнее, только градиент ошибки по данному весу). В целом ряде работ были попытки улучшить скорость и эффективность обучения нейронной сети за счет учета глобального изменения ошибки. Одна из простейших 24 модификаций базового алгоритма оказалась на редкость эффективной. Суть отличия iRPROP+ от RPROP – если изменился знак градиента, производить откат только в случае, если общая ошибка классификации увеличилась с предыдущего шага [6].

Схема алгоритма обучения RPROP представлена на рисунке 4. Схема алгоритма обучения iRPROP+ представлена на рисунке 5.

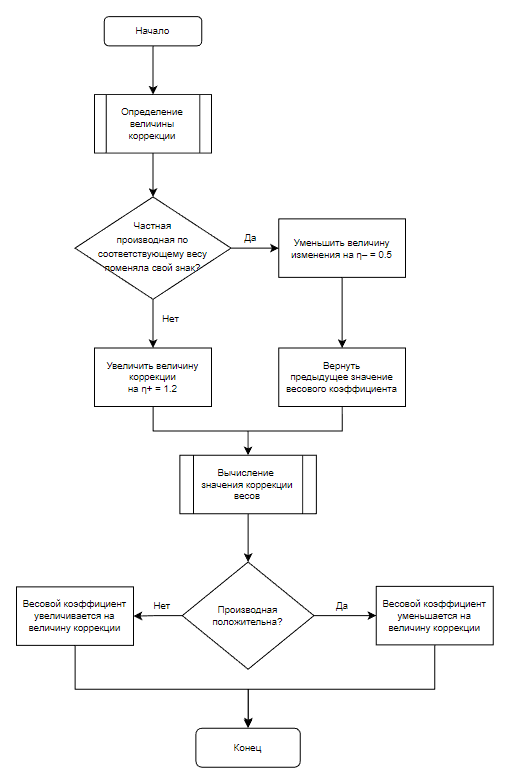


Рисунок 4 – Схема алгоритма обучения RPROP

Схема алгоритма обучения iRPROP+ представлена на рисунке 5.

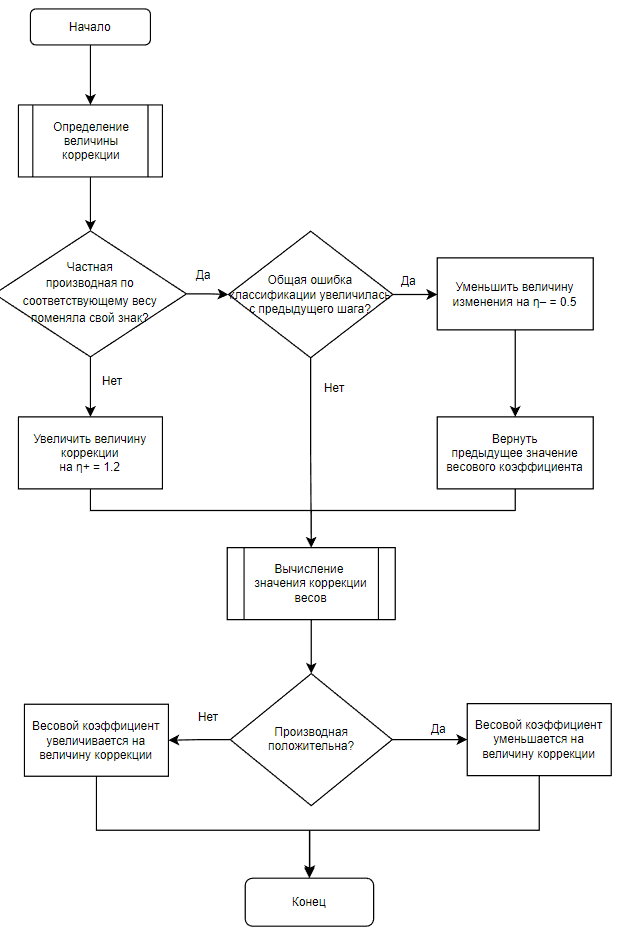


Рисунок 5 – Схема алгоритма обучения iRPROP+

Исследования

1. Исследовать зависимость погрешности обучения для каждого алгоритма от значения коэффициента обучения.
2. Исследовать зависимость погрешности обучения и классификации для каждого алгоритма от объёма обучающей выборки.
3. Исследовать зависимость погрешности обучения и классификации для каждого алгоритма от числа нейронов скрытого слоя.
4. Исследовать зависимость погрешности классификации от числа итераций обучения.

В исследованиях в качестве погрешностей обучения и прогнозирования считается среднеквадратическое отклонение:

*,*

где – объем выборки,

– полученное значение,

- эталонное значение.

Синей линией обозначена погрешность обучения печатных букв. Красной линией обозначения погрешность обучения печатных латинских букв.

1. Исследовать зависимость погрешности обучения для каждого алгоритма от значения коэффициента обучения.

1 Алгоритм обратного распространения ошибки

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 6 изображена зависимость погрешности обучения от значения коэффициента обучения алгоритма обратного распространения ошибки. Для подсчета погрешности были выбраны 100 нейронов скрытого слоя и 10 итераций обучения.

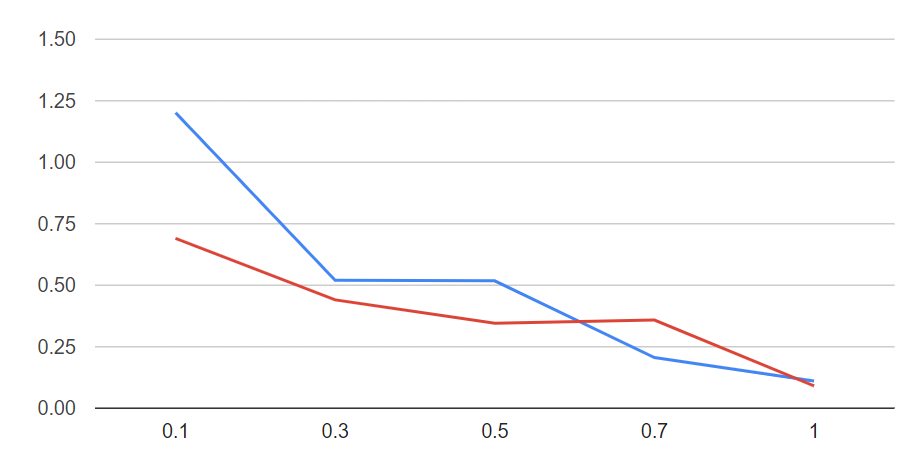


Рисунок 6 - Зависимость погрешности обучения от значения коэффициента обучения алгоритма обратного распространения ошибки.

2 Алгоритм RPROP

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 7 изображена зависимость погрешности обучения от значения коэффициента обучения алгоритма RPROP. Для подсчета погрешности были выбраны 100 нейронов скрытого слоя и 10 итераций обучения.

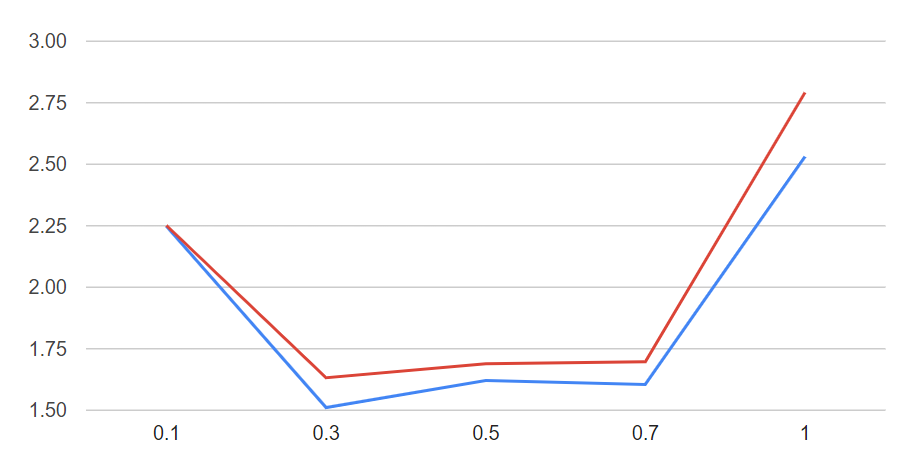


Рисунок 7 - Зависимость погрешности обучения от значения коэффициента обучения алгоритма RPROP

3 Алгоритм iRPROP+

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 8 изображена зависимость погрешности обучения от значения коэффициента обучения алгоритма iRPROP+. Для подсчета погрешности были выбраны 100 нейронов скрытого слоя и 10 итераций обучения.

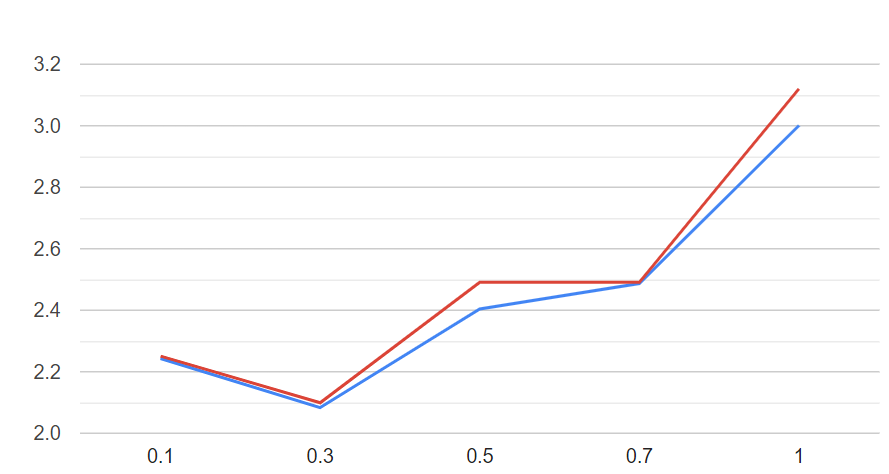


Рисунок 8 - Зависимость погрешности обучения от значения коффициента обучения алгоритма iRPROP+

2 Исследовать зависимость погрешности обучения и классификации для каждого алгоритма от объёма обучающей выборки

1 Алгоритм обратного распространения ошибки

На рисунке 9 изображена зависимость погрешности обучения от объема обучающей выборки алгоритма обратного распространения ошибки. Для подсчета погрешности были выбраны 100 нейронов скрытого слоя, коэффициент обучения 0.1 и 10 итераций обучения.

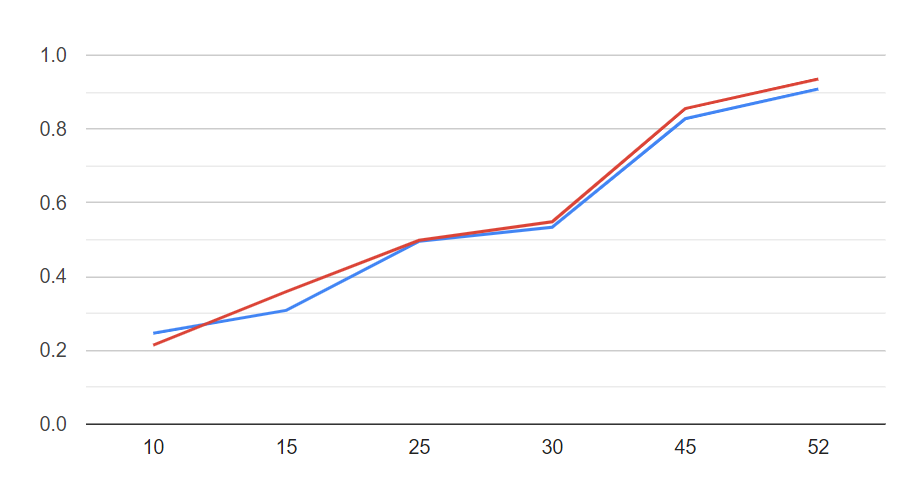


Рисунок 9 - Зависимость погрешности обучения от объема обучающей выборки

2 Алгоритм RPROP

На рисунке 10 изображена зависимость погрешности обучения от объема обучающей выборки алгоритма RPROP. Для подсчета погрешности были выбраны 120 нейронов скрытого слоя, коэффициент обучения 0.3 и 10 итераций обучения.

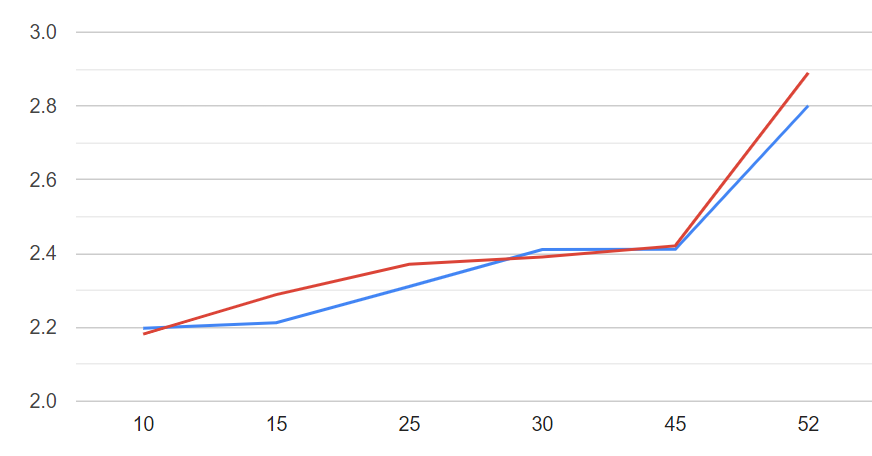


Рисунок 10 - Зависимость погрешности обучения от объема обучающей выборки

3 Алгоритм iRPROP+

На рисунке 11 изображена зависимость погрешности обучения от объема обучающей выборки алгоритма iRPROP+. Для подсчета погрешности были выбраны 120 нейронов скрытого слоя, коэффициент обучения 0.3 и 10 итераций обучения.

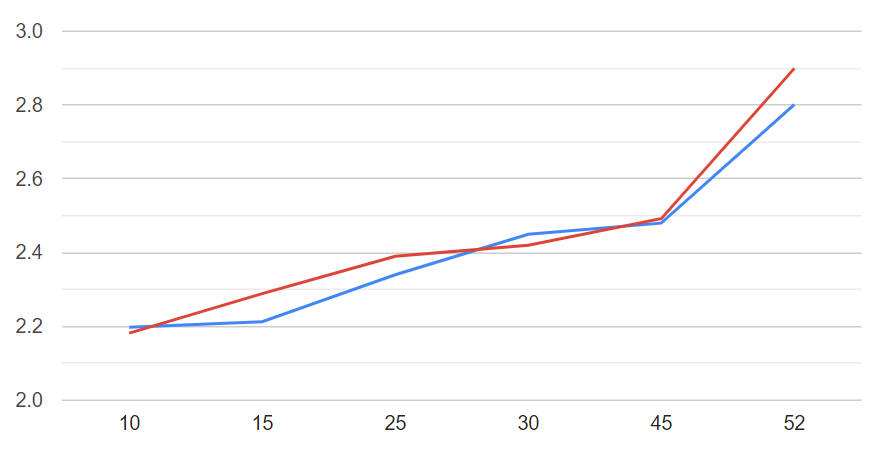


Рисунок 11 - Зависимость погрешности обучения от объема обучающей выборки

3 Исследовать зависимость погрешности обучения и классификации для каждого алгоритма от числа нейронов скрытого слоя

1 Алгоритм обратного распространения ошибки

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 12 изображена зависимость погрешности обучения от числа нейронов скрытого слоя обучения алгоритма обратного распространения ошибки. Для подсчета погрешности были выбраны коэффициент обучения 0.5 и 10 итераций обучения.

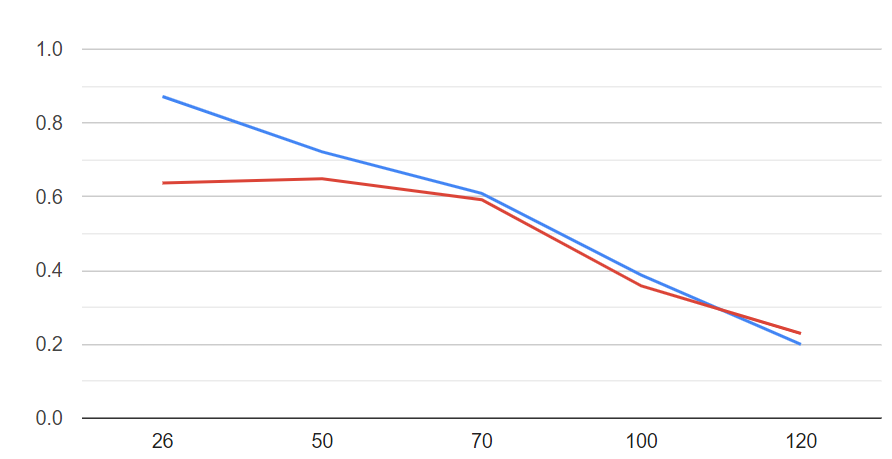


Рисунок 12 - Зависимость погрешности обучения от числа нейронов скрытого слоя

2 Алгоритм обучения RPROP

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 13 изображена зависимость погрешности обучения от числа нейронов скрытого слоя обучения алгоритма RPROP. Для подсчета погрешности были выбраны коэффициент обучения 0.3 и 10 итераций обучения.

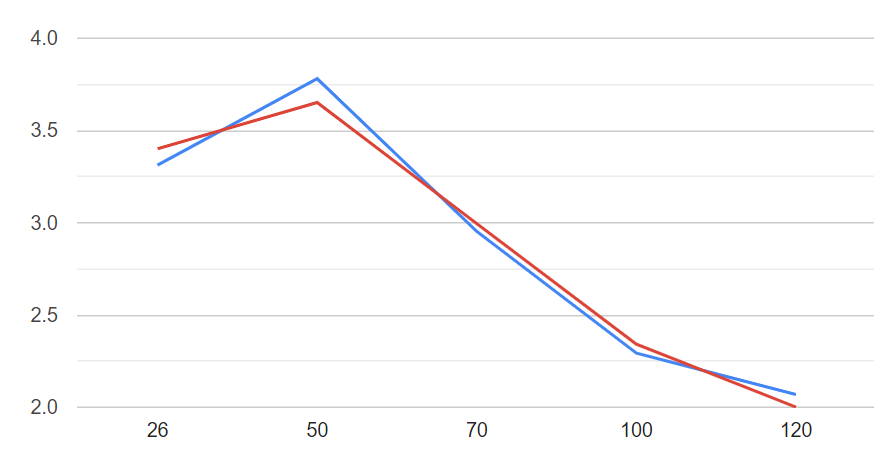


Рисунок 13 - Зависимость погрешности обучения от числа нейронов скрытого слоя

3 Алгоритм обучения iRPROP+

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 14 изображена зависимость погрешности обучения от числа нейронов скрытого слоя обучения алгоритма iRPROP+. Для подсчета погрешности были выбраны коэффициент обучения 0.3 и 10 итераций обучения.

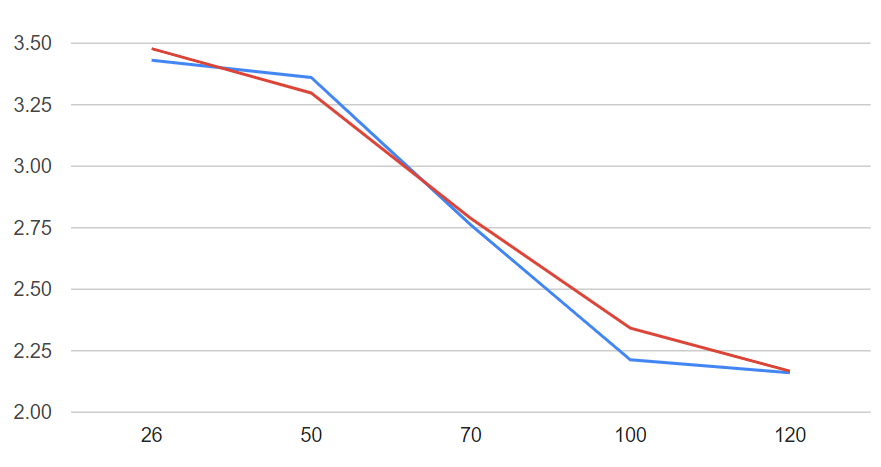


Рисунок 14 - Зависимость погрешности обучения от числа нейронов скрытого слоя

4 Исследовать зависимость погрешности классификации от числа итераций обучения

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 15 изображена зависимость погрешности обучения от числа итераций обучения алгоритма обратного распространения ошибки. Для подсчета погрешности были выбраны коэффициент обучения 0.5 и 100 нейронов скрытого слоя.

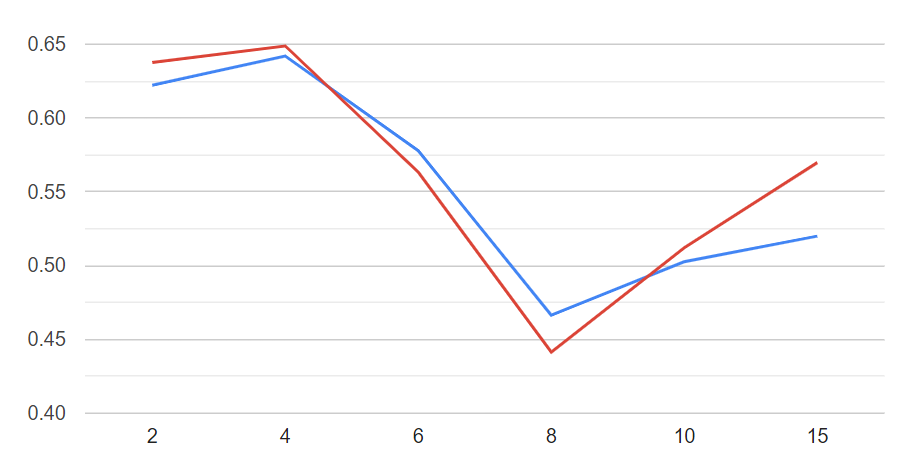


Рисунок 15 - Зависимость погрешности обучения от числа итераций обучения

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 16 изображена зависимость погрешности обучения от числа итераций обучения алгоритма RPROP. Для подсчета погрешности были выбраны коэффициент обучения 0.3 и 120 нейронов скрытого слоя.

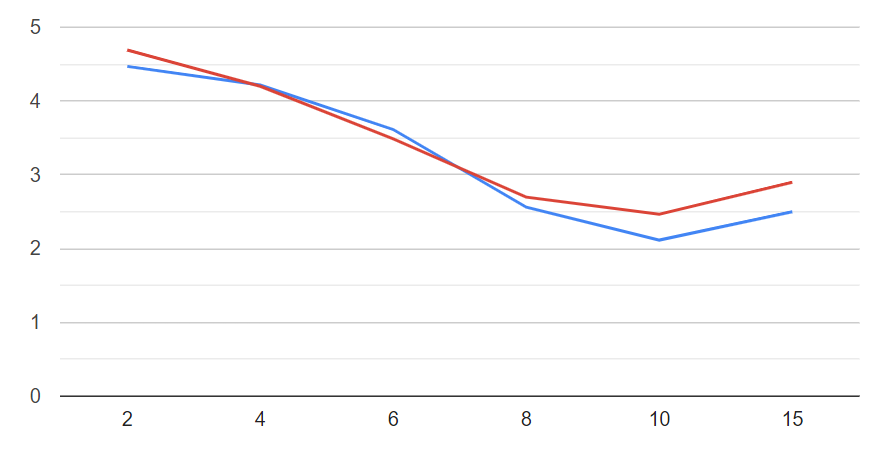


Рисунок 16 - Зависимость погрешности обучения от числа итераций обучения

Для исследования использована обучающая выборка объемом 52 буквы. На рисунке 17 изображена зависимость погрешности обучения от числа итераций обучения алгоритма iRPROP+. Для подсчета погрешности были выбраны коэффициент обучения 0.3 и 120 нейронов скрытого слоя.

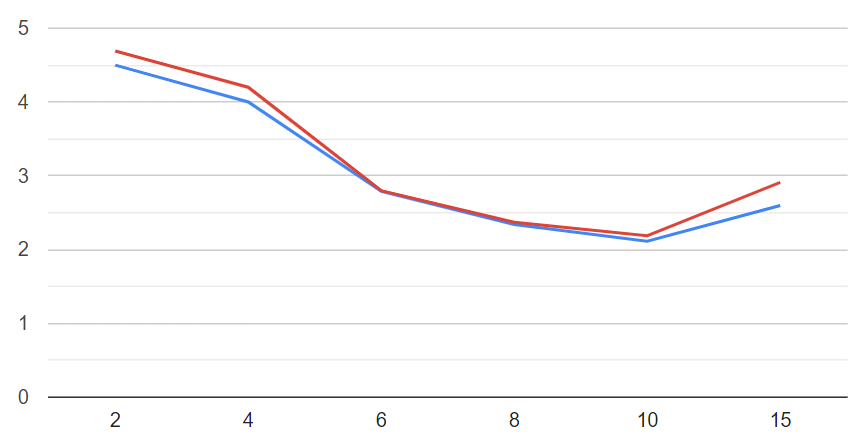


Рисунок 17 - Зависимость погрешности обучения от числа итераций обучения

ВЫВОДЫ

В результате работы была написана программа, имитирующая работу многослойного персептрона и проведено обучение сети по алгоритмам обратного распространения ошибки, RPROP и iRPROP+. Также проведены исследования зависимости погрешностей обучения от разных параметров.

Из полученных результатов можно сделать вывод, что для достижения наилучшего процента распознавания коэффициент обучения для алгоритма обучения обратного распространения ошибки должен быть 1; для алгоритма обучения RProp и iRProp+ - 0.3; а количество нейронов в скрытом слое для алгоритма обучения обратного распространения ошибки должно быть 120; для алгоритмов обучения RProp и iRProp+ - 120. Во всех трех алгоритмах, чем больше объем обучающих данных, тем больше погрешность.

Список использованной литературы

1. Искусственная нейронная сеть [Электронный ресурс]/ – Электрон. текстовые дан., Режим доступа: http://www.machinelearning.ru/wiki/index.php?title=Искусственная\_нейронная\_сеть, свободный;
2. Метод обратного распространения ошибки [Электронный ресурс]. – https://ru.wikipedia.org/wiki./Метод\_обратного\_распространения\_ошибки
3. Алгоритм RPROP [Электронный ресурс] – Электрон. Текстовые дан., Режим доступа: <https://basegroup.ru/community/articles/rprop>, свободный.
4. Гуисов, М.И. Применение нейронных сетей для оценки позиции в игре Го [Текст]/ Гуисов, М.И. <http://is.ifmo.ru/diploma-theses/guisov/guisov.pdf>
5. Рашид, Т. Создаем нейронную сеть [Текст]/ Т. Рашид; пер. с англ. Гузикевича, А.Г., ред. Гузикевича, А.Г. – М: Диалектика, 2017 – 273 с.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Самарский национальный исследовательский университет имени академика С.П. Королёва» (Самарский университет)

Институт информатики и кибернетики

Кафедра информационных систем и технологий

**ОТЧЕТ**

к лабораторной работе №4 по дисциплине

«Интеллектуальныесистемы управления»

Выполнил студент группы 6132

Иванов И.И.

Проверил:

к.т.н., доцент Солдатова О. П.

Самара 2023

ПОСТАНОВКА ЗАДАЧИ

ЗАДАНИЕ №9

Цель работы. Целью данной лабораторной работы является демонстрация способности нейронной сети решать задачи прогнозирования. Сеть необходимо обучить прогнозированию на основе стохастических рядов и рядов реальных данных на 1 шаг.

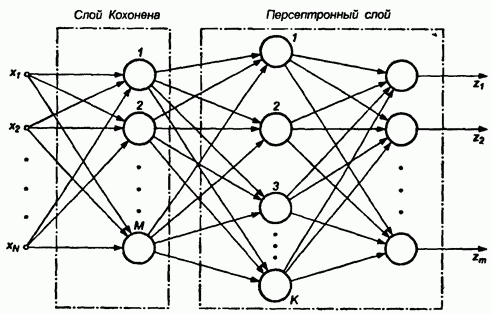
Используемая модель: гибридная сеть Кохонена.

Порядок выполнения работы. Данная лабораторная работа выполняется в несколько этапов:

1. Необходимо выбрать вид функционального ряда и тип реальных данных. Примеры реальных данных приведены ниже.
2. Сгенерировать функциональный временной ряд размерностью от 100 до 300 отсчётов, найти реальные данные той же размерности. Предусмотреть нормализацию входных векторов.
3. Написать программу, имитирующую работу гибридной сети и провести обучение слоя Кохонена по алгоритму WTA (1 занятие).
4. Провести обучение многослойного персептрона по методу наискорейшего спуска с моментами с использованием алгоритма обратного распространения ошибки.
5. Продемонстрировать работу гибридной сети преподавателю (2 занятие).
6. Исследовать погрешности обучения слоя Кохонена и многослойного персептрона от значений коэффициента обучения.
7. Исследовать зависимость погрешности обучения и классификации от числа нейронов слоя Кохонена.
8. Исследовать зависимость погрешности прогнозирования от объёма обучающей выборки.
9. Исследовать зависимость погрешности прогнозирования от числа итераций обучения многослойного персептрона.
10. Исследовать зависимость погрешности прогнозирования от размера скользящего окна.
11. Составить отчёт, который должен содержать постановку задачи, обучающие выборки, результаты исследований по пунктам 6-10 и сдать его преподавателю (3 занятие).

ОПИСАНИЕ СЕТИ

На рисунке 1 показана структура гибридной сети Кохонена

  
Рисунок 1 – Структура гибридной сети Кохонена

Гибридная сеть Кохонена представляет собой каскадное подключение слоя Кохонена к персептронной сети. Самоорганизующийся слой улавливает значимые признаки процесса (локализует их на основе входных данных x), после чего им приписывается входной вектор в персептронном слое. Обучение гибридной сети состоит из двух отдельных этапов, следующих друг за другом.

Алгоритм обучения

В ходе разработки и исследования были использованы два слоя: слой Кохонена, для классификации входных данных и многослойный персептрон. Для обучения слоя Кохонена использован алгоритм WTA (Winner take all). Победитель уточняет свои веса по формуле 1. Остальные веса нейронов уточнению не подлежат.

 (1)

Для уточнения данных использован многослойный персептрон. Выходной вектор нормализуется и поступает на вход многослойного персептрона, который в свою очередь обучения по методу наискорейшего спуска с моментами с использованием алгоритма обратного распространения ошибки.

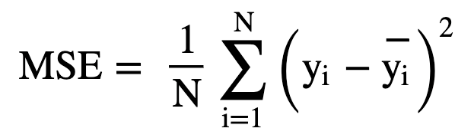
Алгоритм обратного распространения ошибки определяет стратегию подбора весов многослойной сети с применением градиентных методов оптимизации.

В качестве функции активации нейронов была выбрана сигмоидальная нелинейная функция.

|  |  |
| --- | --- |
|  | (2) |

Определение точности прогнозирования

В качестве погрешности обучения и тестирования будем рассчитывать MSE (формула 3).

 (3)

–

n – размер выборки.

Используемые датасеты

1.Функционалный ряд

В качестве набора данных для функционального ряда используется ряд sin(x).

2.Реальный ряд

Набор данных содержит среднемесячное значение температуры в Бангладеше с 1901 по 2015 год.

Результаты работы сети

На рисунке 3 представлен прогноз на тестовой выборке на реальном ряде.

Рисунок 2 – Прогноз сети на тестовой выборке на функциональном ряде

Рисунок 3 – Прогноз сети на тестовой выборке на реальном наборе данных

1. Исследовать погрешности обучения слоя Кохонена и многослойного персептрона от значений коэффициента обучения.

Входные данные:

* Обучающая выборка: 1380 значений
* Тестовая выборка: 12 значений
* Число эпох обучения: 200

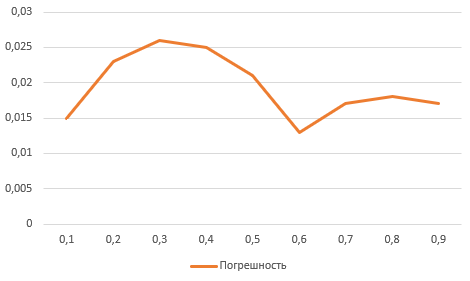


Рисунок 3 – Зависимость погрешности обучения от коэффициента обучения

Таким образом, наименьшая погрешность возникает при коэффициенте обучения равном 0,6.

1. Исследовать зависимость погрешности обучения и классификации от числа нейронов слоя Кохонена.

Входные данные:

* Обучающая выборка: 1380 значений
* Тестовая выборка: 12 значений
* Число эпох обучения: 200
* Коэффициент обучения: 0.6

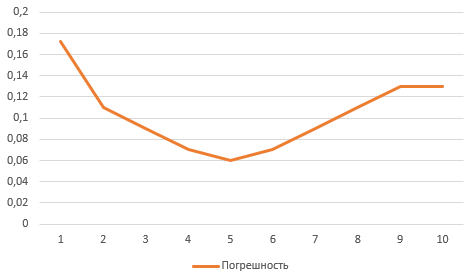


Рисунок 4 – Зависимость погрешности обучения от числа нейронов слоя Кохонена

Таким образом, наименьшая погрешность возникает при числе нейронов равном 5.

1. Исследовать зависимость погрешности классификации от объёма обучающей выборки.

Входные данные:

* Число эпох обучения: 200
* Коэффициент обучения: 0.6
* Количество нейронов: 5

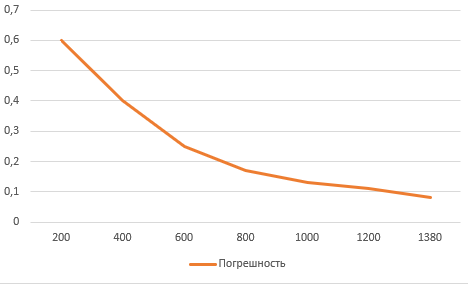


Рисунок 5 – Зависимость погрешности от размера выборки

Таким образом, чем больше размер выборки, тем меньше погрешность обучения. Поэтому выбран весь объем, а именно 1380 значений.

1. Исследовать зависимость погрешности классификации от числа итераций обучения слоя Кохонена и многослойного персептрона.

Входные данные:

* Коэффициент обучения: 0.6
* Количество нейронов: 5

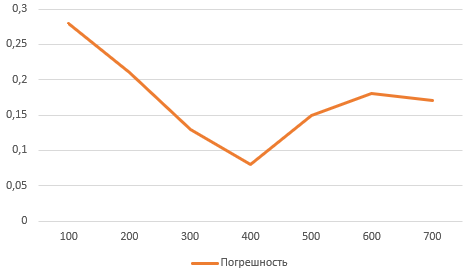


Рисунок 6 – Зависимость погрешности прогнозирования от числа итераций обучения многослойного персептрона

Таким образом, наиболее оптимальным значением, при котором погрешность прогнозирования минимальна, равно 400 эпох обучения.

1. Зависимость погрешности прогнозирования от размера скользящего окна

Входные данные:

* Обучающая выборка: 1380 значений
* Тестовая выборка: 12 значений
* Число эпох обучения: 400
* Коэффициент обучения: 0.6
* Число нейронов: 5



Рисунок 7 – Зависимость погрешности прогнозирования от размера скользящего окна

Таким образом, наименьшая погрешность прогнозирования достигается при размере окна, равном 5.

РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Таблица 1 – Оптимальные параметры сети

|  |  |
| --- | --- |
|  | Значение |
| Обучающая выборка | 1380 |
| Число эпох обучения | 400 |
| Коэффициент обучения | 0.6 |
| Число нейронов | 5 |

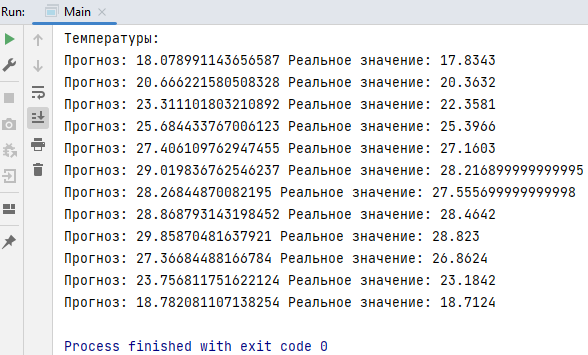


Рисунок 8 – Результат работы сети

ВЫВОДЫ

Таким образом, была написана программа, имитирующая работу гибридной сети Кохонена. В результате исследования установлена зависимость погрешности прогнозирования от числа нейронов, размера выборки, количества итераций обучения.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»

(Самарский университет)

Институт информатики и кибернетики

Кафедра информационных систем и технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

По курсу «Интеллектуальные системы»

на тему:

«Решение задачи классификации с помощью сети AlexNet»

По направлению подготовки 09.04.01 Интеллектуальные системы управления цифровой инфраструктурой предприятия

(уровень магистратуры)

Студент группы 6232-090401D\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_И.И. Иванов

Преподаватель, к.т.н., доцент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О.П. Солдатова

Самара 2023

Цель работы. Продемонстрировать способность нейронной сети решать задачу классификации, реализованной с помощью технологий быстрого создания структуры нейронной сети с помощью библиотек.

Используемая модель: AlexNet.

Порядок выполнения работы. Данная лабораторная работа выполняется в несколько этапов:

1. Создать блокнот ipynb с подключенными библиотеками Keras и TensorFlow
2. Выбрать и подготовить датасет, на котором будет тестироваться работа нейронной сети AlexNet
3. Задать структуру сети при помощи библиотек, согласно структуре сети
4. Выполнить обучение нейронной сети и убедиться в правильности задания структуры с помощью библиотек. Параллельно с этим, собирать данные уменьшения ошибки в зависимости от количества эпох обучения. Параллельно собирать данные о точности распознавания тестовой выборки от количества эпох обучения
5. Выполнить тестирование на тестовой выборке, убедиться в корректности распознавания

6. Составить отчёт, который должен содержать постановку задачи, обучающие выборки, результаты исследований по пунктам 1-5 и сдать его преподавателю (3 занятие).

### Выбранная предметная область:

mnist

Описание предметной области

### Mnist является Объёмной базой данных образцов рукописного написания цифр

Структура выбранной сети

AlexNet — сверточная нейронная сеть, различные модификации алгоритма которой используются для задач компьютерного зрения

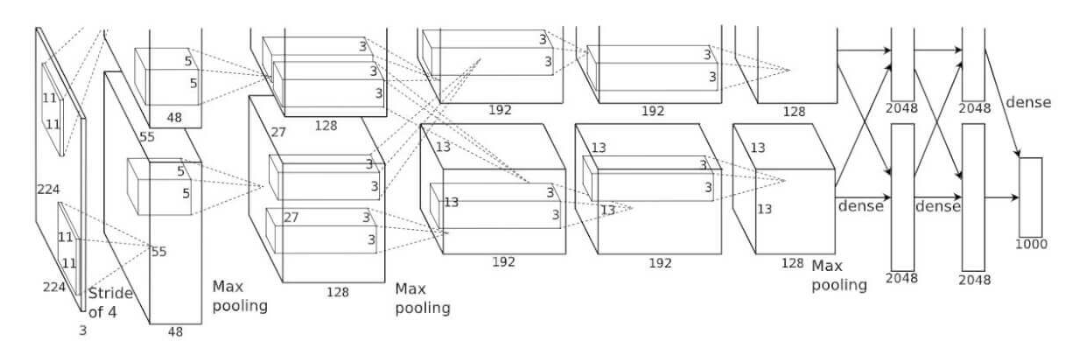


Рисунок 1 – Структура сети Alex Net

Краткое описание сети

Архитектура AlexNet позволила сократить число ошибок для пяти ведущих категорий из выбранного датасета до 16,4 процента — почти вдвое по сравнению с предыдущими передовыми разработками в области распознавания изображений.

В рамках данной архитектуры была представлена такая функция активации, как блок линейной ректификации (ReLU), который является в настоящее время отраслевым стандартом. Далее приведена краткая сводка прочих основных свойств архитектуры AlexNet и процесса ее обучения:

* Интенсивное расширение (аугментация) данных
* Метод исключения
* Оптимизация с помощью момента SGD
* Ручная настройка скорости обучения и ее изменение при обучении

Итоговая модель представляет собой совокупность из семи сверточных нейросетей:

* Сверточный слой с выбором максимума с дальнейшей нормализацией
* 3 сверточных слоя с выбором максимума
* 3 полносвязных слоя

Обучение сети

Обучение сети производится штатными средствами Keras и TensorFlow библиотек.

Исследования

В качестве меры правильности работы сети возьмем итоговое значение ошибочно распознанных объектов к общему числу выборки при тестировании.

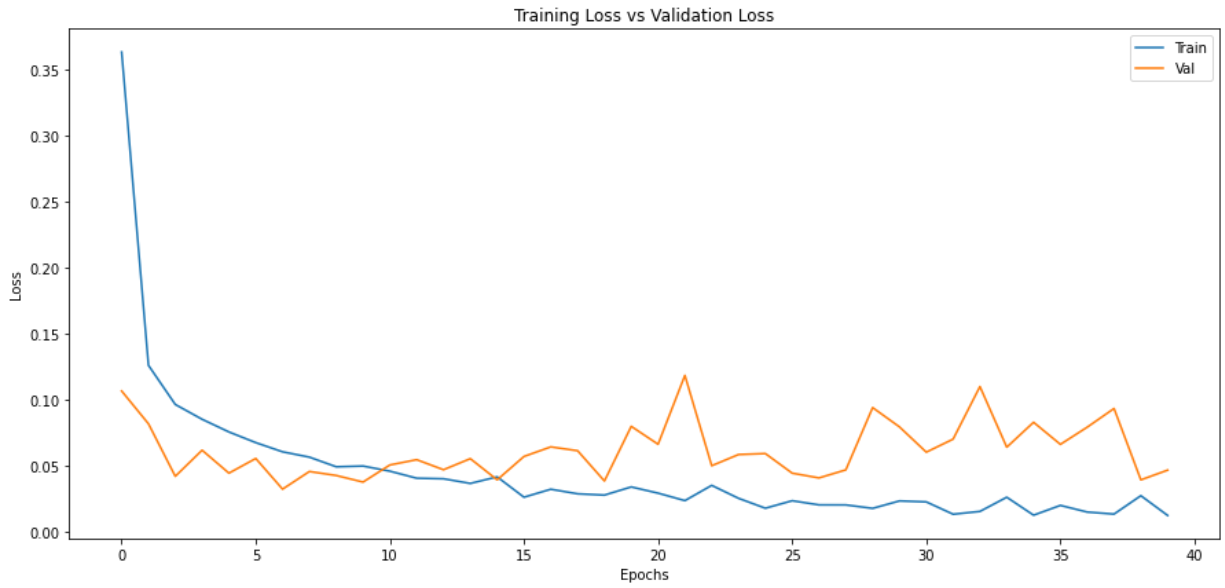


Рисунок 2 – зависимость ошибки от числа эпох обучения, оранжевым показана ошибка распознавания при тестировании, синим – ошибка при обучении

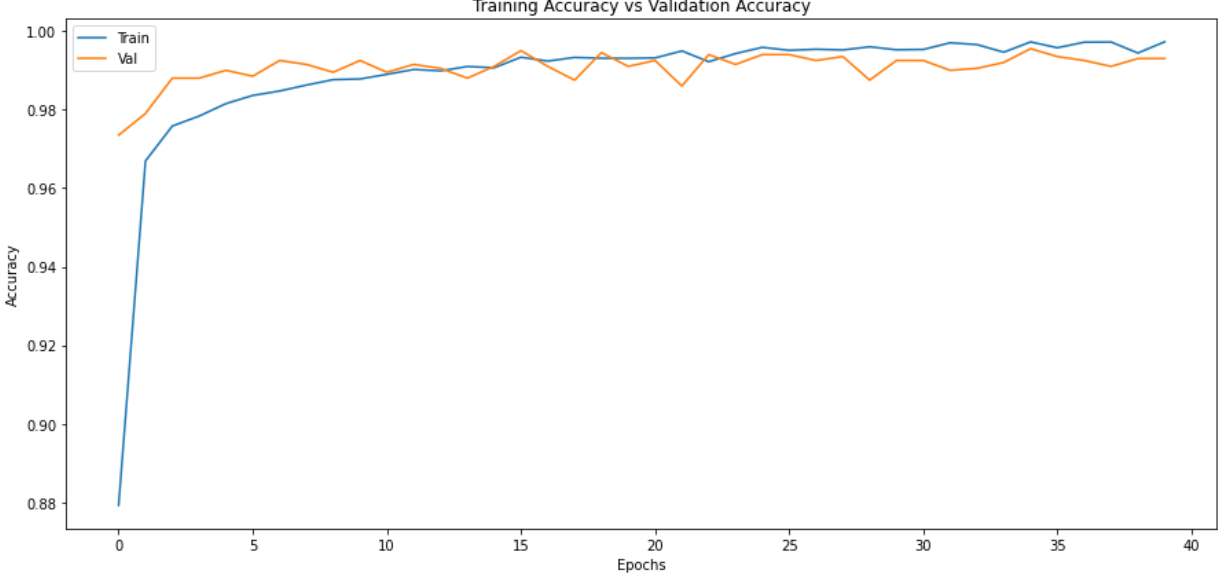


Рисунок 3 – зависимость точности распознавания от числа эпох

Исходя из приведенных графиков на рисунках 2 и 3 можно сделать вывод, что достаточное число эпох обучения составляет 40. Большее число эпох обучения возможно будет иметь более высокую точность распознавания, однако неразрешимо с точки зрения времени обучения (потребовалось около 20 часов на осуществление обучения на кластере colab от Google) в связи с сложностью семикомпонентной структуры данной сети.

Проверка точности распознавания показана на рисунке 4.

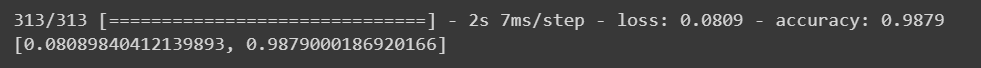


Рисунок 4 – проверка правильности классификации при полном прохождении всех эпох обучения

Выводы

Была построена семикомпонентная нейросеть AlexNet.

Изучены основные компоненты используемых библиотек Keras и TensorFlow.

В ходе процесса обучения была собрана статистика средствами используемых библиотек Keras и TensorFlow.

Точность распознавания является допустимой при полном прохождений всех эпох обучения, составляющих 40 (удалось добиться полной точности распознавания предоставляемых данных при тестировании).

****

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»

Институт информатики и кибернетики

Кафедра информационных систем и технологий

Отчет по лабораторной работе №3

по курсу

«Интеллектуальные системы управления»

Выполнил: студент группы

6231

Иванов И.И.

Проверил:

к.т.н., доцент Солдатова О.П.

Самара 2023

Задание №6

Цель работы

Целью данной лабораторной работы является демонстрация способности нейронной сети решать задачи классификации. Реализовать модель на языке Python в среде Jupyter Notebook с использованием библиотек Numpy, Matplotlib, Keras и Tensorflow.

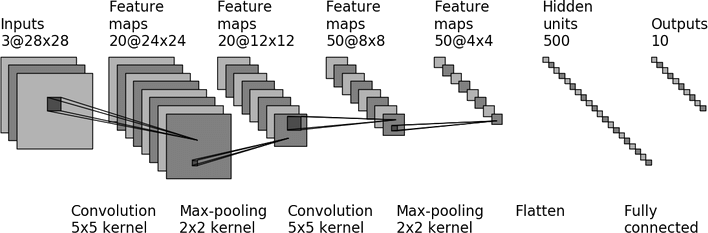
Используемая модель: простая свёрточная сеть (до 10 слоёв).

Порядок выполнения работы. Данная лабораторная работа выполняется в несколько этапов:

1. Выбрана предметная область изображения собак и кошек.
2. Написать программу, имитирующую работу свёрточной сети (в составе должны быть слои свёртки, пулинга, и полносвязные слои) и провести обучение сети с использованием алгоритмов Adam , Adagrad , Adadelta, RMSProp , Nadam .
3. Провести тестирование сети, обученной при помощи всех алгоритмов.
4. Продемонстрировать работу свёрточной сети преподавателю.
5. Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качяества accuracy от объёма обучающей выборки для всех алгоритмов.
6. Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качества accuracy от числа эпох обучения для всех алгоритмов.
7. Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качества accuracy от размера пакета (мини батча) обучения для всех алгоритмов.
8. Построить графики функции погрешности при обучении и классификации, а также метрики качества accuracy для всех исследований.
9. Составить отчёт, который должен содержать постановку задачи, обучающие выборки, результаты исследований по пунктам 5-8 и сдать его преподавателю.

ОПИСАНИЕ

Свёрточная нейронная сеть — специальная архитектура искусственных нейронных сетей, предложенная Яном Лекуном в 1988 год и нацеленная на эффективное распознавание образов, входит в состав технологий глубокого обучения. Архитектура свёрточной нейронной сети представлена на рисунке 1.

  
Рисунок 1 – Архитектура свёрточной нейронной сети

В качестве предметной области был выбран набор данных из 8 000 изображений с 2 категориями: собаки, кошки.

Тестовый набор содержит 2000 изображений. Пример изображения из обучающей выборки представлен на рисунке 2, пример из тестового набора представлен на рисунке 3.



Рисунок 2 – Пример изображения из обучающей выборки

Обучение выполняется по следующему алгоритму:

1. Цветные изображения преобразовываются к размерам 64х64;
2. Первый слой и второй слой сети выполняет свертку входного изображения, в результате чего получается слой размером 60х60х32;
3. Полученный слой отбирается с помощью слоя max-pool с размером фильтра 2 и шагом 2, в результате чего получается слой размером 30х30х32;
4. Далее выполняем свёртку с размером фильтра 3 (четвертый и пятый слои);
5. Полученный слой отбирается с помощью слоя max-pool с размером фильтра 2 и шагом 2;
6. Далее усредняем все значения по последней оси на слое 7;
7. Объединяем с размером фильтра 2 и шагом 2;



Рисунок 3 – Пример изображения из тестовой выборки

Построение модели

В составе модели есть слои свёртки (Conv2D(32, (3,3),…), Conv2D(64, (3,3),…) в качестве функции активации использовалась функция Relu), слои пулинга после каждого сверточного слоя (MaxPooling2D((2, 2)). MaxPooling2D — это способ уменьшить количество параметров модели. GlobalAveragePooling2D()) MaxPooling2D —способ усреднения значений по последней оси. Завершают архитектуру модели полносвязный слой: Dense(units=2, activation='softmax'). Конечный слой имеет выходной размер 2, соответствующий 2 классам изображений. Вся архитектура модели представлена ниже:

*# Initalize CNN*classifier = Sequential()  
  
*# Add 2 convolution layers*classifier.add(Conv2D(filters=32, kernel\_size=(3,3), input\_shape=(64, 64, 3), activation='relu'))  
classifier.add(Conv2D(filters=32, kernel\_size=(3,3), activation='relu'))  
  
*# Add pooling layer*classifier.add(MaxPooling2D(pool\_size=(2,2)))  
  
*# Add 2 more convolution layers*classifier.add(Conv2D(filters=32, kernel\_size=(3,3), activation='relu'))  
classifier.add(Conv2D(filters=32, kernel\_size=(3,3), activation='relu'))  
  
*# Add max pooling layer*classifier.add(MaxPooling2D(pool\_size=(2,2)))  
  
*# Add 2 more convolution layers*classifier.add(Conv2D(filters=32, kernel\_size=(3,3), activation='relu'))  
classifier.add(Conv2D(filters=32, kernel\_size=(3,3), activation='relu'))  
  
*# Add max pooling layer*classifier.add(MaxPooling2D(pool\_size=(2,2)))  
  
*# Add global average pooling layer*classifier.add(GlobalAveragePooling2D())  
  
classifier.add(Dropout(0.5))  
  
*# Add full connection*classifier.add(Dense(units=2, activation='softmax'))

После построении модели нужно ее скомпилировать и затем можно обучать ее. Когда мы компилируем модель, мы объявляем функцию потерь и оптимизатор. В качестве оптимизатора для первоначального исследования модели примем Adam, а так, как в этой работе стоит задача классификации данных, предсказывая вероятность того, принадлежат ли данные к одному или другому классу, в качестве функции потерь возьмем функцию кросс-энтропийных потерь:

*# Compiling the ANN*classifier.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['acc'])

Для обучения размер пакета (мини батча) был выбран равным 32. Сам метод обучения fit представлен ниже:

classifier.fit(  
train\_set,

steps\_per\_epoch=len(train\_set),epochs=50,  
validation\_data=test\_set,  
callbacks = callback\_list,  
validation\_steps=10)

Результат обучения представлен на рисунках ниже (зависимости погрешности при обучении на рис.1 и зависимость метрики качества accuracy при обучении от числа эпох на рис.2). Из рисунка 2 видно, что после 30 эпохи происходит переобучение модели.

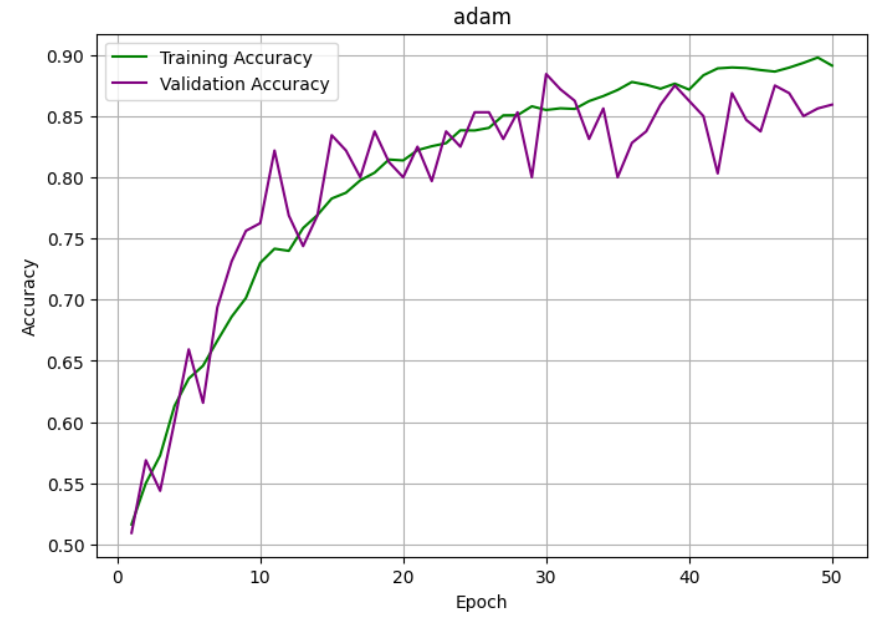


Рисунок 4 – Зависимость погрешности обучения от числа эпох для алгоритма Adam

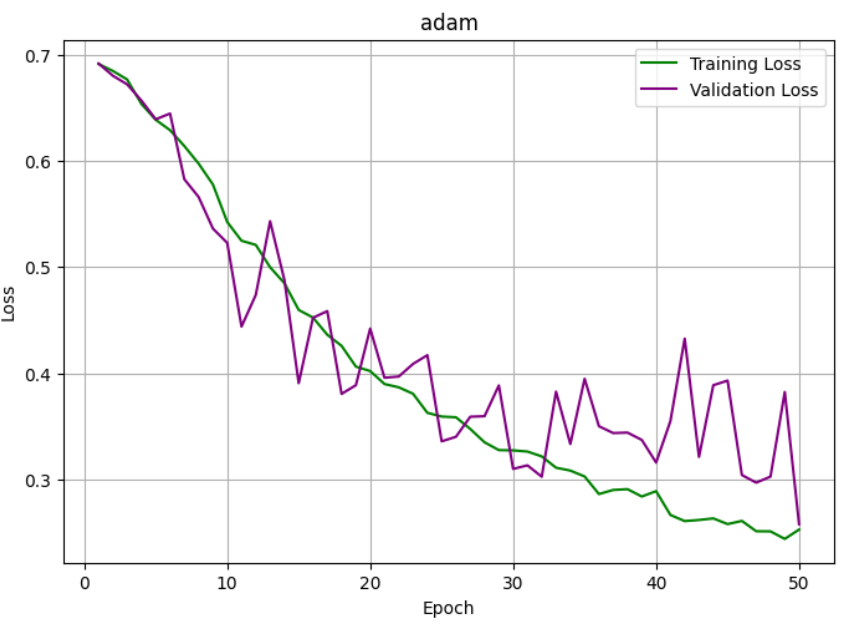


Рисунок 5 – Зависимость метрики качества accuracy при обучении от числа эпох для алгоритма Adam

Таблица 1. Результаты изменения погрешности классификации и метрики accuracy на тестовом наборе

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 5 эпох | 10 эпох | 30 эпох | 50 эпох |
| Loss | 0.64 | 0.52 | 0.31 | 0.26 |
| Accuracy | 0.66 | 0.76 | 0.88 | 0.86 |

Исследования

1. Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качества accuracy от объёма обучающей выборки для всех алгоритмов.

Для этого исследования примем:

- размер пакета (мини батча) = 32;

- количество эпох = 1;

- функция погрешности - categorical\_crossentropy.

На рисунке 6 показана зависимость функции погрешности при обучении от объёма обучающей выборки для всех алгоритмов.

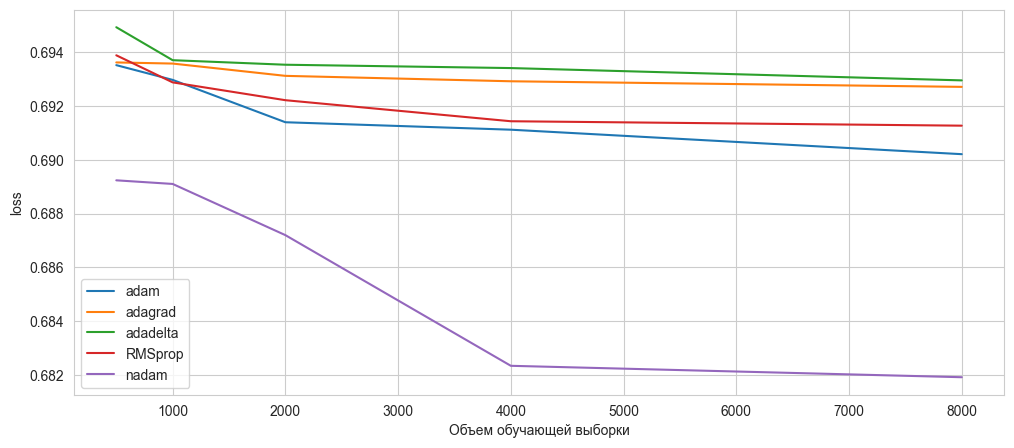


Рисунок 6 – Зависимость погрешности при обучении от объёма обучающей выборки для всех алгоритмов

Как видно из рис.6 наименьшей погрешностью при обучении обладают алгоритмы Adam, Nadam и RMSprop, но наилучшие показатели принадлежат алгоритму Nadam.

На рисунке 7 показана зависимость метрики качества accuracy при обучении от объёма обучающей выборки для всех алгоритмов.

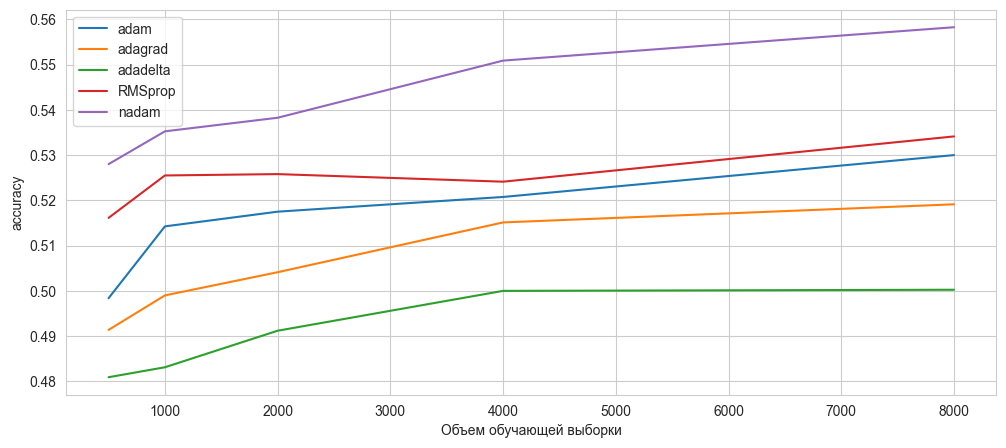


Рисунок 7 – Зависимость accuracy при обучении от объёма обучающей выборки для всех алгоритмов

Как видно из рис.7 наилучшей метрикой accuracy при обучении обладают алгоритмы Adam, Nadam и RMSprop, наилучшие показатели принадлежат алгоритму Nadam.

На тестовом наборе наблюдаются аналогичные результаты при изменении объёма обучающей выборки, как и на обучающем наборе (рисунок 8, рисунок 9).

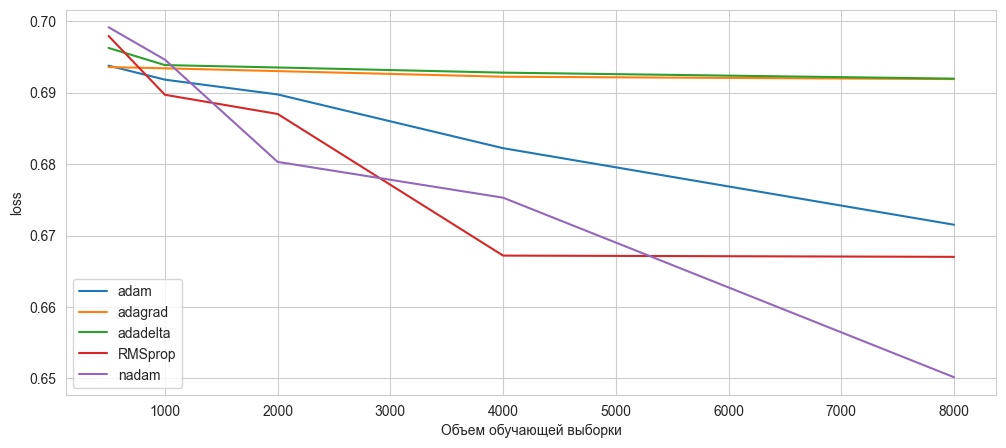


Рисунок 8 – Зависимость погрешности при тестировании от объёма обучающей выборки для всех алгоритмов

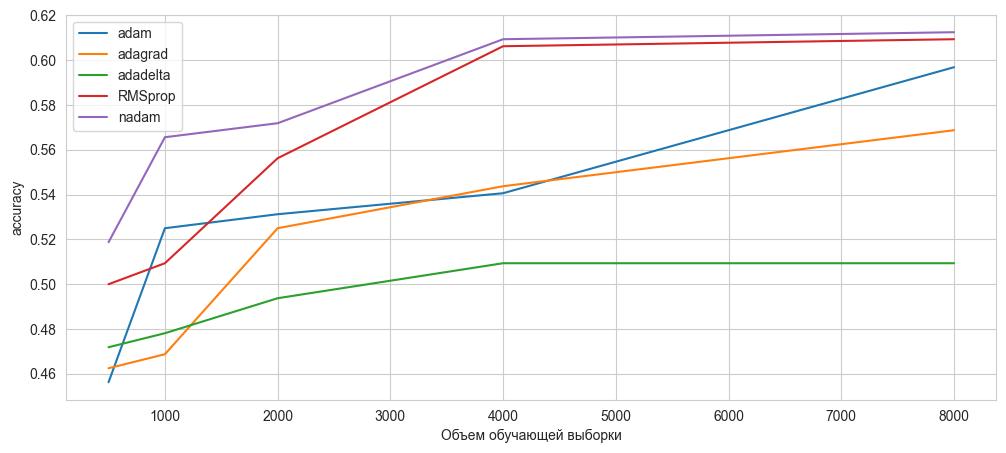


Рисунок 9 – Зависимость accuracy при тестировании от объёма обучающей выборки для всех алгоритмов

Можно сделать вывод, что функция погрешности при обучении и классификации, а также метрика качества accuracy в большинстве случаев обладают наилучшими результатами при наибольшем объёме обучающей выборки.

1. Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качества accuracy от числа эпох обучения для всех алгоритмов.

Для этого исследования примем:

- размер пакета (мини батча) = 32;

- 8000 изображений в обучающем и 2000 в тестовом наборе;

- функция погрешности - categorical\_crossentropy;

На рисунках 7 и 8 показана зависимость функции погрешности при обучении от числа эпох обучения для всех алгоритмов. После 30 эпох все алгоритмы показывают переобучение модели. При увеличении количества эпох уменьшается погрешность для всех алгоритмов.

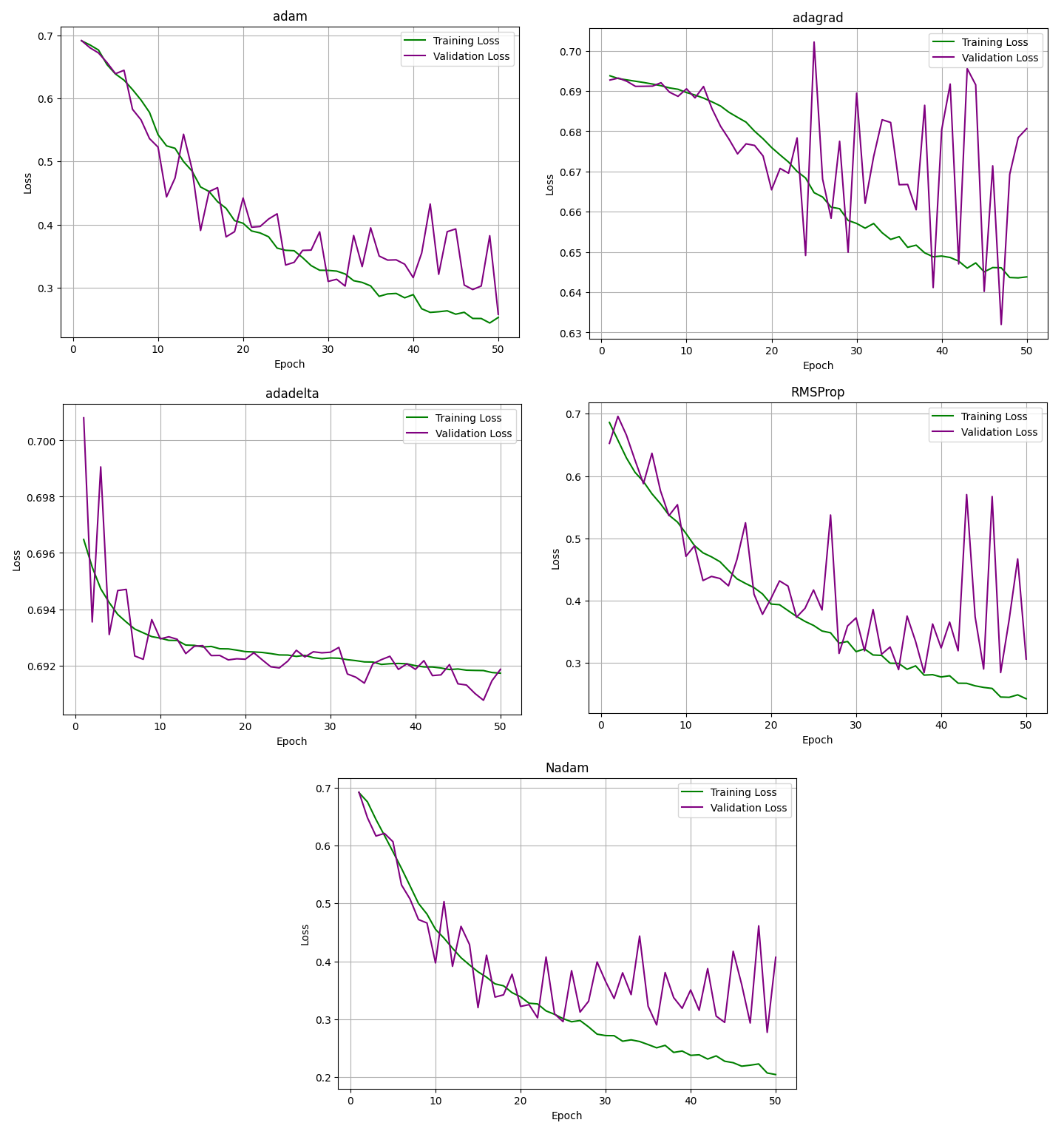


Рисунок 10 – Зависимость погрешности при обучении от числа эпох для всех алгоритмов

При увеличении числа эпох accuracy увеличивается для всех алгоритмов.

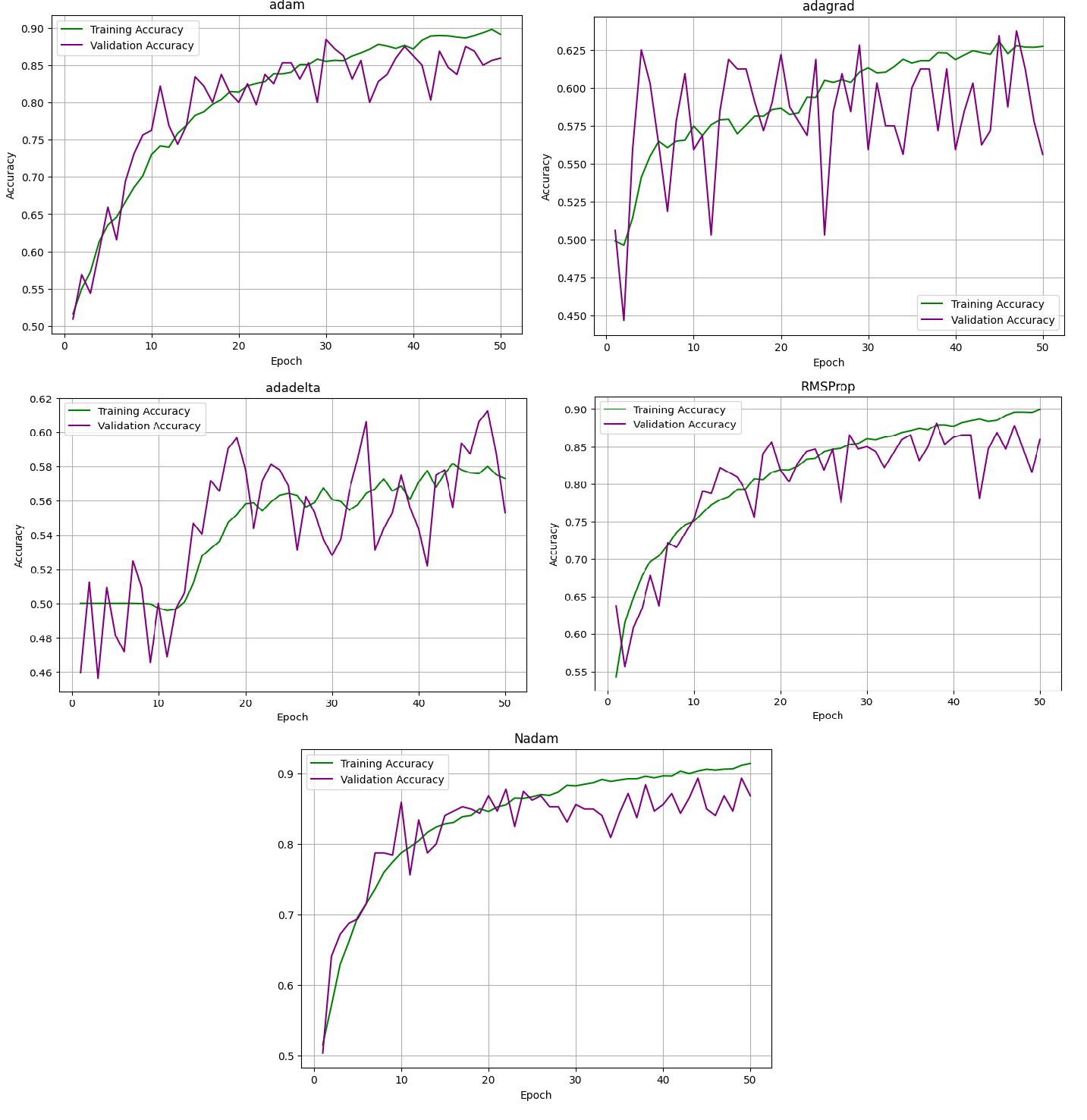


Рисунок 11 – Зависимость метрики качества accuracy от числа эпох для всех алгоритмов

Таблица 3. Результаты изменения погрешности классификации и метрики accuracy на тестовом наборе от алгоритма обучения за 30 эпох обучения

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Adam | Adagrad | Adadelta | RMSprop | Nadam |
| Loss | 0.3097 | 0.6895 | 0.6925 | 0.3722 | 0.3653 |
| Accuracy | 0.8844 | 0.5594 | 0.5281 | 0.8500 | 0.8562 |



Рисунок 12 – Зависимость погрешности при обучении от числа эпох для всех алгоритмов, где dropout равен 0,5

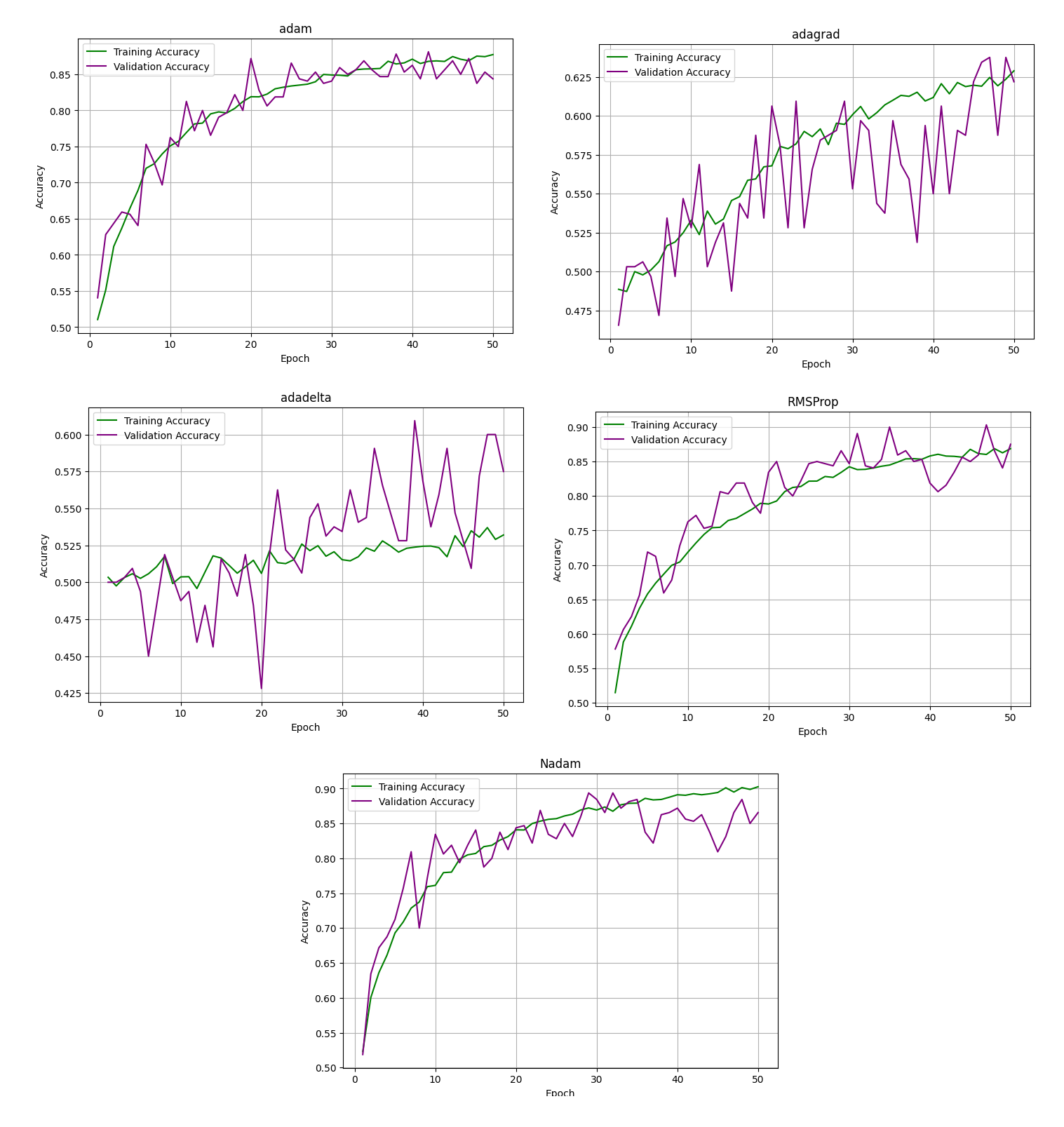


Рисунок 13 – Зависимость метрики качества accuracy от числа эпох для всех алгоритмов, где dropout равен 0,5

Самые лучшие результаты этого исследования показали алгоритмы Adam, Nadam и RMSprop.Оптимальное значение dropout для всех алгоритмов является 0,5.

1. Исследовать зависимость функции погрешности при обучении и классификации, а также метрику качества accuracy от размера пакета (мини батча) обучения для всех алгоритмов.

Для этого исследования примем:

- размер пакета (мини батча) = 32;

- количество эпох = 1;

- функция погрешности - categorical\_crossentropy.

На рисунках 14 и 15 показаны зависимости функции погрешности при обучении и метрики качества accuracy при обучении от размера пакета (мини батча) обучения для всех алгоритмов соответственно. При увеличении размера пакета (мини батча) функция погрешности при обучении увеличивается для всех алгоритмов, а accuracy в общем случае уменьшается.



Рисунок 14 – Зависимость погрешности обучения от размера пакета (мини батча) обучения для всех алгоритмов

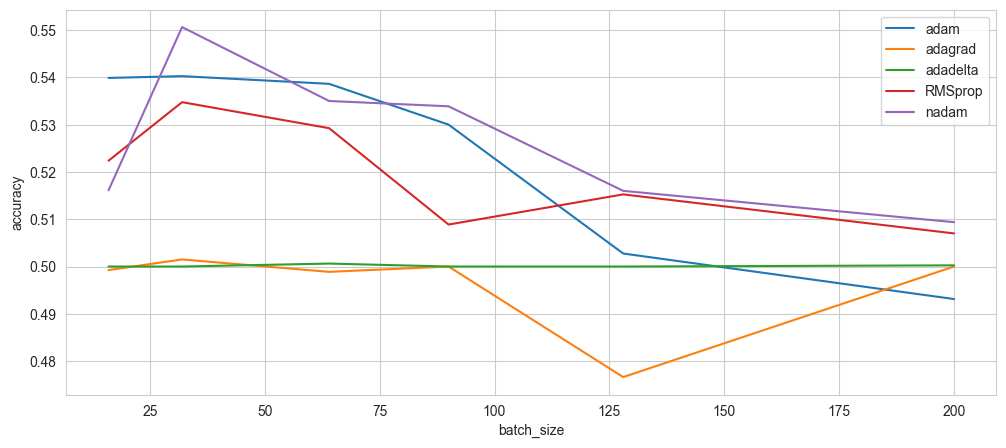


Рисунок 15 – Зависимость метрики качества accuracy от размера пакета (мини батча) обучения для всех алгоритмов

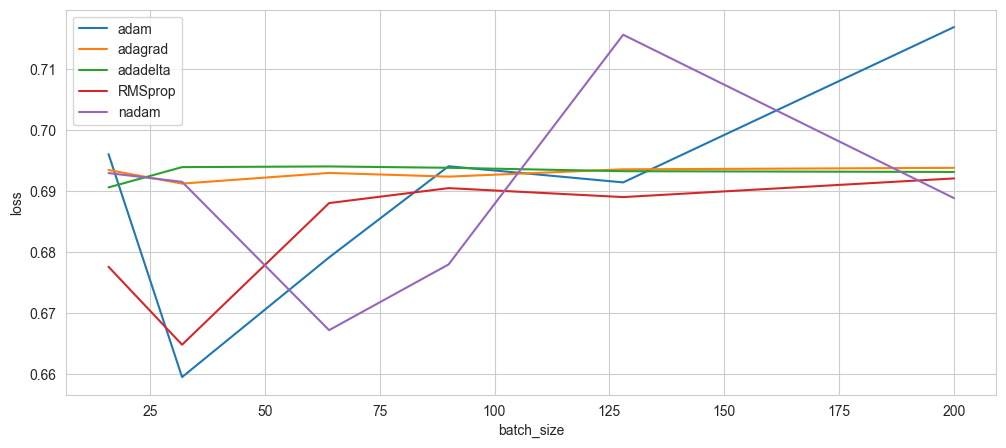


Рисунок 16 – Зависимость погрешности тестирования от размера пакета (мини батча) обучения для всех алгоритмов

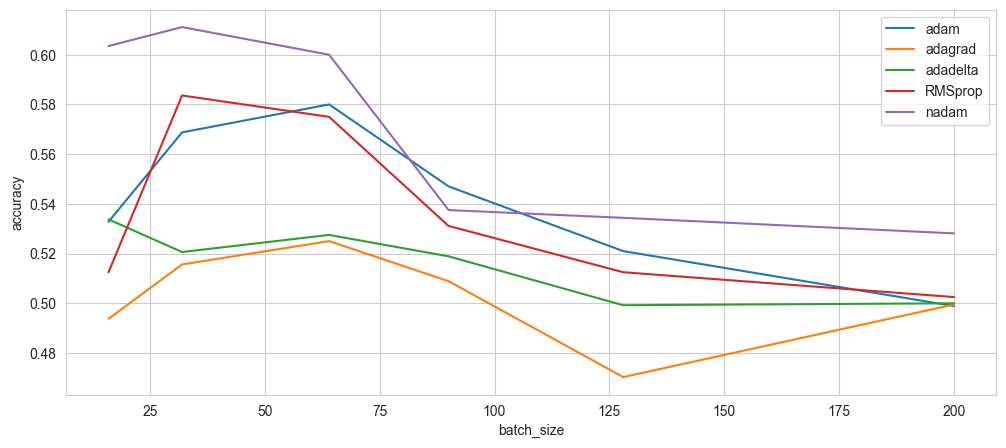


Рисунок 17– Зависимость метрики качества accuracy от размера пакета (мини батча) обучения для всех алгоритмов

Так же, как и в предыдущем исследовании наилучшие показатели погрешности и accuracy принадлежат алгоритмам Nadam и RMSprop при размере пакета равным 32.

Выводы

В результате работы была построена простая свёрточная сеть на языке программирования Python и проведено обучение сети с использованием библиотек Keras и Tensorflow. Были проведены различные исследования данного алгоритма обучения с применением библиотек Numpy и Matplotlib для построения графиков.

Исходя из проведенных исследований, наилучшие результаты по прогнозу дают для алгоритма

Adam:

- количество эпох = 20;

- 8000 изображений в обучающем, 2000 в тестовом наборе;

- размер пакета (мини батча) = 32.

Adagrad:

- количество эпох = 20;

- 8000 изображений в обучающем, 2000 в тестовом наборе;

- размер пакета (мини батча) = 32.

Adadelta:

- количество эпох = 10;

- 8000 изображений в обучающем, 2000 в тестовом наборе;

- размер пакета (мини батча) = 32.

RMSprop:

- количество эпох = 15;

- 8000 изображений в обучающем, 2000 в тестовом наборе;

- размер пакета (мини батча) = 32.

Nadam:

- количество эпох = 20;

- 8000 изображений в обучающем, 2000 в тестовом наборе;

- размер пакета (мини батча) = 64.

Также в результате исследований были отмечены алгоритмы Adam, Nadam и RMSprop, как наилучшие в обучении.