МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по производственной практике НИР

Тема: Исследование параметров нейросетей с управляемыми элементами

Студент гр. 8306 _____ Норбутаев Т.Т.

Руководитель _____ Субботин А.Н.

Санкт-Петербург

2023

ЗАДАНИЕ

НА НАУЧНО-ИССЛЕДОВАТЕЛЬСКУЮ РАБОТУ

Студент Норбутаев Т.Т.

Группа 8306

Тема НИР: Исследование параметров нейросетей с управляемыми элементами

Задание на НИР: Изучение параметров нейронных сетей с управляемыми элементами наборах данных образцов рукописного написания цифр

Сроки прохождения практики: 01.09.2023 – 20.12.2023

Дата сдачи отчета: 18.12.2023

Дата защиты отчета: 26.12.2023

Студент _____ Норбутаев Т.Т.

Преподаватель _____ Субботин А.Н.

АННОТАЦИЯ

Данное исследование посвящено изучению параметров нейронных сетей с управляемыми элементами на примере набора данных MNIST, содержащего рукописные образцы цифр. Целью исследования является выявление эффективных методов управления структурой и обучением нейронных сетей для достижения оптимальной производительности на данном наборе данных. В работе проводится сравнительный анализ различных подходов к управлению элементами нейронных сетей, а также оценивается их воздействие на точность классификации рукописных цифр. Исследование предполагает анализ результатов с целью выявления наиболее перспективных стратегий управления нейронными сетями в контексте данной задачи.

SUMMARY

This study is devoted to studying the parameters of neural networks with controlled elements using the example of the MNIST data set containing handwritten samples of numbers. The purpose of the study is to identify effective methods for controlling the structure and training of neural networks to achieve optimal performance on this data set. The work provides a comparative analysis of various approaches to control elements of neural networks, and also evaluates their impact on the accuracy of classification of handwritten digits. The study involves analyzing the results in order to identify the most promising strategies for controlling neural networks in the context of this task.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире нейронные сети играют ключевую роль в различных областях, начиная от компьютерного зрения и обработки естественного языка и заканчивая управлением производственными процессами и медицинской диагностикой. Однако, несмотря на их мощь, нейронные сети имеют ряд ограничений, таких как сложность интерпретации принимаемых ими решений, сложность обучения в условиях ограниченных данных, склонность к переобучению, и другие.

В данной работе было проведено исследование зависимости качества работы нейронных сетей прямого распространения от скорости обучения.

Скорость обучения - это гиперпараметр [1], который определяет размер шага, с которым обновляются параметры модели во время процесса обучения. Скорость обучения может значительно влиять на качество работы модели, и выбор подходящей скорости обучения является важным условием для достижения оптимальных результатов. Однако выбор оптимальной скорости обучения может быть сложной задачей, так как это зависит от различных факторов, включая выбор оптимизационного метода, размер датасета, и других гиперпараметров.

Библиотеки Keras и TensorFlow использовались для исследования и для сравнения были выбраны: Adagrad, Nadam и Adam.

1. ПОСТАНОВКА ЗАДАЧИ

Актуальность: Актуальность этой темы заключается в том, что оптимизация процесса обучения и выбор подходящих гиперпараметров, включая скорость обучения, являются важными предпосылками для достижения оптимальных результатов в области глубокого обучения. Изучение зависимости качества работы нейронных сетей прямого распространения от скорости обучения [2] поможет определить оптимальные параметры для достижения максимальной точности работы и улучшения результатов в различных приложениях машинного обучения.

Проблема: Проблема заключается в том, что при выборе неправильной скорости обучения [2] нейронная сеть может сходиться к неправильным значениям весов, что может привести к плохому качеству прогнозирования. Также, выбор оптимальной архитектуры сети, выбор оптимальных гиперпараметров и изучение недостаточных объемов данных. Эти проблемы могут привести к плохим результатам обучения или низкой точности прогнозирования.

Цель: Целью этой работы является создание программной модели искусственной нейронной сети прямого распространения сигналов с использованием библиотек TensorFlow и Keras. А также сравнить влияние разных оптимизаторов и их параметров на процесс обучения. Поиск сетевой архитектуры с наивысшей точностью классификации.

Задачи: Создать программную модель искусственной нейронной сети прямого распространения сигналов с использованием библиотек. Выбрать несколько обучающих правил для сравнения. Провести обучение нейронной сети с различными параметрами. Сравнить полученные результаты.

2. РЕЗУЛЬТАТЫ РАБОТЫ В ОСЕННЕМ СЕМЕСТРЕ

Тема диплома была изменена на 3 семестре магистратуры, так что результаты работы за предыдущий семестр не будут приведены.

Для исследования использовалась база данных MNIST.

База данных MNIST (Modified National Institute of Standards and Technology database) представляет собой большую базу данных рукописных цифр, которая обычно используется для обучения различных систем обработки изображений. База данных также широко используется для обучения и тестирования в области машинного обучения. Он был создан путем "повторного смешивания" выборок из исходных наборов данных NIST. База данных MNIST содержит 60 000 обучающих изображений и 10 000 тестовых изображений [3].

Keras – это открытая библиотека, написанная на языке Python и обеспечивающая взаимодействие с искусственными нейронными сетями.

TensorFlow – это открытая библиотека программного обеспечения для машинного обучения, разработанная Google для решения задач автоматического поиска и классификации изображений, построения нейронной сети и обучения с целью достижения качества человеческого восприятия [4].

Оптимизаторы:

Adagrad - адаптивный градиент - это алгоритм, который адаптирует скорость обучения каждого параметра на основе исторической информации о градиенте. Он хорошо работает на разреженных данных и особенно полезен при работе с данными с большим числом функций. Однако он может сходиться медленно к концу обучения.

$$w = w - (learning\_rate / sqrt(sum\_of\_squared\_gradients)) * gradient$$

где w - веса, learning_rate - скорость обучения, gradient - градиент, sum_of_squared_gradients - сумма квадратов градиентов.

Adam - адаптивная оценка момента - это расширение Adagrad, которое использует адаптивные скорости обучения и импульс для ускорения сходимости процесса оптимизации. Он хорошо подходит для оптимизации масштабных нейронных сетей и является наиболее часто используемым алгоритмом оптимизации в глубоком обучении.

$$m = beta1 * m + (1 - beta1) * gradient$$

$$v = beta2 * v + (1 - beta2) * (gradient2)$$

$$m\_hat = m / (1 - beta1t)$$

$$v\_hat = v / (1 - beta2**t)$$

$$w = w - (learning\_rate / (sqrt(v\_hat) + epsilon)) * m\_hat$$

где w - веса, learning_rate - скорость обучения, gradient - градиент, beta1 и beta2

- параметры импульса, m и v - первый и второй моменты градиента, m_hat и v_hat - исправленные оценки моментов, t - текущий шаг, epsilon – маленькое число для стабилизации деления.

Nadam - это адаптивная оценка момента Нестеров с ускорением - это расширение Adam, которое включает в себя импульс Нестеров, технику, которая ускоряет сходимость, учитывая будущие обновления градиента. Надам известен тем, что он более устойчив, чем Adam, и было показано, что он достигает лучшей производительности на некоторых типах проблем, таких как классификация изображений [5].

$$m = beta1 * m + (1 - beta1) * gradient$$

$$v = beta2 * v + (1 - beta2) * (gradient2)$$

*m_hat = (beta1 * m) + ((1 - beta1) * gradient) / (1 - beta1t) v_hat = (beta2 * v) / (1 - beta2**t)*

*w = w - (learning_rate / (sqrt(v_hat) + epsilon)) * m_hat*

где w - веса, learning_rate - скорость обучения, gradient - градиент, beta1 и beta2

- параметры импульса, m и v - первый и второй моменты градиента, m_hat и v_hat - исправленные оценки моментов, t - текущий шаг, epsilon - маленькое число для стабилизации деления.

Создание модели, параметры обучения

Была создана модель Sequental(). Добавленныеслои (входной слой создается по умолчанию):

• Flatten – слой для «разворачивания» тензора – преобразует матрицу изображения в вектор.

• Dense, 256 нейронов, функция активации Relu 4

• Dense, 10 нейронов, функция активации Softmax – для классификации 10 цифр.

Для модели были выбраны следующие неизменяемые параметры:

• Функция потерь - categorical_crossentropy. Наиболее подходящая для задачи классификации. Имеет следующую формулу:

$$CCE(y,p) = -\sum_{c=1}^{N} y_{ic}\log(p_{ic})$$

Здесь N – число классов, y_{ic}– индикатор верного класса (1 или 0), p_{ic} – предсказанная вероятность принадлежности классу.

• Число эпох – 10

• Размер батча – 256

• Метрика – точность

В тестировании будет проведено сравнение работы с разными оптимизаторами и их параметрами.

Тестирование.

Список использованных вариантов оптимизаторов и их параметров, а также условные обозначения на графиках для них представлены в табл. 1.

Таблица 1 – Оптимизаторы

Оптимизатор	Параметры	Название графика
Adam	Коэф. Скорости обучения: 0.001	Adam,lr=0.001
Adam	Коэф. Скорости обучения: 0.01	Adam,lr=0.01
Adam	Коэф. Скорости обучения: 0.1	Adam,lr=0.1
Adagrad	Коэф. Скорости обучения: 0.001	Adagrad,lr=0.001
Adagrad	Коэф. Скорости обучения: 0.01	Adagrad,lr=0.01
Adagrad	Коэф. Скорости обучения: 0.1	Adagrad,lr=0.1
Nadam	Коэф. Скорости обучения: 0.001	Nadam,lr=0.001
Nadam	Коэф. Скорости обучения: 0.01	Nadam,lr=0.01
Nadam	Коэф. Скорости обучения: 0.1	Nadam,lr=0.1

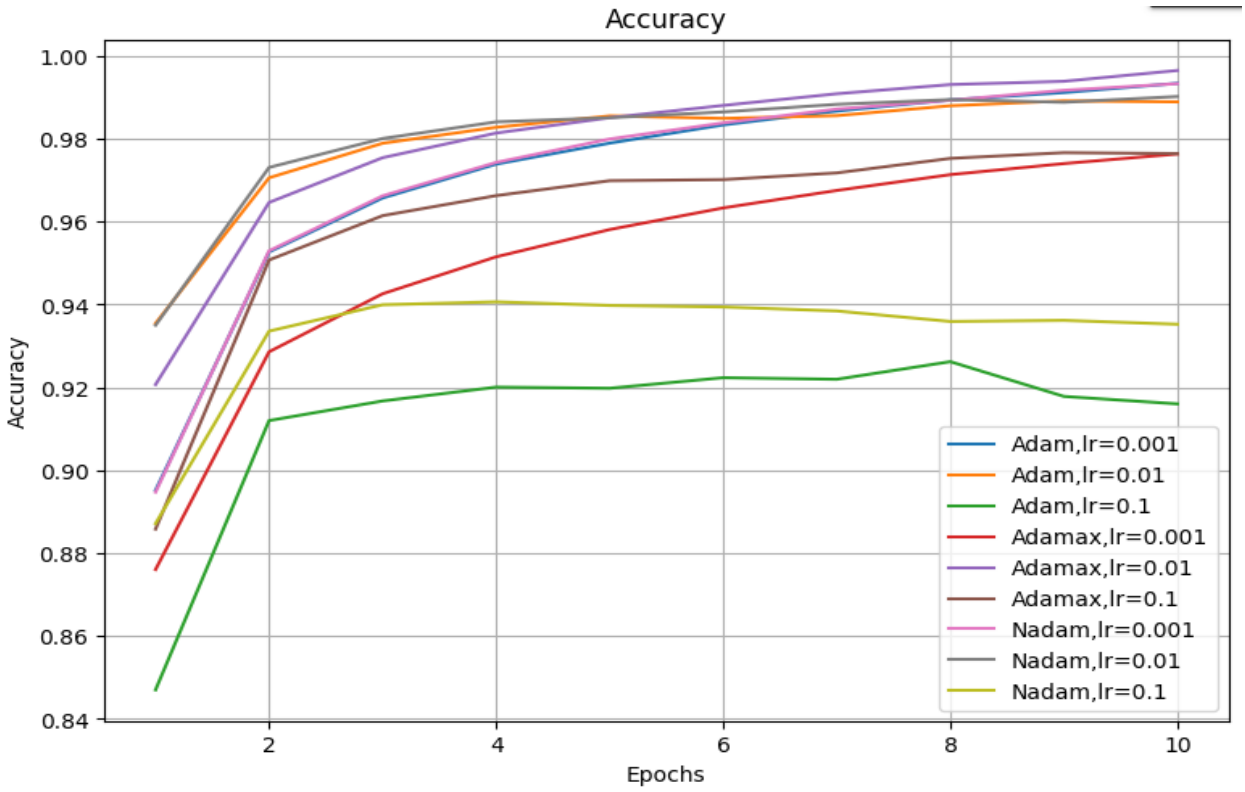

Рисунок 1 – Точность при различных оптимизаторах и их параметрах

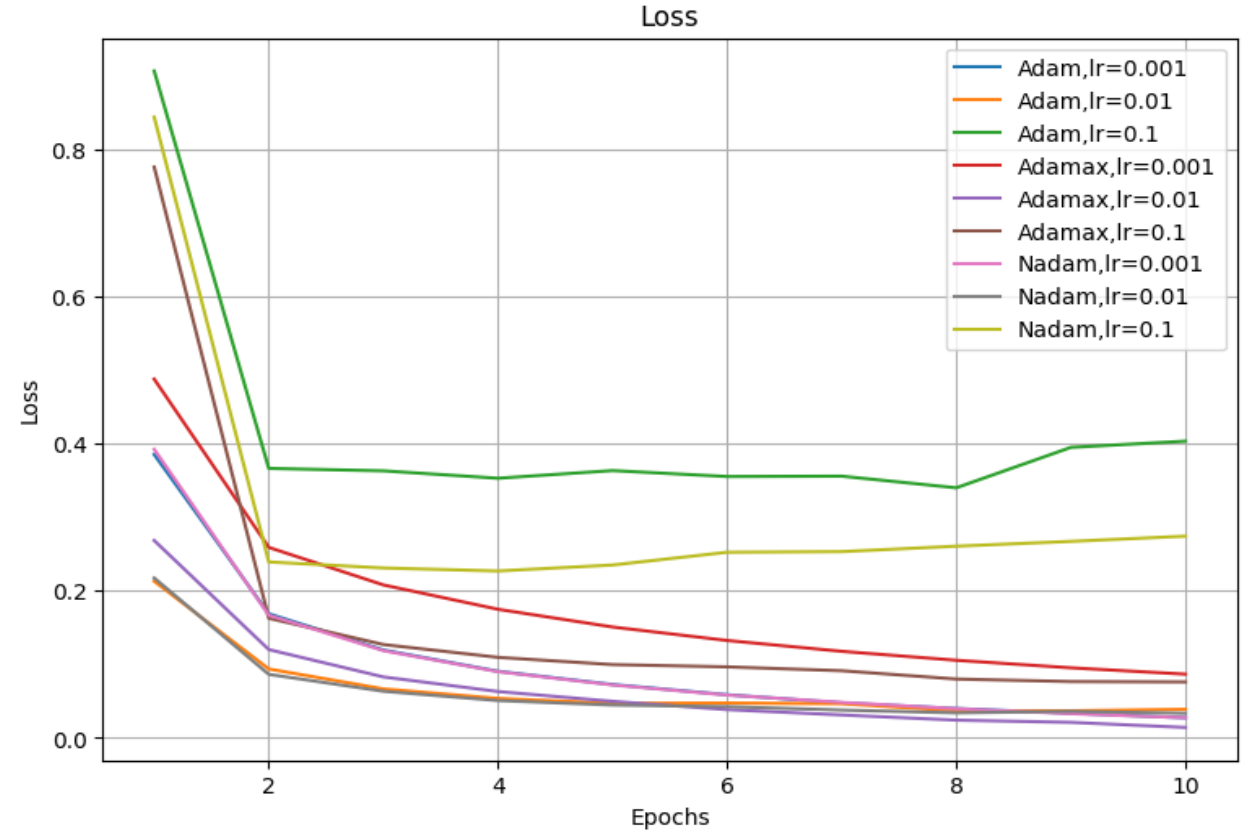

Рисунок 2 – Потери при различных оптимизаторах и их параметрах

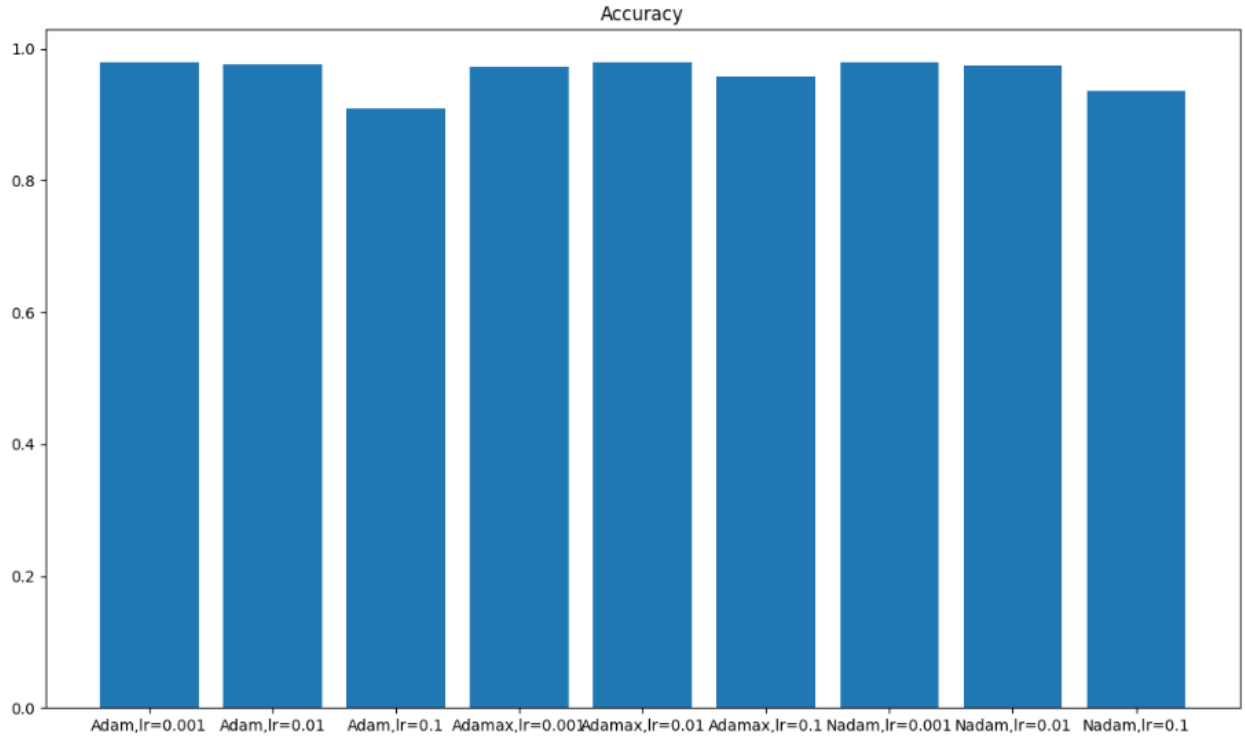

Рисунок 3 – Сравнение точности обученных моделей

В табл. 2. приведены показатели точности при проверке для каждого оптимизатора при различных параметрах и зеленым цветом отмечены наиболее оптимальные параметры.

Таблица 2 – Сравнение точности модели при различных оптимизаторах

Оптимизатор	Параметры	Accuracy
Adam	Коэф. Скорости обучения: 0.001	0.9782000184059143
Adam	Коэф. Скорости обучения: 0.01	0.9751999974250793
Adam	Коэф. Скорости обучения: 0.1	0.8982999920845032
Adagrad	Коэф. Скорости обучения: 0.001	0.8729000091552734
Adagrad	Коэф. Скорости обучения: 0.01	0.9319999814033508
Adagrad	Коэф. Скорости обучения: 0.1	0.974399983882904
Nadam	Коэф. Скорости обучения: 0.001	0.9782999753952026
Nadam	Коэф. Скорости обучения: 0.01	0.9740999937057495
Nadam	Коэф. Скорости обучения: 0.1	0.9193000197410583

Как видно из таблицы, для оптимизаторов Adam и Nadam наилучшее результаты модель показала при коэффициенте скорости обучения $Lr = 0.001$, но разница с результатами при $Lr = 0.01$ при учете различных начальных весов является совсем незначительной. Оптимизатор Adagrad показал лучшую точность при скорости обучения $Lr = 0.1$.

В целом, при удачном коэффициенте скорости обучения оптимизаторы Adam и Nadam показывают хороший результат с точностью около 98%. Оптимизатор Adagrad достигает точности 97%. Стоит отметить, что для оптимизаторов Adam и Nadam слишком большое значение $Lr = 0.1$ повлекло за собой довольно заметное ухудшение точности – скорее всего алгоритмы «перепрыгивают» точку минимума. Также на рис. 1 заметно проявление переобучения.

Ссылка на исходным кодом:
https://github.com/toychibeknorbutayev/Praktika/blob/main/nir3.ipynb

3. ОПИСАНИЕ ПРЕДПОЛАГАЕМОГО МЕТОДА РЕШЕНИЯ

На основе предоставленных данных оценки эффективности различных оптимизаторов и их параметров для нейронной сети, предполагается следующее решение в рамках диплома:

Архитектура Нейронной Сети:

Применение модели искусственной нейронной сети прямого распространения.

Использование библиотек TensorFlow и Keras для построения и обучения модели.

Выбор Оптимизаторов:

Использование оптимизаторов Adam, Adamax и Nadam с различными значениями learning rate.

Процесс сравнения влияния разных оптимизаторов на процесс обучения и точность классификации.

Обучение и Оценка Результатов:

Проведение обучения нейронной сети с различными параметрами, learning rate.

Оценка результатов с использованием метрик, таких как loss и accuracy.

4. ПЛАН РАБОТЫ НА ВЕСЕННИЙ СЕМЕСТР

По результатам выполнения заданий осеннего семестра составлен план работы на весенний семестр следующим образом.

Исследование параметров нейросетей с управляемыми элементами на болезненных состояний пациента наборах данных

- Выбор подходящей архитектуры нейросетей.
- Определение управляемых элементов в контексте болезненных состояний пациентов.
- Разработка методов управления параметрами нейросетей для анализа отдельных заболеваний.
- Проведение экспериментов на выбранных наборах данных.
- Оценка результата работы моделей на контрольных выборках.
- Сравнительный анализ эффективности управляемых элементов нейросетей

ЗАКЛЮЧЕНИЕ

В ходе выполнения этой работы были достигнуты все намеченные цели. С помощью библиотек TensorFlow и Keras была создана программная модель искусственной нейронной сети прямого распространения сигналов. Была создана и обучена модель, которая способна распознавать цифры на изображении с точностью более 98% на изображениях из MNIST. Также было проведено сравнение обучения модели при разных оптимизаторах и их параметрах.

Была найдена оптимальная конфигурация обучения сети, при которой сеть показывала максимальную точность.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Convergence‖ в Машинном обучении простыми словами (сайт) – https://dzen.ru/a/ZBbUFeljbQP4Gcmd
2. Исследование скорости обучения нейронный сетей – А.А. Рындин, В.П. Ульев
3. MNIST database – https://en.wikipedia.org/wiki/MNIST_database
4. «TensorFlow: Open source machine learning» Архивная копия от 15 декабря 2016 на Wayback Machine «It is machine learning software being used for various kinds of perceptual and language understanding tasks» — Jeffrey Dean
5. Методы оптимизации нейронных сетей – https://habr.com/ru/articles/318970/
6. Искусственные нейронные сети и их приложения – Ф.М. ГАФАРОВ, А.Ф. ГАЛИМЯНОВ
7. Исходным кодом https://github.com/toychibeknorbutayev/Praktika/blob/main/nir3.ipynb