Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Робототехника и комплексная автоматизация (РК)

КАФЕДРА Системы автоматизированного проектирования (РК6)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

«Применение свёрточных нейронных сетей (CNN) для решения задач технического анализа при управлении активами на фондовом рынке»

Студент РК6-76Б _____ **Онюшев А.А.**
(Подпись, дата) И.О. Фамилия

Руководитель курсового проекта _____ **Витюков Ф.А.**
(Подпись, дата) И.О. Фамилия

2024 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой РК6
А.П. Карпенко

«_____» _____ 20 ____ г.

З А Д А Н И Е
на выполнение курсового проекта

по дисциплине _____Модели и методы анализа проектных решений_____

Студент группы ____РК6-76Б____

_____Онюшев Артем Андреевич_____
(Фамилия, имя, отчество)

Тема курсового проекта: Изучение сверточных нейронных сетей для задач анализа биржи и стратегии торговли брокера_____
_____
_____

Направленность КП (учебный, исследовательский, практический, производственный, др.) учебный___
Источник тематики (кафедра, предприятие, НИР) __кафедра_____

График выполнения проекта: 25% к 5 нед., 50% к 11 нед., 75% к 14 нед., 100% к 16 нед.

Задание. Разработать архитектуру нейронной сети на основе сверток. Разработать датасет для загрузки необходимых данных в НС. Подобрать необходимые настройки и параметризации для НС. Обучить НС и провести исследование её пригодности для анализа биржи и стратегии торговли брокера_____

Оформление курсового проекта:

Расчетно-пояснительная записка на 22 листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.):
_____
_____
_____
_____

Дата выдачи задания «8» октября 2023 г.

Руководитель курсовой работы _____ **Витюков Ф.А.**
(Подпись, дата) И.О. Фамилия

Студент _____ **Онюшев А.А.**
(Подпись, дата) И.О. Фамилия

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

АННОТАЦИЯ

В данной работе рассмотрено, что такое сверточная нейронная сеть и чем она отличается от простого линейного перцептрона. Описана работа по настройке и подбору нужных нам функций-оптимизации, функций-потерь и функций активации, а также размера и количества нейронных слоев и сверточных слоев. Описана работа по составлению датасета на основе данных о тикере MSFT. Также в данной работе были проведены тесты с разными настройками и проанализированы результаты, полученные на выходе. Было проведено сравнение с результатами работы линейного перцептрона при схожих настройках.

В расчётно-пояснительной записке 28 страниц, 21 рисунок, 10 графических листов.

СОДЕРЖАНИЕ

ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

НС – Нейронная сеть.

CNN – Сверточная НС.

MLP – Линейный перцептрон.

Функция-потерь – Функция показывающая расстояние от предугаданного НС ответа до истинного ответа.

Функция-активации – Функция отвечающая за изменение весов синапсисов.

Тикер – Краткое название в биржевой информации котируемых инструментов (акций, облигаций, индексов).

EMA – (Exponential Moving Average) Экспоненциальная скользящая средняя. Один из показателей, помогающих при техническом анализе.

Stop-Loss – Биржевая заявка трейдера, в которой условием исполнения указано достижение цены, которая хуже, чем текущая рыночная.

Take-Profit – Биржевая заявка трейдера, которая выставляется заранее, чтобы в случае роста рынка зафиксировать прибыль по бумаге.

ВВЕДЕНИЕ

В настоящее время область сверточных нейронных сетей (CNN) активно исследуется и применяется в различных областях, включая финансовый рынок и биржевую торговлю. CNN позволяют анализировать и обрабатывать большие объемы данных, что идеально подходит для работы с финансовой информацией, которая обычно является многомерной и динамичной.

Использование CNN для анализа биржи и стратегий торговли брокера может быть эффективным способом прогнозирования рыночных движений и принятия обоснованных решений о покупке или продаже активов. Эти модели имеют преимущество в распознавании сложных закономерностей и паттернов на графиках цен, что может помочь в принятии более точных и выгодных решений.

Данное исследование будет сосредоточено на разработке и оптимизации сверточных нейронных сетей для анализа данных биржи и выработки стратегий торговли для брокеров. В этом исследовании рассмотрены различные подходы к построению моделей CNN, включая архитектуру сети, выбор оптимальных гиперпараметров и оптимизацию процесса обучения. Также будет проведен анализ полученных НС результатов для разных настроек.

Целью данного исследования является анализ пригодности CNN для выведения таких сложных закономерностей, как стоимости тикеров.

1. Устройство CNN

Сверточная нейронная сеть (ConvNet/CNN) — это специальная архитектура искусственных нейронных сетей, которая может принимать входное изображение (матрицу данных), присваивать важность (изучаемые веса и смещения) аспектам или объектам изображения (матрицы) и отличать одно от другого. При этом изображения (матрицы), в сравнении с другими архитектурами НС, требуют гораздо меньше предварительной обработки. В примитивных методах фильтры разрабатываются вручную, но достаточно обученные сети CNN учатся применять эти фильтры/характеристики самостоятельно.

Архитектура CNN аналогична структуре связей нейронов в мозгу человека, учёные черпали вдохновение в организации зрительной коры головного мозга. Отдельные нейроны реагируют на стимулы только в некоторой области поля зрения, также известного как перцептивное поле. Множество перцептивных полей перекрывается, полностью покрывая поле зрения CNN.

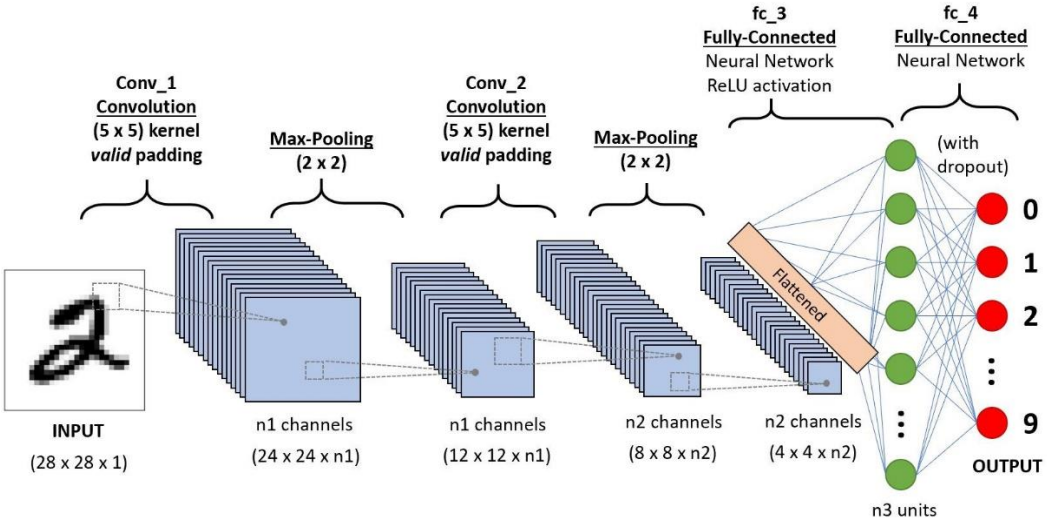


Рис. 1 Архитектура сверточной нейронной сети

Главной особенностью свёрточных сетей является то, что они обычно работают именно с матрицами данных, а потому можно выделить особенности, свойственные именно им. Многослойные персептроны работают с векторами, а потому для них нет никакой разницы, находятся ли какие-то точки рядом или на

противоположных концах, так как все точки равнозначны и считаются совершенно одинаковым образом. Изображения или же матрицы могут обладают локальной связностью. Например, если речь идёт об изображениях человеческих лиц, то вполне логично ожидать, что точки основных частей лица будут рядом, а не разрозненно располагаться на изображении. Поэтому требовалось найти более эффективные алгоритмы для работы с матрицами данных и ими оказались свёрточные сети.

2. Изображение «в глазах» компьютера

Изучая CNN невольно появляется вопрос «Что такое свертка?». Перед тем, как ответить на него, нужно разобраться, как компьютер «видит» какое-то изображение.

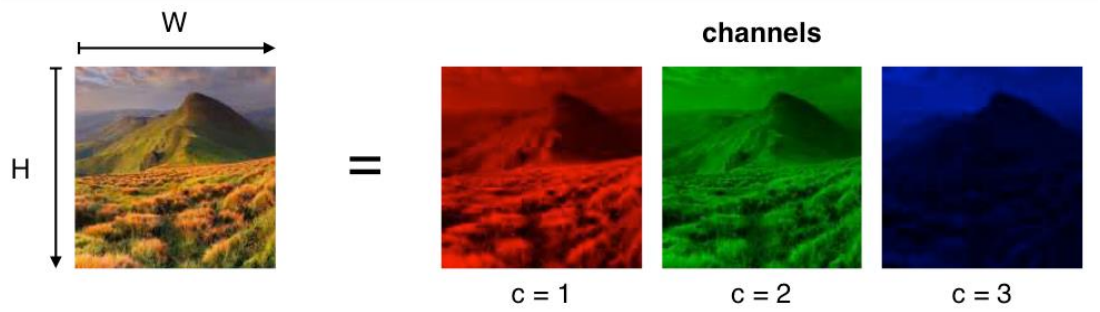


Рис. 2 Как компьютер видит изображение

Изображения в компьютере представляются в виде пикселей, а каждый пиксель – это значения интенсивности соответствующих цветовых каналов. При этом интенсивность каждого из каналов описывается целым числом от 0 до 255. Чаще всего используются цветные изображения, которые состоят из RGB пикселей – пикселей, содержащих яркости по трём каналам: красному, зелёному и синему. Различные комбинации этих цветов позволяют создать любой из цветов всего спектра. Чтобы хранить информацию о всех пикселях удобнее всего использовать тензор – 3D массив чисел, или, проще говоря, массив матриц чисел.

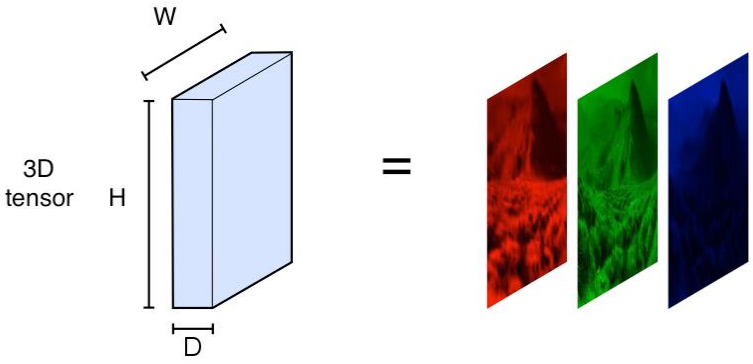
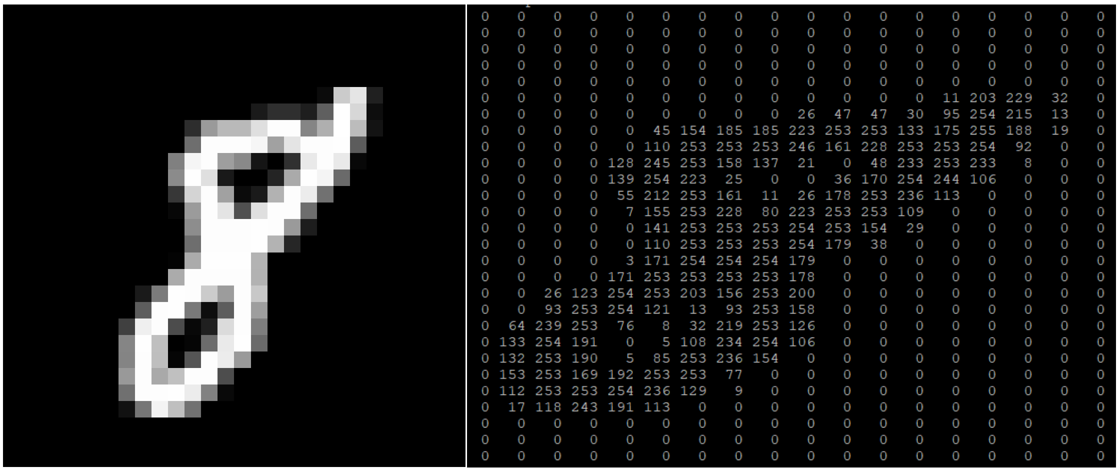


Рис. 3 Преобразование из изображения в тензор



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	11	203	229	32	0
0	0	0	0	0	0	0	0	0	26	47	47	30	95	254	215	13	0
0	0	0	0	0	45	154	185	185	223	253	253	133	175	255	188	19	0
0	0	0	0	0	110	253	253	253	246	161	228	253	253	254	92	0	0
0	0	0	0	128	245	253	158	137	21	0	48	233	253	233	8	0	0
0	0	0	0	139	254	223	25	0	0	36	170	254	244	106	0	0	0
0	0	0	0	55	212	253	161	11	26	178	253	236	113	0	0	0	0
0	0	0	0	7	155	253	228	80	223	253	253	109	0	0	0	0	0
0	0	0	0	0	141	253	253	253	254	253	154	29	0	0	0	0	0
0	0	0	0	0	110	253	253	253	254	179	38	0	0	0	0	0	0
0	0	0	0	3	171	254	254	254	179	0	0	0	0	0	0	0	0
0	0	0	0	171	253	253	253	253	178	0	0	0	0	0	0	0	0
0	0	26	123	254	253	203	156	253	200	0	0	0	0	0	0	0	0
0	0	93	253	254	121	13	93	253	158	0	0	0	0	0	0	0	0
0	64	239	253	76	8	32	219	253	126	0	0	0	0	0	0	0	0
0	133	254	191	0	5	108	234	254	106	0	0	0	0	0	0	0	0
0	132	253	190	5	85	253	236	154	0	0	0	0	0	0	0	0	0
0	153	253	169	192	253	253	77	0	0	0	0	0	0	0	0	0	0
0	112	253	253	254	236	129	9	0	0	0	0	0	0	0	0	0	0
0	17	118	243	191	113	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Рис 4. Изображение цифры и его представление в компьютере.

3. Свертка

Слой свёртки, как можно догадаться по названию типа нейронной сети, является самым главным слоем сети. Его основное назначение – выделить признаки на входном изображении и сформировать карту признаков. Карта признаков – это всего лишь очередной тензор (массив матриц), в котором каждая матрица отвечает за какой-нибудь выделенный признак.

Для того, чтобы слой мог выделять признаки, в нём имеются так называемые фильтры (или ядра). Фильтр – квадратная матрица небольшого размера (обычно 3x3 или 5x5), заполненная определенным образом. Чем больше будет таких фильтров – тем больше признаков удастся выделить и тем больше будет глубина каналов у выходного тензора слоя

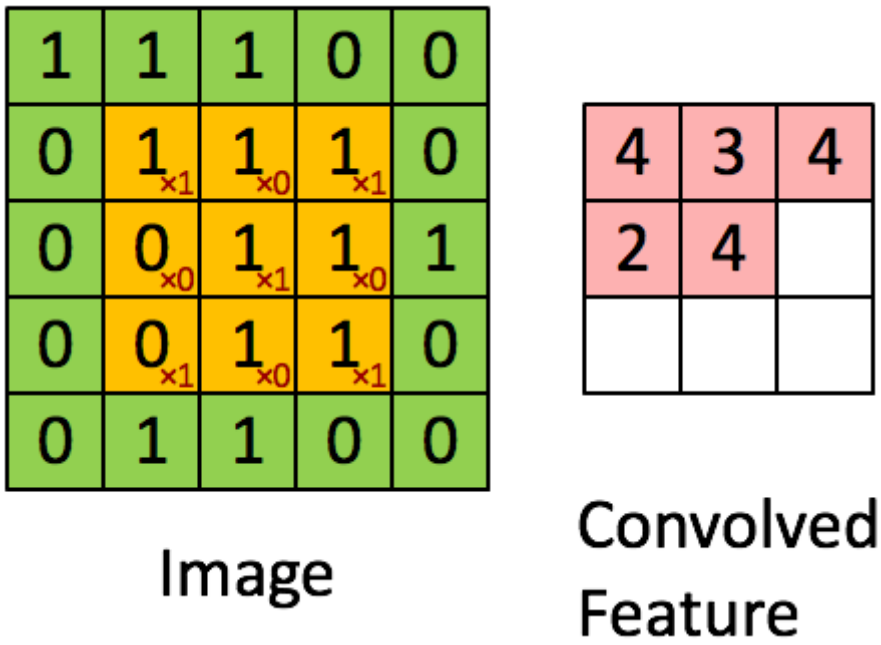


Рис. 5 Свертка изображения 5x5x1 с фильтром 3x3x1 для получения признака 3x3x1

Данный фильтр поочередно накладывается на изображение начиная с левого верхнего угла и пока не дойдет до правого нижнего угла. На каждом шаге числа в матрице фильтра перемножаются с соответствующими числами на изображении, полученные результаты складываются и записываются в матрицу признака. Лучше понять это можно, посмотрев на рисунок 5. На нем желтым цветом изображен фильтр размером 3x3x1, а красные цифры в правом углу –

числа матрицы фильтра. Зеленым цветом изображена матрица изображения 3x3x1, а красным полученный признак. Таким образом на 5-ом шаге итерации мы получим:

$$1 * 1 + 0 * 1 + 1 * 1 + 0 * 0 + 1 * 1 + 0 * 1 + 1 * 0 + 0 * 1 + 1 * 1 = 4$$

Полученные в итоге матрицы признаков могут иметь такой же размер, что и исходное изображение, либо размер меньше (как в нашем случае). Это зависит от заданных размера шага и начального заполнения. В этом репозитории можно найти множество разных GIF- файлов, которые помогут лучше понять, как заполнение и длина шага работают.

4. Слой подвыборки (пулинга)

Так как сверточные НС используют так же и простую сеть прямого распространения, нам необходимо уменьшить кол-во выходных параметров из слоя свертки и при этом не потерять важную информацию. Для этого используют слои подвыборки.

Данный слой позволяет уменьшить пространство признаков, сохраняя наиболее важную информацию. Существует несколько разных версий слоя пулинга, среди которых максимальный пулинг, средний пулинг и пулинг суммы. Наиболее часто используется именно слой макспулинга.

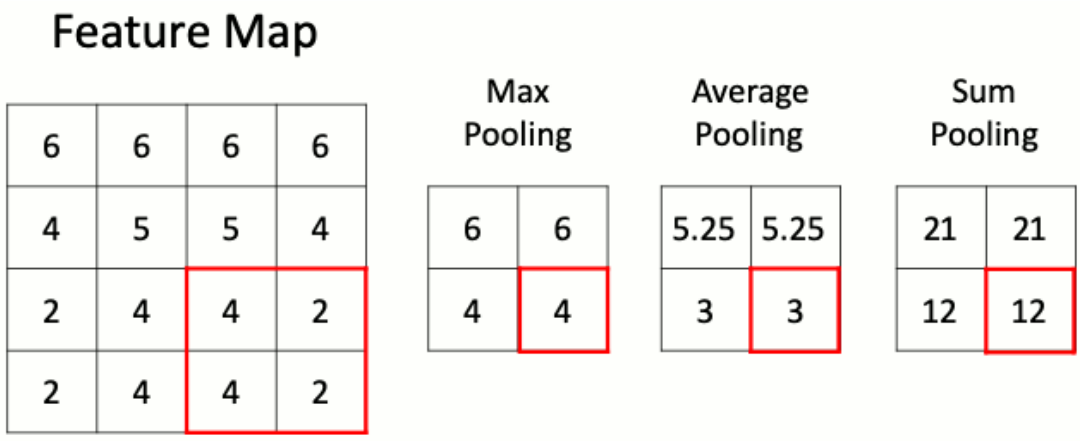


Рис. 6 Преобразования слоя подвыборки

Действия этого слоя идентичны действиям слоя свертки, только используются другие операции (для макспулинга – берется максимальное число, для среднего пулинга – берется среднее арифметическое от чисел).

5. Собственная CNN

Изучив, как устроена сверточная нейронная сеть, можно приступать к написанию собственной.

При разработке архитектуры нейронной сети требуется учитывать, что на вход должны передаваться данные некоторого размера, которые и будут проанализированы НС. В данной работе на вход НС передается информация о тикерах в формате: Массив[N] со значениями High, массив[N] со значениями Low, массив[N] со значениями EMA 200, где N – количество исследуемых дней от 40 до 2000, а также передается массив[N] со значениями asset – число отвечающее за наличие данного тикера в портфеле. Все эти 4 массива по N значений образуют матрицу 4×N. Её мы и будем использовать, как входные данные.

Проанализировав данные, поданные на вход НС, она возвращает некий результат своего анализа – выходные данные. При разработке архитектуры важно учесть правильность выходных данных. В данной работе выходными данными должно быть число в диапазоне от -0.5 до 1.5, означающие биржевые заявки трейдеру – take-profit или stop-loss.

Преобразуем нашу матрицу 4×N в удобный для обычных линейных нейронов формат, используя три слоя свертки с ядрами 2×2 и итоговым выходным количеством сверток 1×64. Далее с помощью скрытых слоев обычных линейных нейронов проанализируем получившиеся свертки. Было использовано 4 скрытых линейных слоя для лучшего анализа.

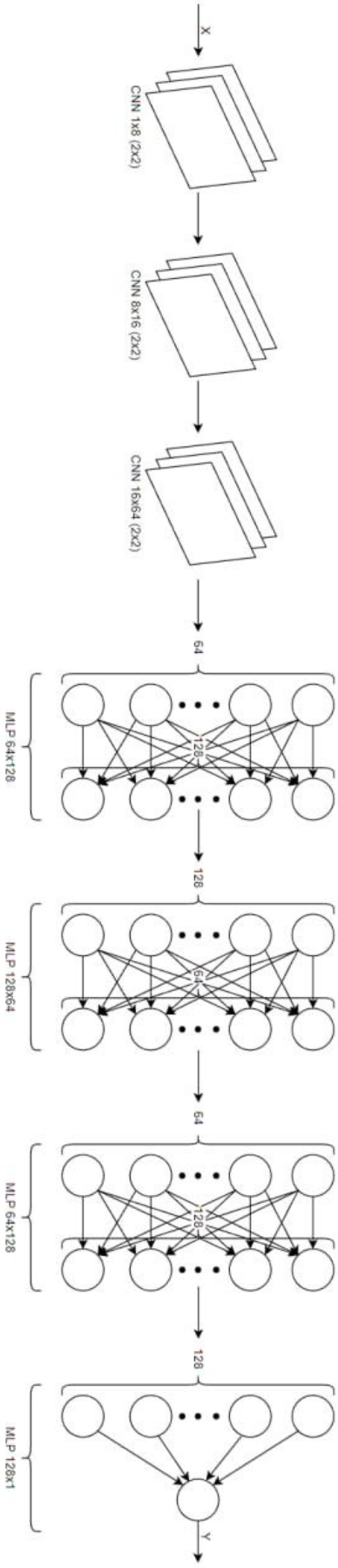


Рис. 7 Архитектура НС.

Реализация данной архитектуры в коде:

Листинг 1 – Класс нейронной сети.

```
class ConvNet(nn.Module):
    def __init__(self):
        super().__init__()

        self.act = nn.LeakyReLU(0.18) #функция активации
        self.conv0 = nn.Conv2d(1, 8, 2, stride=1, padding=0)
        # сверточный слой со сверточным ядром 2 на 2
        self.conv1 = nn.Conv2d(8, 16, 2, stride=1, padding=0)
        self.conv2 = nn.Conv2d(16, 64, 2, stride=1, padding=0)

        self.adaptivepool = nn.AdaptiveAvgPool2d((1, 1))
        self.flatten = nn.Flatten()
        self.linear1 = nn.Linear(64, 128)
        # линейный слой 64 на 128
        self.extra_linear_1 = nn.Linear(128, 64)
        # линейный слой 128 на 64
        self.extra_linear_2 = nn.Linear(64, 128)
        # линейный слой 64 на 128
        self.linear2 = nn.Linear(128, 1)
        # линейный слой 128 на 1


    def forward(self, x):
        # функция описывающая одну итерацию обучения НС
        out = self.conv0(x)
        out = self.act(out)
        out = self.conv1(out)
        out = self.act(out)
        out = self.conv2(out)
        out = self.act(out)

        out = self.adaptivepool(out)
        out = self.flatten(out)
        out = self.linear1(out)
        out = self.act(out)

        out = self.extra_linear_1(out)
        out = self.act(out)
        out = self.extra_linear_2(out)
        out = self.act(out)

        out = self.linear2(out)
        return out
```

6. Работа с yahoo finance

Для получения необходимой информации о дневных свечах тикера, необходимо обратится к API yahoo finance, но так как они прекратили его поддержку в 2016 году, воспользуемся модулем yfinance.

Чтобы выбрать конкретный тикер, создадим переменную ticker. Чтобы выбрать количество дней информацию по которым мы хотим получить, создадим переменную days. Чтобы выбрать количество дней для EMA введем переменную EMA_N. Реализация данного кода на python:

Листинг 2 – Функция загрузки информации с yahoo finance

```
1  Days = 200 # Кол-во дней в датасете
2  EMA_N = 200 # Кол-во дней в EMA
3  num_candle = Days+EMA_N
4
5  ticker = "MSFT" # Названия тикера который отслеживаем
6  hist = fin.download(ticker, period=f'{num_candle}d', interval='1d')
```

Функция fin.download() загрузит всю необходимую нам информацию о свечах, а код указанный в листинг 3 создаст столбец с информацией о EMA.

Листинг 3 – Создание столбца с EMA

```
1  hist[f'EMA{EMA_N}'] = hist['Close'].ewm(span=EMA_N,
   ajust=False).mean() # Считаем нужный EMA
```

Итоговая таблица будет иметь вид, указанный на рисунке 7.

Date	Open	High	Low	Close	EMA200	Assets	ake_profi	Stop_loss
07.12.2021	331,64	335,8	330,1	334,92	334,92	Na	Na	Na
08.12.2021	335,31	335,5	330,8	334,97	334,9205	Na	Na	Na
09.12.2021	334,41	336,49	332,12	333,1	334,9024	Na	Na	Na
10.12.2021	334,98	343	334,79	342,54	334,9784	Na	Na	Na
13.12.2021	340,68	343,79	339,08	339,4	335,0224	Na	Na	Na
14.12.2021	333,22	334,64	324,11	328,34	334,9559	Na	Na	Na
15.12.2021	328,61	335,19	324,5	334,65	334,9529	Na	Na	Na
16.12.2021	335,71	336,76	323,02	324,9	334,8528	Na	Na	Na
17.12.2021	320,88	324,92	317,25	323,8	334,7428	Na	Na	Na
20.12.2021	320,05	322,8	317,57	319,91	334,5953	Na	Na	Na
21.12.2021	323,29	327,73	319,8	327,29	334,5226	Na	Na	Na
22.12.2021	328,3	333,61	325,75	333,2	334,5094	Na	Na	Na
23.12.2021	332,75	336,39	332,73	334,69	334,5112	Na	Na	Na
27.12.2021	335,46	342,48	335,43	342,45	334,5902	Na	Na	Na

Рисунок 8. Вид полученной таблицы датасета

Получив такую таблицу, можно приступать к заполнению колонок «Stop-Loss» и «Take-Profit». Данные колонки заполняются брокером от руки в зависимости от принятой им стратегии ведения торгов. После заполнения этих колонок, наша таблица будет готова.

	Date	Open	High	Low	Close	EMA200	Assets	Take_profi	Stop_loss
249	02.12.2022	249,82	256,06	249,69	255,02	265,245	1	1,090888	0,671586
250	05.12.2022	252,01	253,82	248,06	250,2	265,0953	1	1,090888	0,680956
251	06.12.2022	250,82	251,86	243,78	245,12	264,8965	1	1,090888	0,671586
252	07.12.2022	244,83	246,16	242,21	244,37	264,6923	1	1,090888	0,671586
253	08.12.2022	244,84	248,74	243,06	247,4	264,5202	1	1,090888	0,711408
254	09.12.2022	244,7	248,31	244,16	245,42	264,3302	1	1,090888	0,739518
255	12.12.2022	247,45	252,54	247,17	252,51	264,2126	1	1,090888	0,762942
256	13.12.2022	261,69	263,92	253,07	256,92	264,14	0	0,149931	1,011487
257	14.12.2022	257,13	262,59	254,31	257,22	264,0711	0	0,149931	1,011487
258	15.12.2022	253,72	254,2	247,34	249,01	263,9213	0	0,149931	1,011487
259	16.12.2022	248,55	249,84	243,51	244,69	263,7299	0	0,149931	1,011487
260	19.12.2022	244,86	245,21	238,71	240,45	263,4983	0	0,149931	1,011487
261	20.12.2022	239,4	242,91	238,42	241,8	263,2824	0	0,149931	1,011487
262	21.12.2022	241,69	245,62	240,11	244,43	263,0948	0	0,149931	1,011487
263	22.12.2022	241,26	241,99	233,87	238,19	262,847	0	0,149931	1,011487

Рисунок 9. Итоговый вид таблицы датасета

7. Проведение анализов

Чтобы НС хорошо работала требуется обучить её на данных из всего датасета. Прогоняя эти данные снова и снова мы повышаем вероятность более правильного ответа НС. Прогон всего датасета через НС называется – эпоха. Совсем не обязательно в одной эпохе обучать НС всем датасетом за раз, можно разбить его на меньшие части. Такие части называют – бач данных.

Для исследования был составлен пробный датасет, который содержал 200 сущностей для обучения и 50 сущности для тестов. Размер бача равен 25 сущностей. Используемый код для обучения НС на Python:

Листинг 4 – Код обучающего цикла НС.

```
for epoch in range(epochs):
    loss_val = 0
    acc_val = 0
    for sample in train_loader:
        info, lbl = sample['info'], sample['label']
        # переносим требуемые данные на то устройство, где будем обучать НС (CPU или GPU)
        info = info.to(device)
        lbl = lbl.to(device)
        optimizer.zero_grad()

        with autocast(use_amp):
            pred = CNNet(info)
            loss = loss_fn(pred, lbl)
        # Считаем функцию-потерь
        scaler.scale(loss).backward()
        loss_item = loss.item()
        loss_val += loss_item

        # Обновляем веса
        scaler.step(optimizer)
        scaler.update()

        acc_current = accuracy_v3(pred.cpu().float(), lbl.cpu().float(), epsilon=epsilon)
        acc_val += acc_current

    print(f"Epoch : {epoch+1}")
    print(f"Loss : {loss_val / len(train_loader)}")
    print(f"Acc : {acc_val / (len(train_loader)*batch_size)}")

print(f'Full time learning : {time.time() - start_time}')
```

Исследуемая задача более всего похожа на задачу регрессии, основываясь на этом, максимально логичным решением будет использовать функцию

MSELoss (Mean Squared Error Loss) или MAELoss (Mean Absolute Error Loss), как функцию-потерь.

Формула MSE выглядит так:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = mean squared error
n = number of data points
Y_i = observed values
$\hat{Y}_i$ = predicted values

Рисунок 9. Формула подсчета MSE

Формула MAE выглядит так:

$$\text{MAE} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_1|}{n}$$

Рисунок 10. Формула подсчета MAE

После выбора подходящей функции-потерь, необходимо подобрать функцию-оптимизатор. На данный момент одной из самых популярных и эффективных функций-оптимизаторов является Adam (adaptive moment), он отлично подойдет и для этой задачи. Lr (шаг поиска) достаточно будет указать равным 10^{-3}. Отличный пример сравнения разных функций-оптимизаций приведен на данном сайте: https://emiliendupont.github.io/2018/01/24/optimization-visualization/

Подобрав все возможные необходимые настройки, можно приступать к тестированию.

Тестирование будем проводить и на CNN архитектуре, и на MLP. В качестве функций-потерь будем тестировать MSELoss и MAELoss. Также рассмотрим разное количество эпох (50, 100, 200, 400, 800, 1600, 3200, 6400, 10000). Все предложенные параметры нужны нам, чтобы четко понять как

архитектура НС, функция-потерь и количество эпох влияют на итоговый результат.

Полученные результаты представлены ниже. График красного цвета – требуемый (истинный) результат. График синего цвета – результат, предугаданный НС. По оси X – номера сущностей из тестового датасета. По оси У – “stop loss”.

Используя 50 эпох:

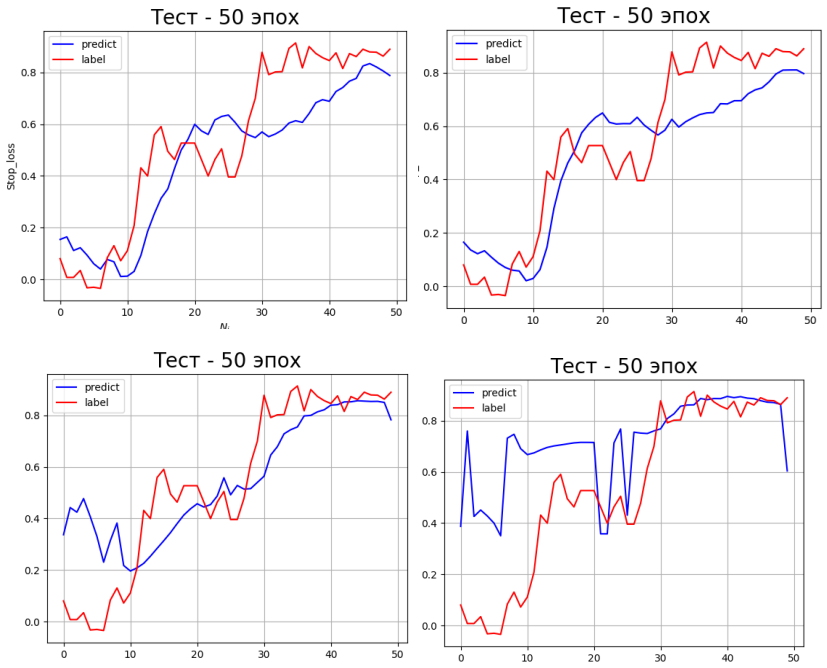
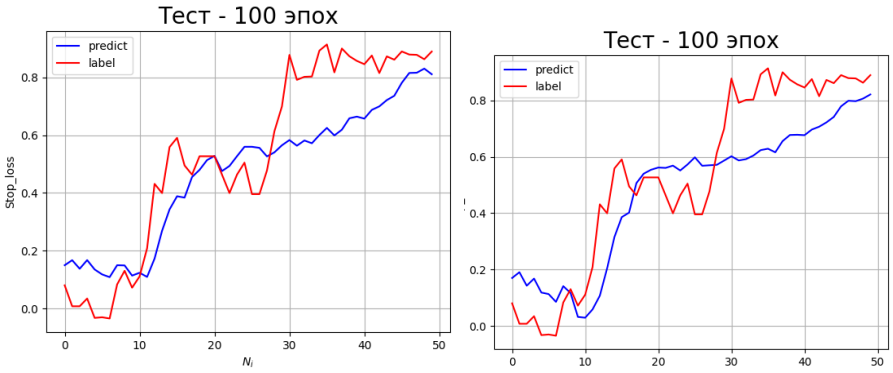


Рисунок 11. Тест с 50 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 100 эпох:



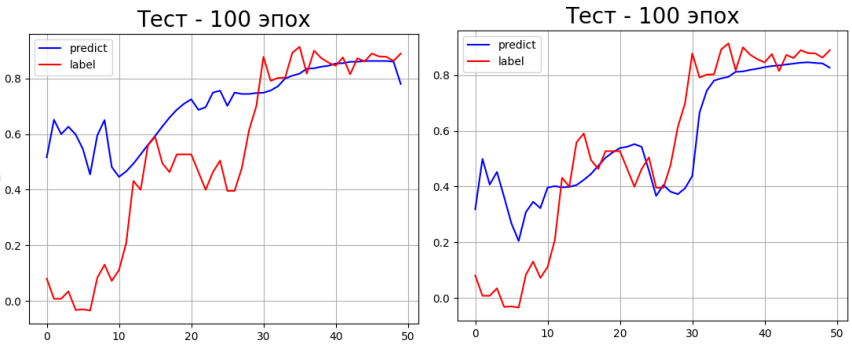


Рисунок 12. Тест с 100 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 200 эпох:

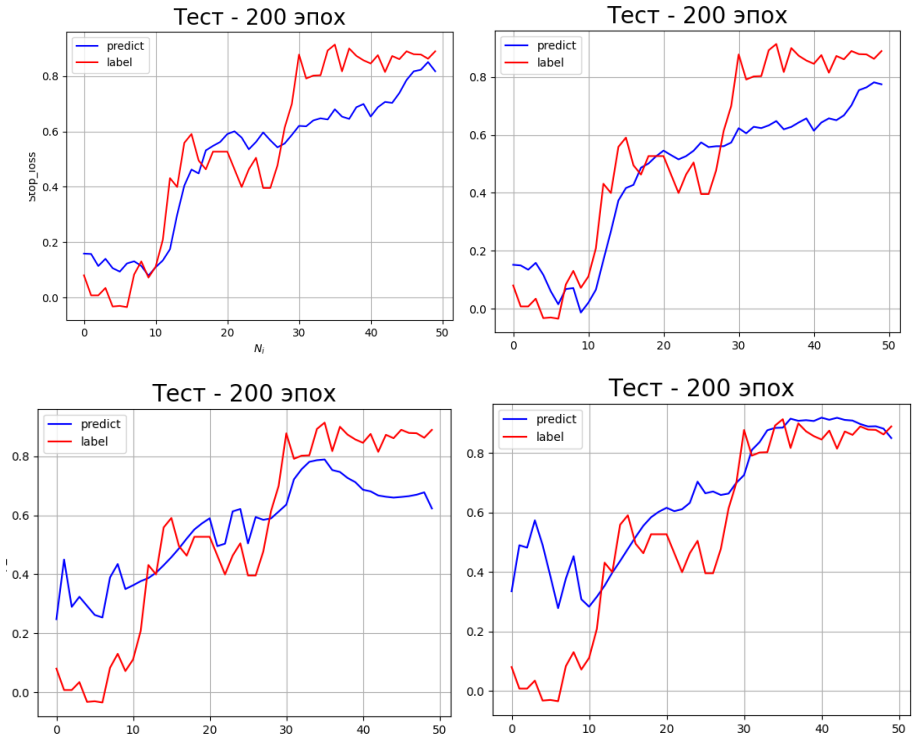
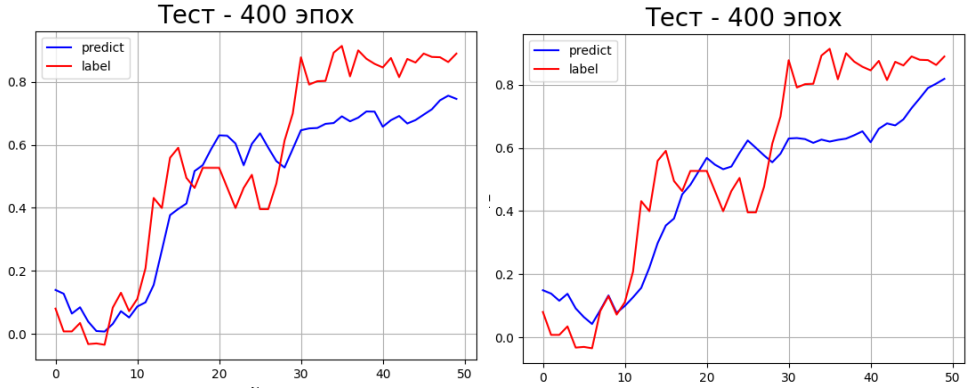


Рисунок 13. Тест с 200 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 400 эпох:



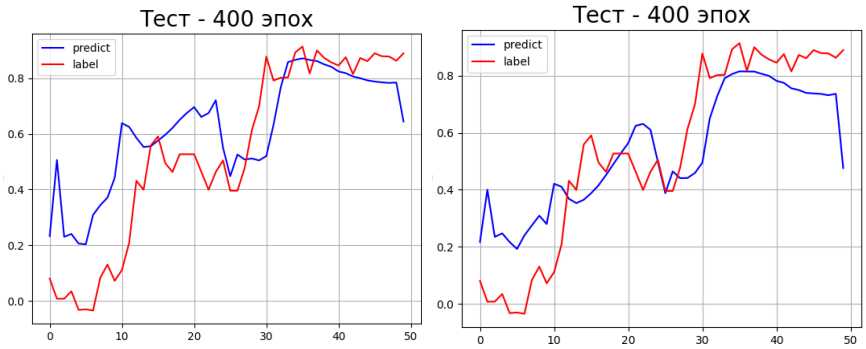


Рисунок 14. Тест с 400 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 800 эпох:

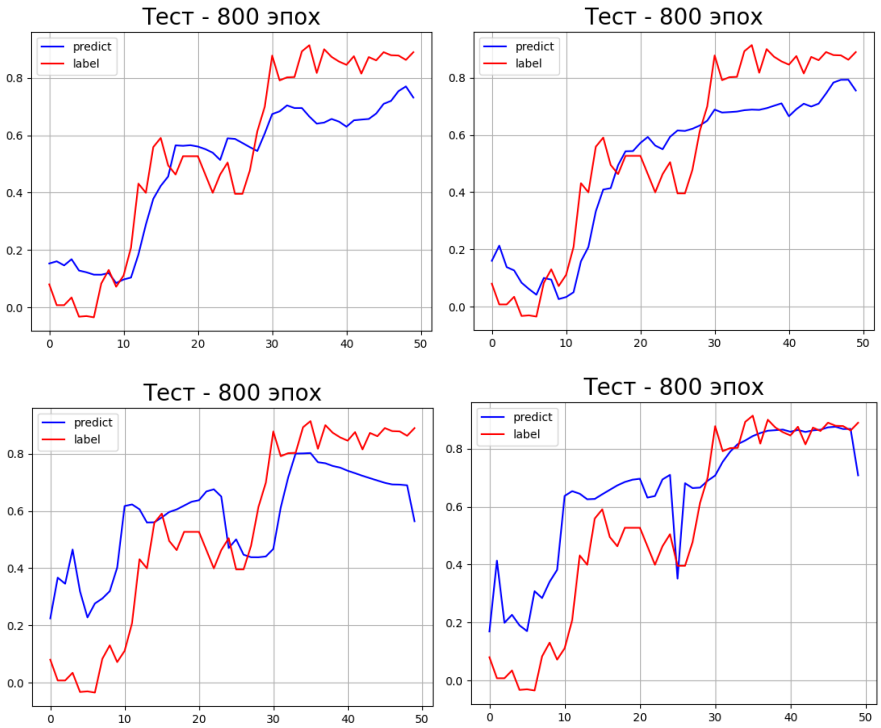
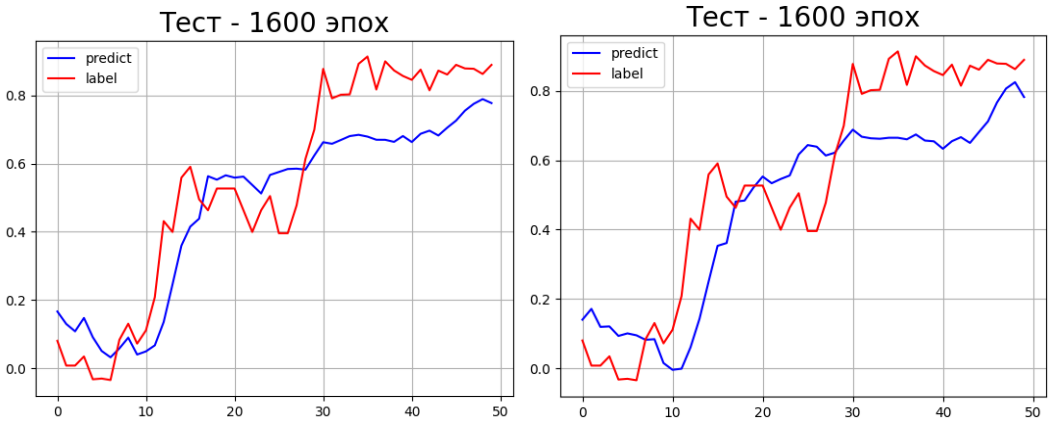


Рисунок 15. Тест с 800 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 1600 эпох:



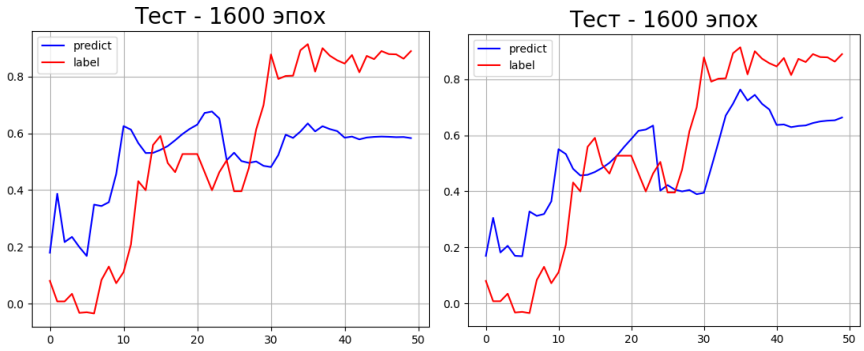


Рисунок 16. Тест с 1600 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 3200 эпох:

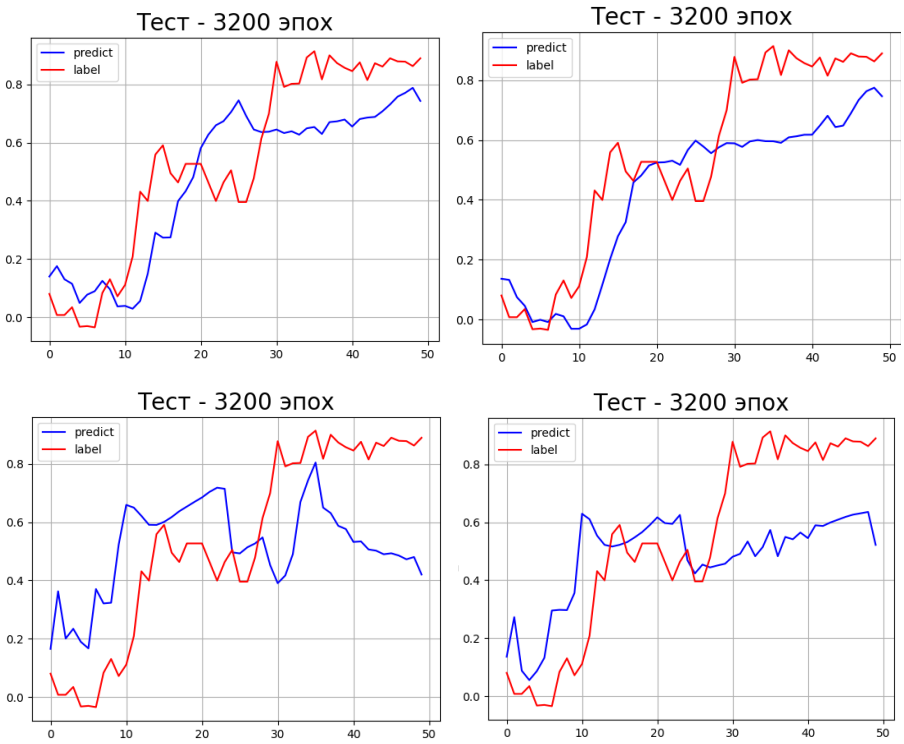
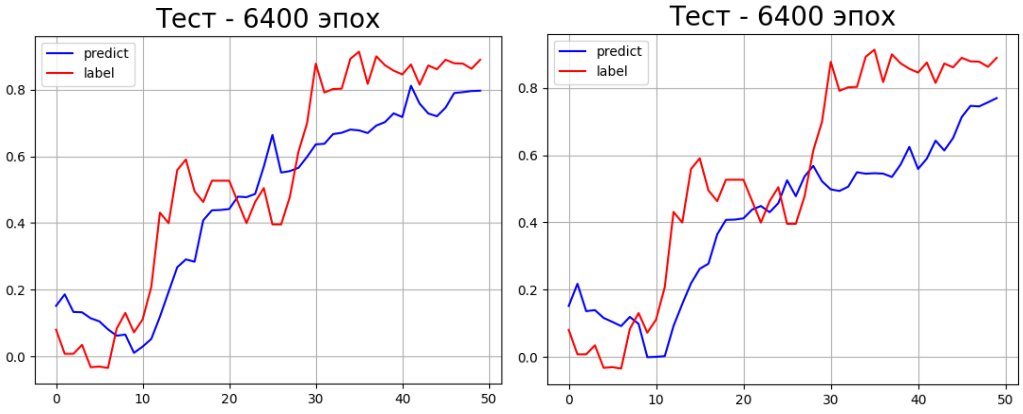


Рисунок 17. Тест с 3200 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 6400 эпох:



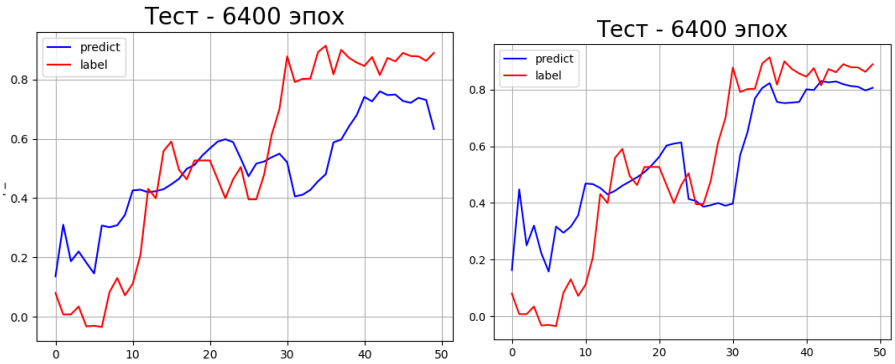


Рисунок 18. Тест с 6400 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Используя 10000 эпох:

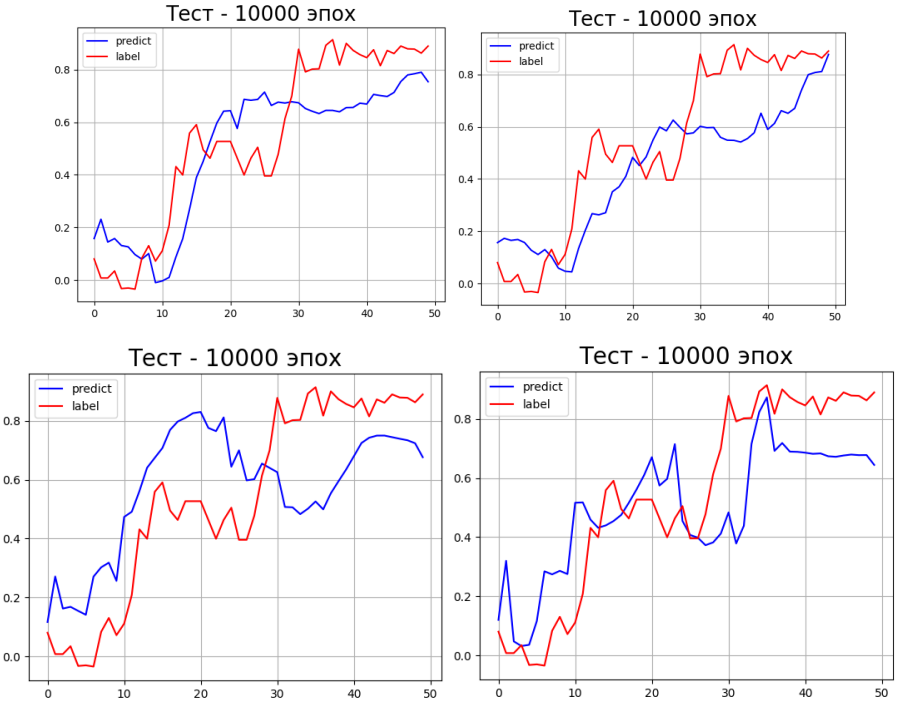


Рисунок 19. Тест с 10000 эпохами (слева – MSE, справа – MAE; сверху – MLP, Снизу – CNN).

Данные графики описывают результаты «предугадывания» НС в результате работы с тестовым датасетом (этот набор информации НС еще не видела и обучалась она не на нем).

MLP	Time	CNN	Time
50	1.0	50	3.484
100	2.0	100	4.673
200	3.586	200	8.708
400	6.720	400	13.0394
800	13.572	800	25.757
1600	27.80	1600	48.983
3200	56.043	3200	98.291
6400	195.29	6400	193.839
10000	180.47	10000	299.83

Рисунок 20. Зависимость времени обучения НС от количества эпох при использовании MSELoss.

MLP	Time	CNN	Time
50	1.0	50	3.895
100	2.0	100	4.762
200	3.455	200	8.210
400	7.208	400	13.410
800	14.6937	800	25.776
1600	28.3952	1600	51.802
3200	58.797	3200	102.44
6400	117.310	6400	202.842
10000	187.9521	10000	317.748

Рисунок 21. Зависимость времени обучения НС от количества эпох при использовании MAELoss.

По рисунку 20 и 21 можно увидеть, что время обучения НС от количества эпох зависит линейно. По остальным метрикам, которые представлены с точными значениями в файле «KP.xlsx», можно понять, что MLP обучается в 1.5 раза быстрее, чем CNN и имеет более точный результат и на этапе обучения, и на этапе тестирования. Также можно увидеть, что отсутствует сильная разница между использованием MSELoss или MAELoss. Обе функции справляются со своей задачей.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы были достигнуты следующие цели:

1) Рассмотрено, что такое сверточная нейронная сеть
2) Чем сверточная сеть отличается от простого линейного перцептрона
3) Описана работа по составлению датасета на основе данных о тикере MSFT
4) Описана работа по настройке и подбору нужных функции-оптимизации, функции-потерь, функции-активации и других настроек архитектуры НС.
5) Были проведены анализы работы сверточной НС при разных параметрах
6) Было проведено сравнение с работой линейного перцептрона при разных параметрах

Результаты экспериментов оказались достаточно «живые» и говорят о том, что сверточные нейронные сети способны анализировать не только изображения, но и большие матрицы данных. Однако биржа является предметом с очень сложными зависимостями и закономерностями и CNN не имеет возможностей точно предугадать следующее поведение биржи.

Оптимизация параметров модели и правильный выбор архитектуры CNN существенно влияют на достижение высокой точности классификации. Также важным аспектом является качество предварительной обработки данных, включая масштабирование, аугментацию и нормализацию, что дополнительно улучшает способность модели к обобщению.

Полученные результаты предоставляют основу для дальнейших исследований и применения сверточных нейронных сетей в различных областях. Возможным продолжением для данной работы, является проведение тестовых торгов на основе предсказанных нейронной сетью результатов. Таким способом можно будет корректно проанализировать прибыльность использования такой сети и ее актуальность.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. «Сверточная нейронная сеть с нуля» [Электронный ресурс] – URL: https://programforyou.ru/poleznoe/convolutional-network-from-scratch-part-zero-introduction (дата обращения: 20.10.2023 - 24.10.2023);
2. Блог о сверточной нейронной сети [Электронный ресурс] – URL: https://habr.com/ru/companies/skillfactory/articles/565232/ (дата обращения: 20.10.2023);
3. PyTorch Tutorials [Электронный ресурс] – URL: https://pytorch.org/tutorials/ (дата обращения: 20.10.2023 - 30.10.2023);
4. PyTorch Documentation [Электронный ресурс] – URL: https://pytorch.org/docs/stable/index.html (дата обращения: 20.10.2023 - 30.10.2023);
5. Сайт с датасетами [Электронный ресурс] – URL: https://www.kaggle.com/datasets/d4rklucif3r/cat-and-dogs?select=dataset (дата обращения: 21.10.2023).