



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 ЗАПISКА К КУРСОВОМУ ПРОЕКТУ

**НА ТЕМУ:**

***«Изучение сверточных нейронных сетей для задач анализа  
биржи и стратегии торговли брокера »***

Студент РК6-76Б

\_\_\_\_\_  
(Подпись, дата)

**Онюшев А.А.**

И.О. Фамилия

Руководитель курсового проекта

\_\_\_\_\_  
(Подпись, дата)

**Витюков Ф.А.**

И.О. Фамилия

2023 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 кастомный датасет для загрузки необходимых данных в НС. Подобрать необходимые настройки и параметризацию 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**

\_\_\_\_\_  
(Подпись, дата)

**Витюков Ф.А.**

И.О. Фамилия

**Студент**

\_\_\_\_\_  
(Подпись, дата)

**Онюшев А.А.**

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. Также в данной работе будут проведены тесты с разными настройками и проанализированы результаты, полученные на выходе.

В расчётно-пояснительной записке 32 страницы, 21 рисунок, 3 графических листа.

## СОДЕРЖАНИЕ

АННОТАЦИЯ.....	3
ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ .....	5
ВВЕДЕНИЕ.....	6
1. Устройство CNN .....	7
2. Изображение «в глазах» компьютера .....	9
3. Свертка.....	11
4. Слой подвыборки (пулинга) .....	13
5. Собственная CNN .....	14
ЗАКЛЮЧЕНИЕ .....	19
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	20

## ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

**НС** – Нейронная сеть

**CNN** – Сверточная НС

**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)

**ЕМА** – (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.

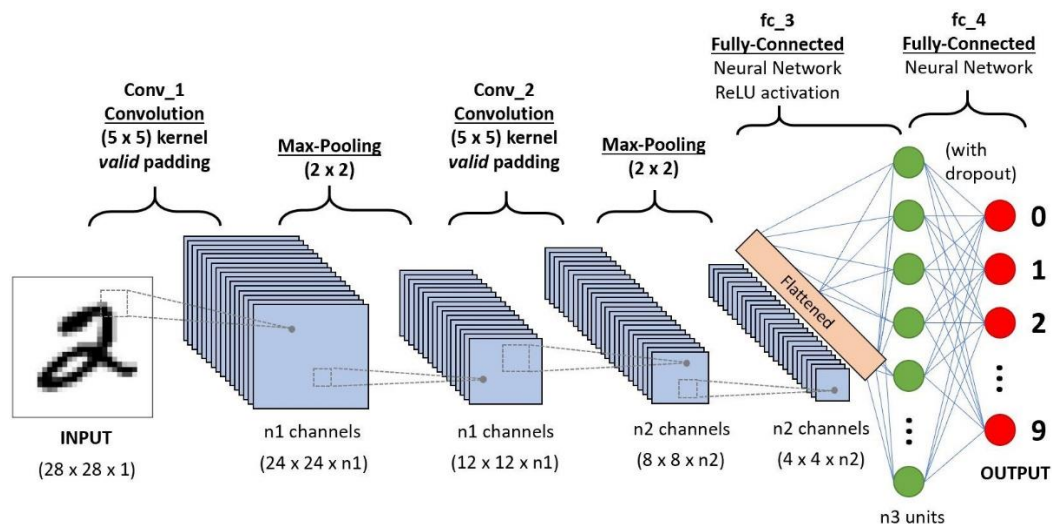


Рис. 1 Архитектура сверточной нейронной сети

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.

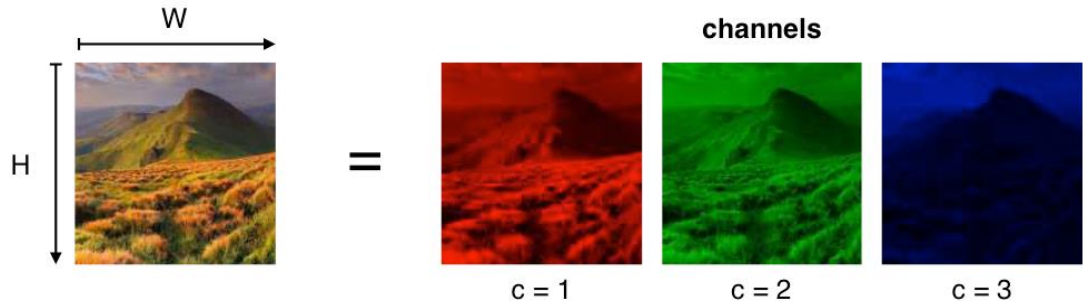


Рис. 2 Как компьютер видит изображение

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.

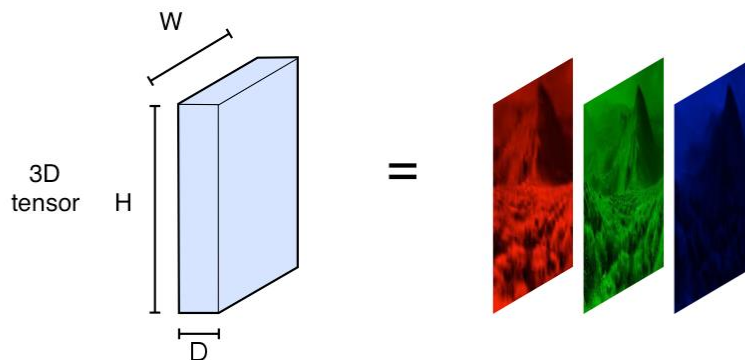


Рис. 3 Преобразование из изображения в тензор



### 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

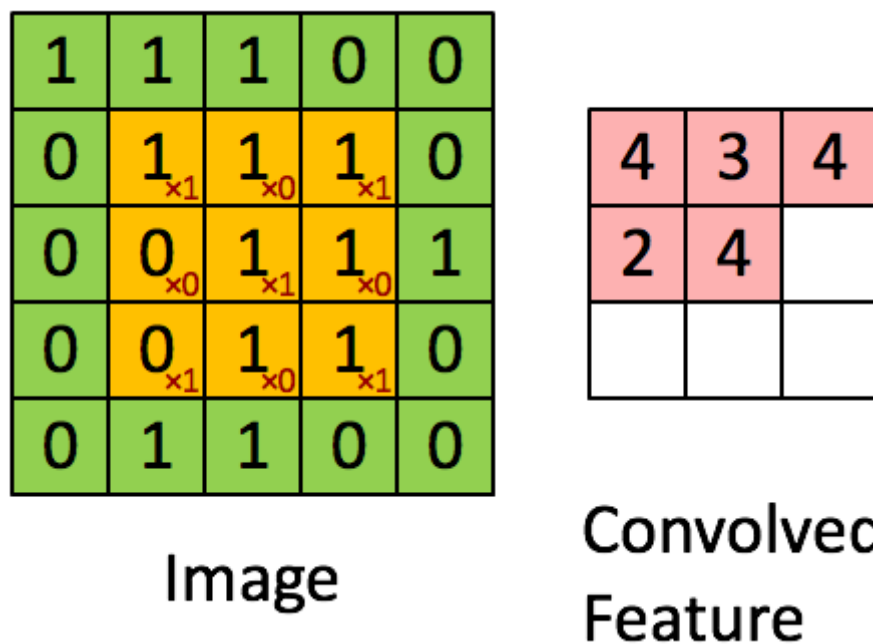


Рис. 5 Свертка изображения 5x5x1 с фильтром 3x3x1 для получения признака 3x3x1

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  $3 \times 3 \times 1$ , 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.

Feature Map				Max Pooling		Average Pooling		Sum Pooling	
6	6	6	6	6	6	5.25	5.25	21	21
4	5	5	4	4	4	3	3	12	12
2	4	4	2						
2	4	4	2						

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 приступить 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  $4 \times N$ . 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 должны быть два числа в диапазоне от -0.5 до 1.5, означающие биржевые заявки трейдеру – take-profit и stop-loss.

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

Реализация данной архитектуры в коде представлена в листинг 1.

## 7. Работа с yahoo finance

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

Чтобы выбрать конкретный тикер, создадим переменную ticker (смотреть листинг 2). Чтобы выбрать количество дней информацию по которым мы хотим получить, создадим переменную days (смотреть листинг 2). Чтобы выбрать количество дней для ЕМА введем переменную ЕМА\_N (смотреть листинг 2).

Функция fin.download() (смотреть листинг 2) загрузит всю необходимую нам информацию о свечах, а код указанный в листинг 3 создаст столбец с информацией о ЕМА.

Итоговая таблица будет иметь вид, указанный на рисунке 7.

Date	Open	High	Low	Close	EMA200	Assets	Take_profit	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 7. Итоговый вид 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.

## 8. 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 – батч данных.

Для исследования был составлен пробный датасет, который содержал 200 сущностей для обучения и 50 сущности для тестов. Размер батча равен 50 сущностей. Количество эпох равно 50. Используемый код для обучения НС (смотреть листинг 4).

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 8. Формула подсчета MSE

Формула MAE выглядит так:

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

Рисунок 9. Формула подсчета MSE



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 (смотреть листинг 8). 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 приступить к тестированию.

Используя 50 эпох:

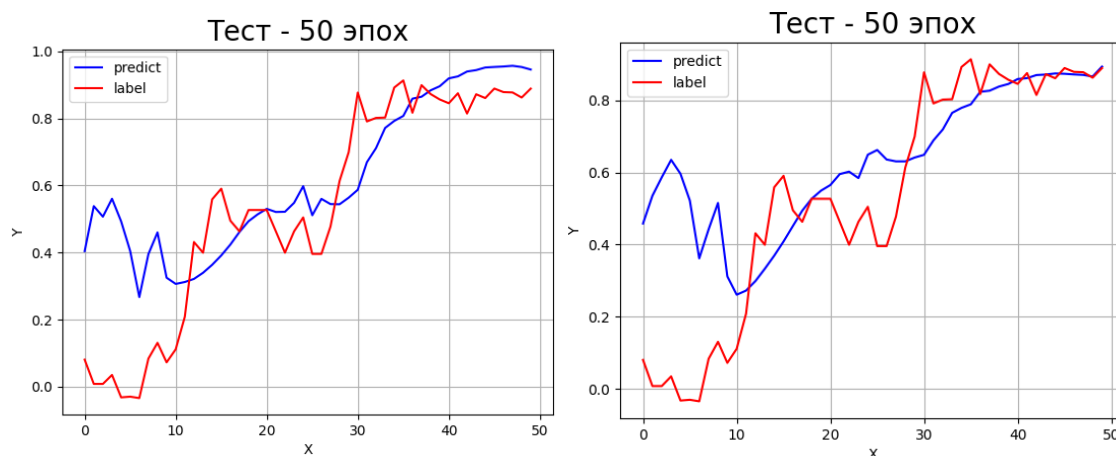


Рисунок 10. Тест с 50 эпохами(слево – MSE, справа – MAE).

Используя 150 эпох:

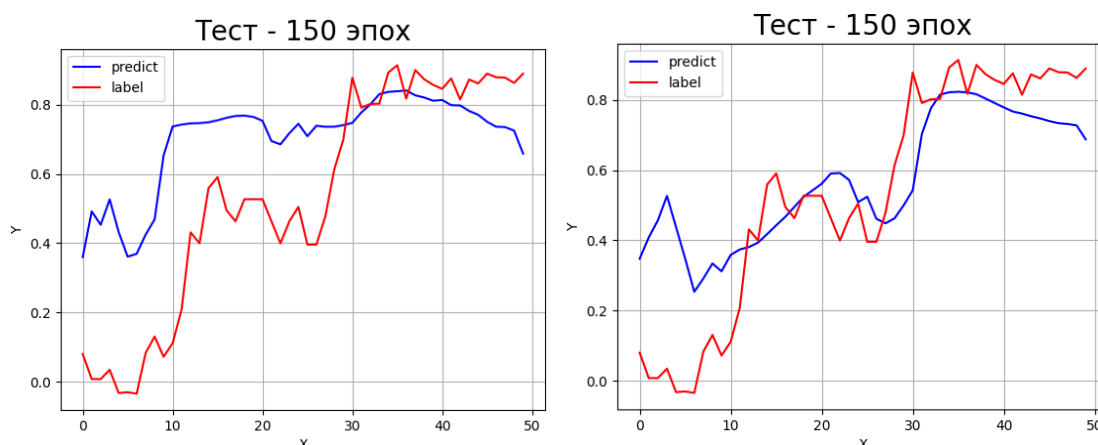


Рисунок 11. Тест с 150 эпохами(слево – MSE, справа – MAE).

Используя 250 эпох:

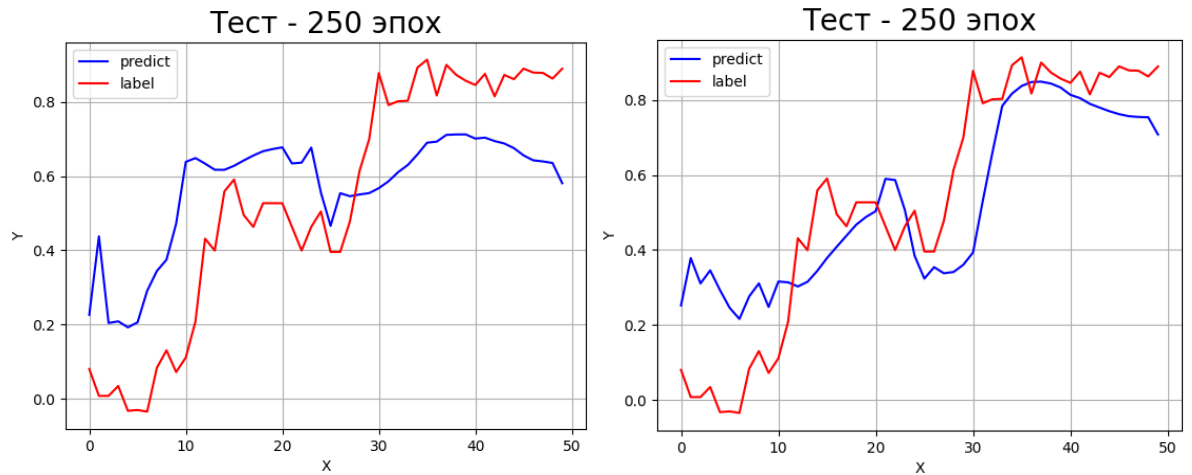


Рисунок 12. Тест с 250 эпохами(слево – MSE, справа – MAE).

Используя 450 эпох:

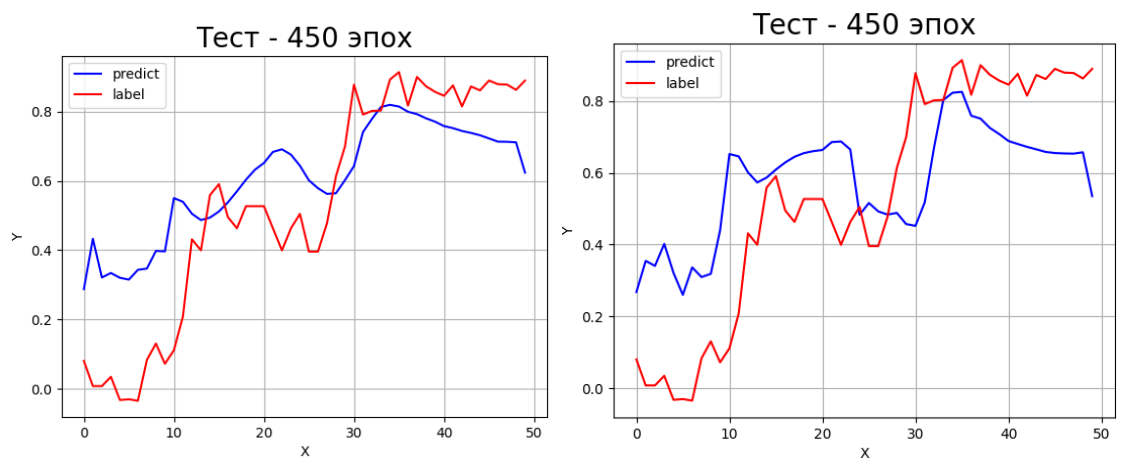


Рисунок 13. Тест с 450 эпохами(слево – MSE, справа – MAE).

Используя 8000 эпох:

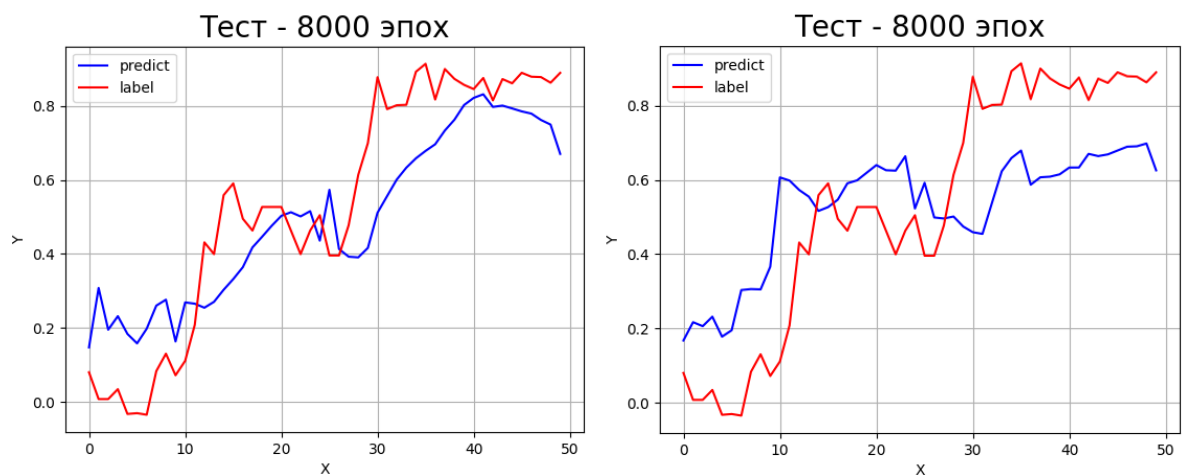


Рисунок 13. Тест с 8000 эпохами(слево – MSE, справа – MAE).

## ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы можно утверждать, что сверточные нейронные сети (CNN) представляют собой мощный инструмент для задач разного типа связанных с анализом больших матриц данных. В процессе проведения исследования были изучены различные архитектуры CNN, методы предварительной обработки данных и оптимизации параметров модели. 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).

## Приложение

### Листинг 1 – Класс нейронной сети.

```
1 class ConvNet(nn.Module):
2     def __init__(self):
3         super().__init__()
4
5         self.act = nn.LeakyReLU(0.18)
6         self.conv0 = nn.Conv2d(1, 8, 2, stride=1, padding=0)
7         self.conv1 = nn.Conv2d(8, 16, 2, stride=1, padding=0)
8         self.conv2 = nn.Conv2d(16, 64, 2, stride=1, padding=0)
9
10        self.adaptivepool = nn.AdaptiveAvgPool2d((1, 1))
11        self.flatten = nn.Flatten()
12        self.linear1 = nn.Linear(64, 128)
13        self.extra_linear_1 = nn.Linear(128, 64)
14        self.extra_linear_2 = nn.Linear(64, 128)
15        self.linear2 = nn.Linear(128, 1)
16
17    def forward(self, x):
18        out = self.conv0(x)
19        out = self.act(out)
20        out = self.conv1(out)
21        out = self.act(out)
22        out = self.conv2(out)
23        out = self.act(out)
24
25        out = self.adaptivepool(out)
26        out = self.flatten(out)
27        out = self.linear1(out)
28        out = self.act(out)
29
30        out = self.extra_linear_1(out)
31        out = self.act(out)
32        out = self.extra_linear_2(out)
33        out = self.act(out)
34
35        out = self.linear2(out)
36        return out
```

### 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',
7 interval='1d')
```

### Листинг 3 – Создание столбца с ЕМА

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

#### Листинг 4 – Класс нейронной сети.

```
1  for epoch in range(epochs):
2      loss_val = 0
3      acc_val = 0
4      for sample in train_loader:
5          info, lbl = sample['info'], sample['label']
6          info = info.to(device)
7          lbl = lbl.to(device)
8          optimizer.zero_grad()
9
10         with autocast(use_amp):
11             pred = CNNNet(info)
12             loss = loss_fn(pred, lbl)
13             scaler.scale(loss).backward()
14             loss_item = loss.item()
15             loss_val += loss_item
16
17         scaler.step(optimizer)
18         scaler.update()
19
20         acc_current = accuracy_v3(pred.cpu().float(),
21                                   lbl.cpu().float(), epsilon=epsilon)
22         acc_val += acc_current
23
24         print(f"Epoch : {epoch+1}")
25         print(f"Loss : {loss_val / len(train_loader)}")
26         print(f"Acc : {acc_val / (len(train_loader)*batch_size)}")
27     print(f'Full time learning : {time.time() - start_time}')
```

#### Листинг 5 – Функция подсчета точности

```
1  def accuracy_v3(pred, label, epsilon=0.0001):
2      count = 0
3      predict = pred.detach().numpy()
4      lbl = label.detach().numpy()
5      for i in range(len(label)):
6          accuracy_now = abs(lbl[i]-predict[i])
7          if accuracy_now < epsilon:
8              count += 1
9      return count
```