Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Програмної інженерії
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження ефективності використання нейромережевих методів і моделей для підвищення точності прогнозування вихідних показників об'єктів
(тема)

Виконав: студент 2 курсу, групи ІПЗм-17-1
спеціальності 121 – Інженерія програмногозабезпечення
(код і повна назва спеціальності)
Освітньо-наукової програми
Інженерія програмного забезпечення
(повна назва спеціалізації)

Мінаков А. Г.
(прізвище, ініціали)

Керівник проф. Лєсна Н. С.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. Кафедри _____ Дудар З. В.
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук

Кафедра програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121– Інженерія програмного забезпечення
(код і повна назва)

Освітньо-наукова програма Інженерія програмного забезпечення
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри_____
(підпис)
«_____»_____ 20 __р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

Студентові Мінакову Артему Геннадійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження ефективності використання нейромережевих методів і моделей для підвищення точності прогнозування вихідних показників об'єктів затверджена наказом по університету від «18» квітня 2019р № 546 Ст
2. Термін подання студентом роботи до екзаменаційної комісії «24» червня 2019 р.
3. Вихідні дані до роботи *методи для підвищення точності проогнозування вихідних даних, пояснювальна записка. Використовувати ОС Windows, мова програмування Python, середовище розробки PyCharm.*
4. Перелік питань, що потрібно опрацювати в роботі *позначка роботи, аналіз проблемної галузі і постановка задачі, опис проблем методів прогнозування, використовувані методи та алгоритми, опис розробленої програмної системи, аналіз можливих застосувань*
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *Мета завдання, обґрунтування доцільності розроблення, постановка задачі, об'єктна модель системи, базові моделі, інтерфейс програмної системи, результати тестування програмної системи, демонстраційні матеріали*

6 Консультанти розділів роботи

Найменування Розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	проф. Лєсна Н.С.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка*
1.	Аналіз предметної галузі	22 квітня 2019р.	
2.	Опрацювання першоджерел та постановка задачі	10 травня 2019р.	
3.	Розробка методики та моделі	20 травня 2019р.	
4.	Оформлення результатів дослідження	28 травня2019р.	
5.	Підготовка пояснювальної записки	5 червня 2019р.	
6.	Підготовка презентації та доповіді	15 червня 2019р.	
7.	Перевірка роботи на антиплагіат	16 червня 2019р.	
8.	Нормоконтроль	18 червня 2019р.	
9.	Рецензування	20 червня 2019р.	
10.	Занесення атестаційної роботи в електронний архів	21 червня 2019р.	
11.	Попередній захист	22 червня 2019р.	
12.	Допуск до захисту у зав. кафедри	22 червня 2019р.	
* заповнюється вручну після виконання чергового пункту			

Дата видачі завдання 22_квітня 2019 р.

Студент _____
(підпис)

Керівник роботи _____ __проф. Лєсна Н.С._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до атестаційної роботи: 85 с., 24 рис., 3 табрл., 4 додатки, 32 джерела.

KERAS, DATA MINING, ПРОГНОЗУВАННЯ ФІНАНСОВИХ ЧАСОВИХ РЯДІВ, PYTHON.

Об’єктом дослідження є сучасні методи та моделі штучних нейронних мереж, які забезпечують ефективне вирішення задачі прогнозування.

Метою роботи є дослідження методів прогнозування на основі нейронних мереж (НМ), які будуть використовуватись в системах прогнозування вихідних показників об'єктів.

В результаті роботи здійснена програмна реалізація системи для підвищення точності і прогнозування вихідних показників на основі нейронних мереж.

KERAS, DATA MINING, FINANCIAL TIME SERIES FORECASTING, PYTHON.

The object of the research is the modern methods and models of artificial neural networks that provide an effective solution to the problem of prediction.

The purpose of the work is to research a forecasting methods based on neural networks that will be used in forecasting systems of output indicators.

As a result of the work the program implementation of the system to improve accuracy and prediction of output indicators based on neural networks.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

НМ – нейронна мережа;

РБФ – радіально-базисна функція;

ШНМ – штучна нейронна мережа;

ШІ − штучний інтелект;

Data Mining – інтелектуальний аналіз даних;

MLP – багатошаровий перцептрон;

Deep Neural Network – глибинні нейронні мережі.

ВСТУП

В епоху інформаційних технологій динамічне зростання обсягу даних викликало стрімкий розвиток наукового напрямку штучний інтелект. Ведуться роботи зі створення машини, яка зможе змалювати нейронну структуру, подібну мозку, і мати можливість формувати уявлення про знання, зберігати і обробляти отримані дані, а також самонавчатися. Але виникає проблема в тому, що немає повного уявлення механізму біологічного інтелекту, тому здійснити перехід від природнього інтелекту до штучного не вдалося. Проте під інтелектом в межах цієї науки розуміється тільки обчислювальна складова, котра забезпечує автоматизацію і швидкість обробки даних в релевантну інформацію.

Існує багато напрямків науки штучного інтелекту і найбільш обширними є машинне навчання та інтелектуальний аналіз даних (Data Mining). В рамках цих наук вирішується доволі багато завдань, а саме таких як регресія, класифікація, кластеризація, прогнозування та інші. Найбільш ефективні методи реалізовані на основі штучних нейронних мереж.

Задача прогнозування, мабуть, може вважатися однією з найбільш складних задач інтелектуального аналізу даних, вона вимагає ретельного дослідження вихідного набору даних і методів, придатних для аналізу.

Задачі прогнозування вирішуються в найрізноманітніших галузях людської діяльності, таких як наука, економіка, виробництво і безліч інших сфер. Прогнозування є важливим елементом організації управління як окремими господарюючими суб'єктами, так і економікою в цілому.

Таким чином, актуальність атестаційної роботи зумовлена наступними факторами:

– прогнозування вихідних показників об’єктів дослідження, особливо фінансових часових рядів, є необхідним елементом будь-якої інвестиційної діяльності;

– моделі штучних нейронних мереж успішно використовуються у вирішенні складних задач, які не мають аналітичного рішення;

– безліч існуючих нейромережевих моделей і розрізненні дані щодо їх ефективності не дозволяють здійснити однозначний вибір найбільш точних моделей прогнозування.

Метою роботи є визначення ефективності використання нейромережевих методів і моделей з метою підвищення точності прогнозування вихідних показників об'єктів.

Об’єктом дослідження є прогнозування вихідних показників об`єктів.

Предметом дослідження даної роботи є дослідження ефективності сучасних нейромережевих методів і моделей для підвищення точності прогнозування вихідних показників об'єктів.

Методи дослідження. Застосовувались емпіричні методи програмної інженерії, а також загальнологічні методи наукового пізнання. В першу чергу це стосується аналізу предметної галузі, з узагальненням ефективності поточних практичних та теоретичних підходів до підвищення точності прогнозування вихідних показників об'єктів.

Наукова новизна полягає у визначені критеріїв та розробці методики оцінювання ефективності нейромережевих моделей прогнозування.

Практичне значення. Практична цінність роботи полягає у порівнянні результатів аналізу показників ефективності нейромережевих моделей MLP, GoogLeNet, Inception-V3 та ResNet.

Запропоновані в роботі моделі прогнозування фінансових часових рядів можуть бути використані для прогнозування часових рядів, а саме фінансових рядів, які мають випадкову природу та непередбачуваність. Нейромережеві моделі прогнозування довели свою ефективність в задачах прогнозування.

Публікації. По результатам атестаційної роботи опублікована стаття «Ефективність використання нейромережевих моделей для прогнозування руху цін, акцій компаній на ринку» в міжнародному електронному науковому журналі «Наука онлайн».

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Актуальність роботи

На сьогоднішній день технології штучних нейронних мереж досить часто і досить успішно використовуються у вирішенні складних задач, які, як правило, не мають аналітичного рішення. На сьогодні нейронні мережі стають все поширенішими для вирішення різних задач обробки сигналів, оптимізації, оптимального і адаптивного управління, розпізнавання образів, ідентифікації, прогнозування в реальному часі і т.п. Створено реальні системи обробки зображень та комп'ютерного зору, управління аерокосмічними об'єктами, технічної та медичної діагностики, в економіці і фінансах (планування, управління, аналіз ринків, прогнозування курсів, технічний аналіз, пошук інформації, ідентифікація кредитних карт), у військовій справі, управлінні рухом, в енергетиці (оцінка станів, виявлення розладнань, прогнозування навантажень), в криміналістиці, аналізі сигналів різної природи та ін., причому цей перелік розширюється практично щодня.

Задача прогнозування фінансових часових рядів була і залишається актуальною, оскільки передбачення є необхідним елементом будь-якої інвестиційної діяльності, адже сама ідея інвестування − вкладення грошей з метою отримання доходу в майбутньому − ґрунтується на ідеї прогнозування майбутнього. Останнім часом, коли стали доступні потужні засоби збору та обробки інформації, задача прогнозування фінансових часових рядів також стала однією з найпопулярніших задач для практичного застосування різних методів інтелектуального аналізу даних (Data Mining). Широке застосування Data Mining методів в даній області обумовлено наявністю в більшості часових рядів складних закономірностей, які не виявляються лінійними методами.

На сьогоднішній день існує безліч моделей прогнозування часових рядів: регресивні і авторегресійні моделі, нейромережеві моделі, моделі експоненціального згладжування, моделі на базі ланцюгів Маркова, класифікаційні

моделі та ін. Найбільш популярними і широко використовуваними є класи авторегресійних і нейромережевих моделей.

Класичні штучні нейронні мережі багаторазово показали свою придатність в області фінансового аналізу в силу своєї високоустойчивої природи, здібностей до універсальної апроксимації.

Виходячи з цього слід зазначити, що використання моделей та методів нейронних мереж для прогнозування часових рядів, а саме фінансових рядів, які мають випадкову природу та непередбачувані є актуальною та основною метою даної роботи.

1.2 Виявлення проблем та актуалізація рішень

Концепція обробки великої кількості даних, їх аналітика, обробка, пошук по ним була протягом багатьох років, більшість організацій в даний час розуміють, що якщо вони захоплюють всі дані, потоки в свій бізнес, вони можуть застосувати аналітику і отримати значну віддачу від нього. Але навіть в 1950-і роки, перш ніж хтось винайшов такий термін як «великі дані», підприємства мало використовували аналітику здебільшого це були таблиці, обстеженні вручну, щоб розкрити ідеї і тенденції [1].

Нові переваги, які приносить аналіз великих даних – це швидкість і ефективність. У той час як бізнес зібрав інформацію, запустив аналітику і розкопав інформацію, яка може бути використана для прийняття майбутніх рішень, бізнес може ідентифікувати інформацію для негайного вирішення. Здатність працювати швидше і залишатися гнучкими надає організаціям конкурентну перевагу, чого в них не було раніше. Високоефективна аналітика даних дозволяє робити речі, про які не думали раніше, оскільки обсяги даних були просто занадто великими. Наприклад, можливість отримати своєчасні ідеї для прийняття рішень при швидкоплинних можливостях та отримати точні відповіді на важко вирішувані

проблеми і відкрити нові можливості для зростання ефективності, також це дає можливість для використання ресурсів та часу більш ефективно. Поява величезної кількості мобільних телефонів, планшетних ПК, фліп-комп'ютерів, а також зростання хмарних обчислень, поряд з іншими джерелами даних, такими як датчики і інтернет , значно прискорили великі об'єми даних. Тільки за останні два роки, більше даних було створено, ніж у всій історії. Обробка, аналіз та пошук великого масиву даних, як правило, відносяться до будь-якої великої кількості необроблених даних, які можуть бути зібрані, збережені, і аналізовані за допомогою різних засобів, що розкривають закономірність або тенденцію, що стосуються поведінки − особливо у споживачів − які можуть бути використані, щоб максимізувати потенціал бізнесу. Також слід більше зосередити увагу на ефективному використанні великих обсягів даних за рахунок використання візуалізації даних, тому як візуалізація сприяє кращому сприйняттю даних.

Візуалізація даних є ключем для аналітиків для розуміння трендів ринку та аналізу даних. Збільшуючи використання візуалізацій, бізнес знайшов цінність, яку він шукав. Створення більше інфографіки і візуальних елементів, дозволило їм швидше отримувати більше інформації. Візуалізація даних дозволяє збільшити функціональність відділу аналітиків, щоб вони могли розуміти краще тренди їх систем, візуалізація дозволяє створити зв'язок між точками даних, які, здавалося б, не мають будь-яких посилань на них, та дає змогу знайти хороші і погані дані, і при цьому аналітики можуть збільшити свою продуктивність, а також вартість інформації, яку вони зібрали.

1.3 Штучна нейронна мережа та її складові

Наш мозок являє собою складну біологічну нейронну мережу, яка приймає інформацію від органів почуттів і якимось чином її обробляє (впізнавання осіб,

виникнення відчутів і т.д.). Наш мозок, як і будь-яка біологічна нейронна мережа, складається із сукупності нейронів.

Біологічний нейрон − надзвичайно складна система. Багато в чому це пояснюється тим, що нейрон, крім обробки сигналу (основне його призначення), змушений ще виконувати купу інших функцій, що підтримують його життя. Більш того, сам механізм передачі сигналу від нейрона до нейрона теж дуже складний з біологічної та хімічної точки зору.

Штучний нейрон це структура, яка приймає сигнал, перетворює його (приблизно так, як це роблять справжні нейрони), і передає іншим нейронам (які роблять те ж саме).

Біологічні нейронні мережі являють собою сукупність біологічних нейронів. Однак в таких мережах теж багато непотрібних для обробки сигналу аспектів. Плюс до всього нейронів в біологічної нейромережі дуже багато.

Штучні нейронні мережі являють собою систему з'єднаних між собою простих обробників (штучних нейронів), які взаємодіють. Такі обробники зазвичай є доволі простими. Кожен обробник подібної мережі має справу лише з сигналами, які він періодично отримує, і сигналами, які він періодично надсилає іншим обробникам, такі локально прості обробники разом здатні виконувати доволі складні завдання[2]. Синапсами називають зв'язки, за якими вихідні сигнали одних нейронів надходять на вхід інших. Кожен зв'язок характеризується своєю вагою. Зв'язки з позитивним вагою називаються збудливими, а з негативним − гальмуючими. Вихід нейрона називається аксон. У ШНМ штучний нейрон − це деяка нелінійна функція, аргументом якої є лінійна комбінація всіх вхідних сигналів. Така функція називається функцією активації. Результат функції активації надсилається на вихід нейрона.

Функція активації нейрона характеризує залежність сигналу на виході нейрона від суми сигналів на його входах. Зазвичай функція є монотонно зростаючою і знаходиться в області значень [-1,1] (гіперболічний тангенс) і [0,1] (сигмоіда). Для деяких алгоритмів навчання необхідно, щоб функція активації була

безперервно диференціюючею по всій числовій осі. Штучний нейрон характеризується своєї активаційною функцією.

Основні функції активації:

– порогова функція активації (функція Хевісайда). Не використовується в алгоритмах зворотного поширення помилки

$$f(x) = \begin{cases} 1 \; if \; x \geq -w_0 x_0 \\ 0 \; else \end{cases}, \tag{1.1}$$

де x − зважена сума сигналів на входах нейрона,

$-w_0x_0$ − зсув функції активації відносно горизонтальної осі.

– сигмоідальна функція активації

$$\sigma(x) = \frac{1}{1+e^{-x}} \; , \tag{1.2}$$

– гіперболічний тангенс

$$\tanh(x) = \frac{e^{Ax} - e^{-Ax}}{e^{Ax} + e^{-Ax}}. \tag{1.3}$$

В основі перцептрона лежить математична модель сприйняття інформації мозком. Різні дослідники по-різному його визначають. У найзагальнішому своєму вигляді перцептрон представляє систему з елементів трьох різних типів: сенсорів, асоціативних елементів і реагуючих елементів. Архітектура нейронної мережі перцептрона наведена на рисунку 1.1.

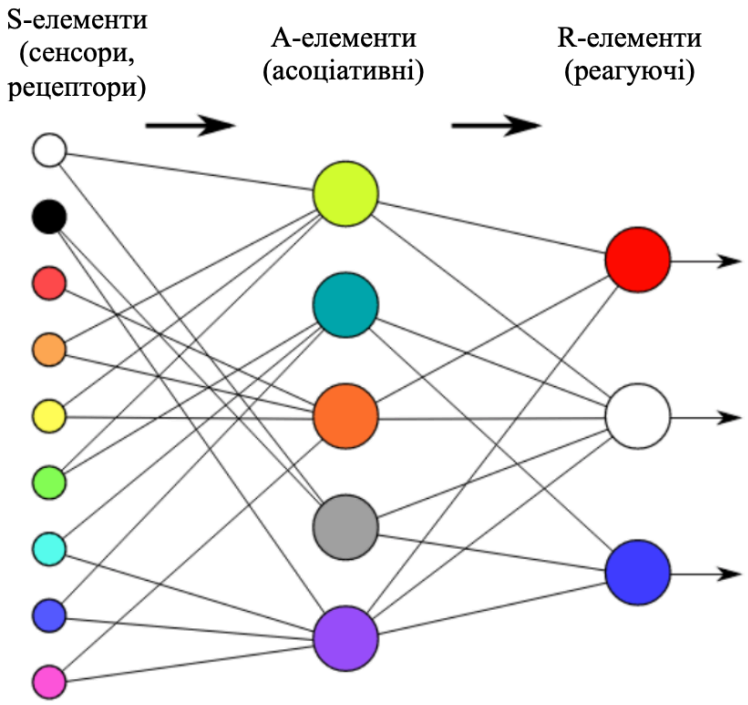

Рисунок 1.1 – Архітектура нейронної мережі перцептрон

Першими в роботу включаються *S*-елементи. Вони можуть перебувати в стані покою (сигнал дорівнює 0) або в стані збудження (сигнал дорівнює 1).

Далі сигнали від *S*-елементів передаються *A*-елементам по так званим *S-A* зв'язкам. Ці зв'язки можуть мати ваги, рівні тільки -1, 0 або 1.

Потім сигнали від сенсорних елементів, які пройшли по *S-A* зв'язкам потрапляють в *A*-елементи, які ще називають асоціативними елементами. Варто зауважити, що одному *A*-елементу може відповідати кілька *S*-елементів. Якщо сигнали, що надійшли на *A*-елемент, в сукупності перевищують деякий його поріг θ, то цей *A*-елемент збуджується і видає сигнал, рівний 1. В іншому випадку (сигнал від *S*-елементів не перевищив порогу *A*-елемента) генерується нульовий сигнал.

Перцептрон (Perceptron) − найпростіший вид нейронних мереж. В основі лежить математична модель сприйняття інформації мозком, що складається з сенсорів, асоціативних і реагують елементів.

Одношаровий перцептрон (перцептрон Розенблатта) − одношарова нейронна мережа, всі нейрони якої мають жорстку порогову функцію активації.

Одношаровий перцептрон має простий алгоритм навчання і здатний вирішувати лише найпростіші задачі. Ця модель викликала до себе великий інтерес на початку 1960-х років і стала поштовхом до розвитку штучних нейронних мереж.

Класичний приклад такої нейронної мережі − одношаровий перцептрон наведено на рисунку 1.2.

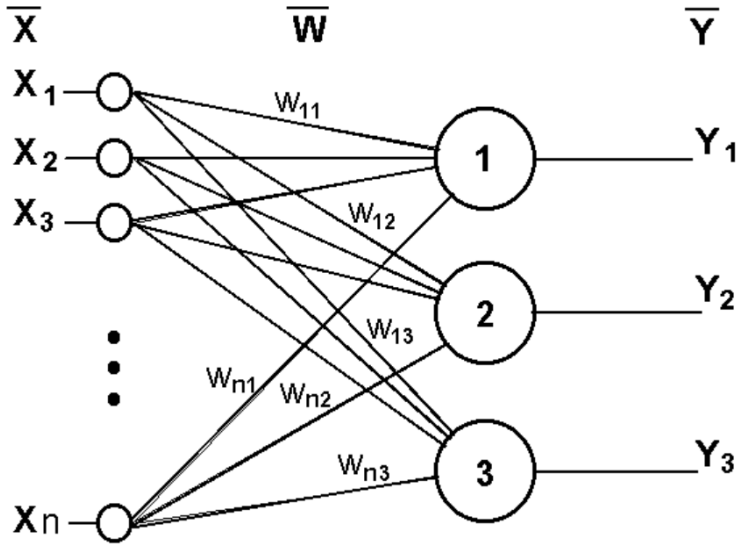

Рисунок 1.2 – Архітектура одношарового перцептрона

Мережа, зображена на малюнку, має *n*-входів, на які надходять сигнали, що йдуть по синапсах на 3 нейрона. Ці три нейрона утворюють єдиний шар даної мережі і мають три вихідних сигнала.

Багатошаровий перцептрон (MLP) − нейронна мережа прямого поширення сигналу (без зворотних зв'язків), в якій вхідний сигнал перетворює в вихідний, проходячи послідовно через кілька шарів. Перший з таких шарів називають вхідним, останній − вихідним. Ці шари містять так звані вироджені нейрони і іноді в кількості шарів не враховуються. Крім вхідного і вихідного шарів, в багатошаровому перцептроні є один або кілька проміжних шарів, які називають прихованими. У цій моделі перцептрона повинен бути хоча б один прихований шар. Присутність декількох таких шарів виправдана лише в разі використання нелінійних функцій активації. Приклад двошарового перцептрона наведено на рисунку 1.3.

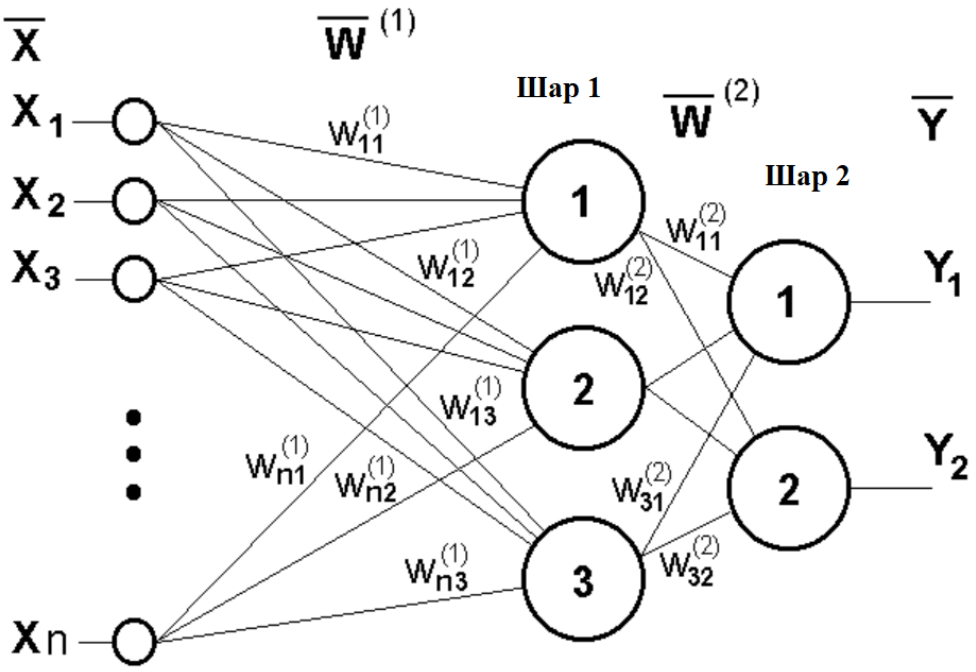

Рисунок 1.3 – Архітектура двошарового перцептрона

Мережа, зображена на малюнку, має *n* входів. На них надходять сигнали, що йдуть далі по синапсах на 3 нейрона, які утворюють перший шар. Вихідні сигнали першого шару передаються двом нейронам другого шару. Останні, в свою чергу, видають два вихідних сигнали.

1.4 Навчання штучної нейронної мережі

Основною властивістю біологічного мозку є його здатність до навчання, а оскільки штучна нейронна мережа є моделлю мозку, поняття «навчання» є також ключовим в теорії ШНМ. Математичними проблемами, пов'язаними з процесами навчання, займається напрямок в загальній теорії штучних нейронних мереж, що отримав назву «нейроматематика» [3-7].

З позиції нейроматематики, процес навчання розглядається як адаптація параметрів, а можливо і архітектури мережі для вирішення поставленого завдання шляхом оптимізації прийнятого критерію якості. Таке формулювання є

загальноприйнятим і неявно передбачає, що в основі нейроматематики лежать методи оптимізації та ідентифікації.

Зазвичай вважається, що процес навчання має перманентний характер і з плином часу мережа покращує свої характеристики, поступово «наближаючись» до оптимального рішення поставленого завдання. Тобто в процесі навчання нейронна мережа здатна виявляти складні залежності між вхідними даними й вихідними, а також здійснювати узагальнення. Це означає, що в разі успішного навчання мережа зможе повернути правильний результат на підставі даних, які були відсутні в навчальній вибірці.

З погляду математики, навчання нейронних мереж – це багатопараметрична задача нелінійної оптимізації. Більшість методів навчання можна розділити на два класи: навчання з учителем (із заохоченням) та навчання без учителя (без заохочення або самонавчання). Методи навчання з учителем застосовують у випадках, коли відома бажана реакція системи в кожну мить часу, себто відомий навчальний сигнал, який впливає на налаштування параметрів системи, що навчається. Рівень «навченості» системи формально визначають за значенням цільової функції, тобто за тим станом, якого має набути система в результаті навчання.

Тип і характер навчання визначаються насамперед обсягом апріорної і поточної інформації про середовище, в яку «занурена» мережа, а також критерієм навчання (цільовою функцією), що характеризує ступінь відповідності нейромережі розв'язуваної нею задачі. Інформація про зовнішнє середовище задана, як правило, у вигляді навчальної вибірки образів або прикладів, обробляючи яку, мережа витягує відомості, необхідні для отримання шуканого рішення. Саме характер і обсяг цієї інформації визначають як тип навчання, так і конкретний алгоритм.

Більшість задач поділяються на наступні:

Проблеми малої/середньої складності, які точно відомо як вирішувати:

- вирішити прості рівняння;
- вивести на екрані вікно програми;

- роздрукувати документ на принтері.

Такі завдання вирішуються за допомогою звичайних комп'ютерних програм. Нічого складного і незвичайного. Ніяких проблем.

Проблеми малої/середньої складності, які частково відомо, як вирішувати:

- найпростіше прогнозування;
- розрахунок похибок;
- наближене рішення рівнянь.

Такі завдання можна вирішувати за допомогою спеціальних статистичних методів.

Проблеми високої складності, які незрозуміло як вирішувати:

- розпізнавання образів;
- розпізнавання мови;
- складні прогнози.

Нейронні мережі застосовують для вирішення задач, алгоритм вирішення яких невідомий. Ця здатність нейромереж і зробила їх такими популярними. Нейронні мережі можна навчити грати в ігри, впізнавати голос, прогнозувати зростання/падіння цін. Крім можливості вирішувати новий клас задач нейромережі мають ряд значних переваг. Всі плюси нейронних мереж є наслідками плюсів біологічних нейронних мереж тому, що сама модель обробки інформації не змінювалася. Звідси дуже просто пояснювати переваги ШНМ, просто виводячи їх з властивостей біологічних нейромереж.

Нейронні мережі здатні коректно функціонувати, навіть якщо на вході дані зашумлені. Нейронні мережі можуть підлаштовуватися під навколишнє оточення, що змінюється [8].

Комп'ютер виконує команди послідовно. Однак в голові людини кожен нейрон є маленьким процесором (який приймає сигнал, перетворює його, і подає на вихід). І таких процесорів в голові мільярди. Отримуємо гігантську мережу розподілених обчислень. Сигнал обробляється нейронами одночасно. Ця властивість потенційно проявляється і в штучних нейронних мережах. Якщо у вас

є багатоядерний комп'ютер, то ця властивість буде виконуватися. Для одноядерних комп'ютерів ніякої різниці помітно не буде.

У нейронних мереж є ряд недоліків, які теж можна вивести з біологічних нейронних мереж. Варто зауважити, що нейронні мережі, незважаючи на широкий спектр завдань, які вони можуть вирішувати, все ж залишаються лише корисним додатковим функціоналом. На першому місці завжди стоять комп'ютерні програми. Новина полягає в тому, що інтегруючи звичайні програмні алгоритми та нейронні мережі можна майже повністю позбавитися від всіх потенційних недоліків.

Нейрони штучної нейромережі в загальному випадку не залежать один від одного. Вони просто отримують сигнал, перетворюють його і віддають далі. Вони не дивляться один на одного і, в залежності від нейрона-сусіда, змінюють свої синапси. Звідси випливає, що нейронна мережа може вирішувати завдання тільки в один захід.

У реальній біологічній нейронній мережі від входів мережі до виходів передається електричний сигнал. В процесі проходу по нейронній мережі він може змінюватися. Електричний сигнал завжди буде електричним сигналом. Змінюється величина цього електричного сигналу (сильніше/слабше). А будь-яку величину завжди можна виразити числом (більше/менше).

У моделі штучної нейронної мережі абсолютно не потрібно реалізовувати поведінку електричного сигналу тому, що від його реалізації все одно нічого залежати не буде. На входи мережі подаються числа, що символізують величини електричного сигналу, якби він був. Ці числа просуваються по мережі і якимось чином змінюються. На виході мережі отримується якесь результуюче число, яке є відгуком мережі.

Синапс це зв'язок між двома нейронами. У синапсів є один параметр − вага. Завдяки йому, вхідна інформація змінюється, коли передається від одного нейрона до іншого. Припустимо, є три нейрона, які передають інформацію з наступних підстав. Тоді у нас є три ваги, відповідні кожному з цих нейронів. У того нейрона, у якого вага буде більше, та інформація і буде домінуючою в наступному нейроні

(приклад − змішування кольорів). Насправді, сукупність ваг нейронної мережі або матриця ваг − це своєрідний мозок всієї системи. Саме завдяки цим вагам, вхідна інформація обробляється і перетворюється в результат. Синапси можуть посилювати чи послаблювати електричний сигнал, який проходить по ним. Зв'язок характеризується певним числом, званим вагою зв'язку. Сигнал, що пройшов через даний зв'язок, множиться на вагу відповідного зв'язку. Це ключовий момент в концепції штучних нейронних мереж. Кожному зв'язку відповідає певна кількість w_i вага зв'язку (див. рис. 1.4). І коли сигнал проходить по цьому зв'язку, його величина множиться на вагу зв'язку з ним.

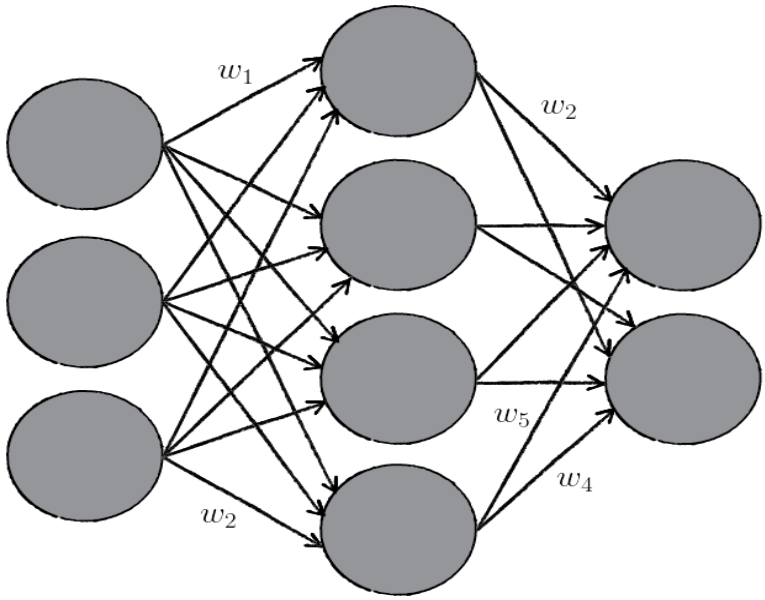

Рисунок 1.4 – Вагові зв'язки між нейронами

У кожного нейрона повинні бути входи, через які він приймає сигнал. Ми вже вводили поняття ваг, на які множаться сигнали, що проходять по зв'язку. Сигнали, які надійшли на входи, множаться на свої ваги. Сигнал першого входу x_1 множиться на відповідний цьому запису ваги w_1. У підсумку отримуємо x_1w_1. І так до n - го входу. У підсумку на останній вхід отримуємо x_nw_n. Після цього всі значення передаються в суматор, який просто підсумовує всі вхідні сигнали, помножені на відповідні ваги. Роль суматора очевидна − він агрегує всі вхідні сигнали (яких може бути багато) в якесь одне число − зважену суму, яка характеризує сигнал в цілому,

який надійшов на нейрон. Ще зважену суму можна уявити як ступінь загального збудження нейрона.

У кожного нейрона повинні бути входи, через які він приймає сигнал. Ми вже вводили поняття ваг, на які множаться сигнали, що проходять по зв'язку. Сигнали, які надійшли на входи, множаться на свої ваги. Сигнал першого входу x_1 множиться на відповідний цьому запису ваги w_1. У підсумку отримуємо x_1w_1. І так до n - го входу. У підсумку на останній вхід отримуємо x_nw_n. Після цього всі значення передаються в суматор, який просто підсумовує всі вхідні сигнали, помножені на відповідні ваги. Роль суматора очевидна − він агрегує всі вхідні сигнали (яких може бути багато) в якесь одне число − зважену суму, яка характеризує сигнал в цілому, який надійшов на нейрон. Ще зважену суму можна уявити як ступінь загального збудження нейрона.

Після цього нейрон обробляє інформацію і формує вихідний сигнал. Саме для цих цілей і використовують функцію активації. Вона перетворює зважену суму в якесь число, яке і є виходом нейрона (вихід нейрона позначимо *out*). Для різних типів штучних нейронів використовують найрізноманітніші функції активації. У загальному випадку їх позначають символом $\varphi(net)$. Вказівка зваженого сигналу в дужках означає, що функція активації приймає зважену суму як параметр.

У найпростішому виді функції активації вихід нейрона може дорівнювати лише 0 або 1. Якщо зважена сума більше певного порогу b, то вихід нейрона дорівнює 1. Якщо нижче, то 0. Загалом, нейрон дивиться на зважену суму i якщо вона виходить більше його порога, то нейрон видає вихідний сигнал, який дорівнює 1. Існує ціле сімейство сигмоідальних функцій, деякі з яких застосовують як функції активації в штучних нейронах. Всі ці функції мають деякі дуже корисними властивостями, заради яких їх і застосовують в нейронних мережах.

Вибір даних для навчання мережі та їх обробка є найскладнішим етапом вирішення задачі. Набір даних для навчання повинен задовольняти декільком критеріям:

− репрезентативність − дані повинні ілюструвати справжній стан речей в предметної області;

− несуперечливість − суперечливі дані в навчальній вибірці призведуть до поганої якості навчання мережі.

Вихідні дані перетворюються до вигляду, в якому їх можна подати на входи мережі. Кожен запис у файлі даних називається навчальною парою або навчальним вектором. Навчальний вектор містить по одному значенню на кожен вхід мережі і, в залежності від типу навчання (з учителем або без), по одному значенню для кожного виходу мережі. Навчання мережі на «сирому» наборі, як правило, не дає якісних результатів. Існує ряд способів поліпшити «сприйняття» мережі.

Нормування виконується, коли на різні входи подаються дані різної розмірності. Наприклад, на перший вхід мережі подаються величини зі значеннями від нуля до одиниці, а на другий − від ста до тисячі. При відсутності нормування значення на другому вході будуть завжди надавати істотно більший вплив на вихід мережі, ніж значення на першому вході. При нормуванні розмірності всіх вхідних і вихідних даних зводяться воєдино;

Квантування виконується над безперервними величинами, для яких виділяється кінцевий набір дискретних значень. Наприклад, квантування використовують для завдання частот звукових сигналів при розпізнаванні мови;

Фільтрація виконується для «зашумлених» даних. Крім того, велику роль відіграє саме уявлення як вхідних, так і вихідних даних. Припустимо, мережа навчається розпізнаванню букв на зображеннях і має один числовий вихід − номер букви в алфавіті. У цьому випадку мережа отримає неправильне уявлення про те, що букви з номерами 1 і 2 більш схожі, ніж букви з номерами 1 і 3, що, загалом, не так. Для того, щоб уникнути такої ситуації, використовують топологію мережі з великим числом виходів, коли кожен вихід має свій сенс. Чим більше виходів в мережі, тим більша відстань між класами і тим складніше їх сплутати.

Вибирати тип мережі слід, виходячи з постановки задачі і наявних даних для навчання. Для навчання з учителем потрібна наявність для кожного елемента вибірки «експертної» оцінки. Іноді отримання такої оцінки для великого масиву даних просто неможливе. У цих випадках природним вибором є мережа, яка навчається без учителя (наприклад, самоорганізована карта Кохонена або нейронна

мережа Хопфілда). При вирішенні інших завдань (таких, як прогнозування часових рядів) експертна оцінка вже міститься у вихідних даних і може бути виділена при їх обробці.

Після вибору загальної структури потрібно експериментально підібрати параметри мережі. Для мереж, подібних перцептрона, це буде число шарів, число блоків в прихованих шарах (для мереж Ворда), наявність або відсутність обхідних з'єднань, передавальні функції нейронів. При виборі кількості шарів і нейронів у них слід виходити з того, що здатності мережі до узагальнення тим вище, чим більше сумарне число зв'язків між нейронами. З іншого боку, число зв'язків обмежене зверху кількістю записів в навчальних даних.

Після вибору конкретної топології необхідно вибрати параметри навчання нейронної мережі. Цей етап особливо важливий для мереж, що навчаються з учителем. Від правильного вибору параметрів залежить не тільки те, наскільки швидко відповіді мережі будуть сходитися до правильних відповідей. Наприклад, вибір низькій швидкості навчання збільшить час сходження, проте іноді дозволяє уникнути паралічу мережі. Збільшення часу навчання може привести як до збільшення, так і до зменшення часу збіжності, в залежності від форми поверхні помилки. Виходячи з такого суперечливого впливу параметрів, можна зробити висновок, що їх значення потрібно вибирати експериментально, керуючись при цьому критерієм завершення навчання (наприклад, мінімізація помилки або обмеження за часом навчання).

В процесі навчання мережа в певному порядку переглядає навчальну вибірку. Порядок перегляду може бути послідовним, випадковим і т. п. Деякі мережі, які навчаються без учителя (наприклад, мережі Хопфілда), переглядають вибірку тільки один раз. Інші (наприклад, мережі Кохонена), а також мережі, які навчаються з учителем, переглядають вибірку безліч разів, при цьому один повний прохід по вибірці називається епохою навчання. При навчанні з учителем набір вихідних даних ділять на дві частини − власне навчальну вибірку і тестові дані; принцип поділу може бути довільним. Навчальні дані подаються мережі для навчання, а перевірочні використовуються для розрахунку помилки мережі

(перевірочні дані ніколи для навчання мережі не застосовуються). Таким чином, якщо на перевірочних даних помилка зменшується, то мережа дійсно виконує узагальнення. Якщо помилка на навчальних даних продовжує зменшуватися, а помилка на тестових даних збільшується, значить, мережа перестала виконувати узагальнення і просто «запам'ятовує» навчальні дані. Це явище називається перенавчанням мережі. У таких випадках навчання зазвичай припиняють. У процесі навчання можуть проявитися інші проблеми, такі як параліч або потрапляння мережі в локальний мінімум поверхні помилок. Неможливо заздалегідь передбачити прояв тієї чи іншої проблеми, так само як і дати однозначні рекомендації щодо їх вирішення.

Все вище сказане відноситься тільки до ітераційних алгоритмів пошуку нейромережевих рішень. Для них дійсно не можна нічого гарантувати і не можна повністю автоматизувати навчання нейронних мереж. Однак, поряд з ітераційним алгоритмами навчання, існують неітераційні алгоритми, що володіють дуже високою стійкістю і дозволяють повністю автоматизувати процес навчання.

Навіть в разі успішного, на перший погляд, навчання мережа не завжди навчається саме тому, чого від неї хотів творець. Відомий випадок, коли мережа навчалася розпізнаванню зображень танків по фотографіях, проте пізніше з'ясувалося, що всі танки були сфотографовані на одному і тому ж фоні. У результаті мережа «навчилася» розпізнавати цей тип ландшафту, замість того, щоб «навчитися» розпізнавати танки. Таким чином, мережа «розуміє» не те, що від неї вимагалося, а те, що найпростіше узагальнити.

Тестування якості навчання нейромережі необхідно проводити на прикладах, які не брали участі в її навчанні. При цьому число тестових прикладів має бути тим більше, чим вище якість навчання. Якщо помилки нейронної мережі близькі до однієї мільярдної, то і для підтвердження цієї ймовірності потрібен мільярд тестових прикладів. Виходить, що тестування добре навчених нейронних мереж стає дуже важким завданням.

1.5 Постановка задачі

На основі виконаного аналізу предметної галузі основною задачею стає дослідження ефективності використання нейромережевих методів і моделей для підвищення точності прогнозування вихідних показників об'єктів. Розробка та тестування програмного забезпечення.

Найважливіша особливість нейронної мережі, яка свідчить про її широкі можливості і величезний потенціал, полягає в паралельній обробці інформації усіма ланками, що дозволяє значно прискорити процес обробки інформації. Інша, не менш важлива властивість − здатність до навчання і узагальнення накопичених знань. Нейронна мережа має риси штучного інтелекту. В даний час нейронні мережі використовуються для вирішення цілої низки задач, однією з яких є задача прогнозування.

На основі проведеного аналізу предметної області можна сформулювати завдання, які необхідно вирішити:

− провести аналіз існуючих нейромережевих методів і моделей;

− проаналізувати існуючі проблеми та визначити шляхи їх вирішення;

− визначити критерії та розробити методику оцінювання ефективності нейромережевих моделей прогнозування;

− розробити нейромережеву модель для прогнозування курсу акцій компаній;

− розробити програму для визначення показників ефективності моделей прогнозування вихідних показників об’єктів;

− провести порівняльний аналіз показників ефективності нейромережевих моделей прогнозування.

2 АНАЛІЗ НЕЙРОМЕРЕЖЕВИХ МЕТОДІВ І МОДЕЛЕЙ

2.1 Основні принципи навчання штучних нейронних мереж

Поняття «навчання» є ключовим в теорії ШНМ. Тип і характер навчання визначаються на основі розв'язуваної задачі, властивостей даних, які надходять на вхід мережі із зовнішнього середовища у вигляді навчальної вибірки образів або прикладів. Відомі дві основні парадигми навчання: з учителем і без вчителя.

Навчання з учителем є більш простим і очевидним. Завідомо відома інформація про зовнішнє середовище, яка задана у вигляді послідовності або пакета вхідних векторів x. Також відомий навчальний сигнал d. Реакція ненавченої мережі, відрізняється від «правильної» реакції вчителя, в результаті чого виникає помилка. У процесі навчання необхідно так налаштувати параметри ШНМ, щоб деяка скалярна функція помилки (критерій якості) досягла свого мінімального значення. Навченою вважається мережа, яка в деякому, як правило, статистичному сенсі повторює реакцію вчителя. Оскільки інформація про зовнішнє середовище зазвичай має нестаціонарний характер, процес навчання йде безперервно, для чого використовуються ті чи інші рекурентні процедури. На рисунку 2.1 запропонована схема навчання з учителем.

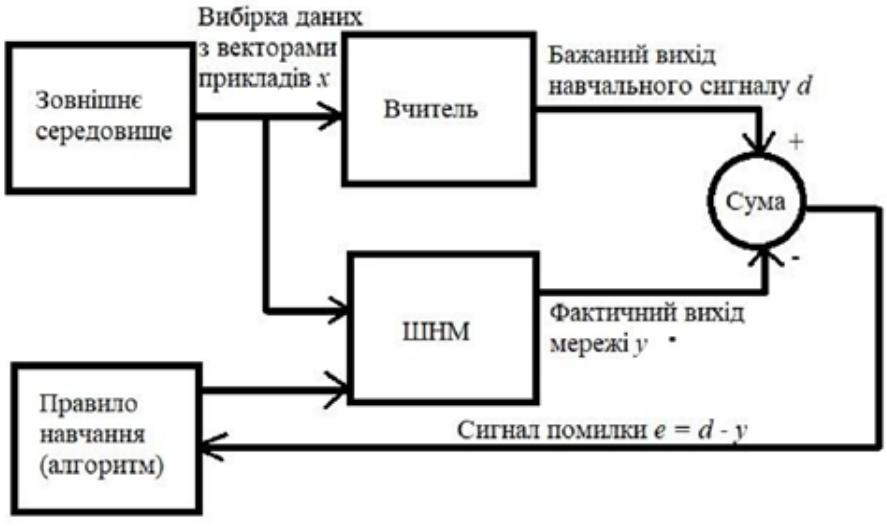

Рисунок 2.1 – Схема навчання з учителем

Парадигмою навчання без учителя або самонавчання, називають навчання, коли правильна реакція на сигнали зовнішнього середовища невідома. Процес самонавчання схематично представлений на рисунку 2.2.

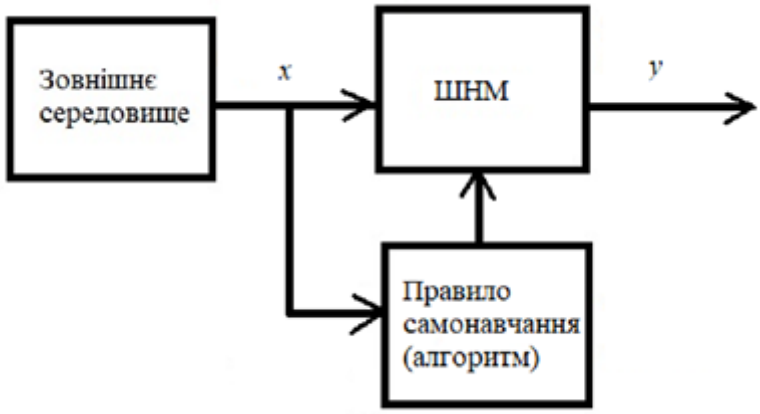

Рисунок 2.2 – Схема навчання без учителя

Мережі, які реалізують парадигму самонавчання, призначені, як правило, для аналізу внутрішньої латентної структури вхідної інформації і вирішують завдання автоматичної класифікації, кластеризації, факторного аналізу, компресії даних. Ці задачі є більш складними, виходячи з математичної постановки, але вирішують безліч реальних проблем.

Своєрідним компромісом між двома цими парадигмами є навчання з підкріпленням, при якому доступна лише непряма інформації про правильну реакції на вхідний сигнал. Нейронна мережа виробляє відображення вхідної інформації у вихідний вектор, проте, оскільки бажаний навчальний сигнал в явному вигляді не заданий, неможливо отримати помилку, на підставі якої відбувається навчання. Передбачається, що є деякі апріорні знання, що дозволяють зв'язати евристичний сигнал підкріплення з бажаним виходом за допомогою деякої функції. Зазвичай ця функція враховує зв'язок вихідних сигналів мережі з подіями у зовнішньому середовищі, для чого в схему навчання вводиться додатковий блок – «критик», що відображає поведінку мережі. Далі обчислюється евристична помилка, на основі якої і реалізується процес навчання.

Досить широке поширення набула також парадигма змішаного навчання, коли частина параметрів мережі налаштовується за допомогою навчання з учителем, а інша частина або архітектура в цілому − за допомогою самонавчання. Цей підхід набув найбільшого поширення при навчанні радіально-базисних ШНМ.

З введеними парадигмами тісно пов'язані правила навчання, що лежать в основі конкретних алгоритмів. У конкурентному навчанні можуть бути реалізовані всі описані парадигми, при цьому його відмінною рисою є процес «змагання» нейронів вихідного шару. Існує два принципи змагання згідно з Кохоненом «переможець отримує все» і «переможець отримує більше». У першому випадку збуджується тільки один вихідний нейрон – «переможець». У другому відбувається виправлення всіх синаптичних ваг і переможець отримує більше, ніж всі інші. Найбільш яскравими прикладами мереж, що використовують це правило, є мережі адаптивного резонансу і самоорганізовані мапи.

2.2 Навчання нейронних мереж, засноване на оптимізації

Безліч реальних задач характеризується тим, що дані надходять в online режимі, їх обробка повинна проводитися в темпі функціонування об'єкта, а сам об'єкт є нестаціонарним. Зрозуміло, що класичний багатошаровий перцептрон, що є універсальним апроксиматором, в даному випадку не може бути використано, а в якості альтернативи йому в ряді випадків можуть бути використані радіально-базисні нейронні мережі (РБНМ) [9], [10], [11]. Ці мережі також є універсальними апроксиматорами, а оскільки їх вихідний сигнал лінійно залежить від параметрів, що налаштовуються, тобто синаптичних ваг, для їх навчання в реальному часі може бути використаний рекурентний метод найменших квадратів або його модифікації, що є, по суті, алгоритмами оптимізації другого порядку, що забезпечують квадратичну збіжність до оптимального рішенням.

Уникнути подібних «провалів» можна, скориставшись, так званим,

одиничним розбиттям вхідного простору, що реалізовуються за допомогою нормалізованих радіально-базисних нейронних мереж (НРБНМ) [11], в яких вихідний сигнал мережі нормується на суму виходів всіх нейронів. Дані мережі зазвичай навчаються за допомогою рекурентних градієнтних алгоритмів, загальним недоліком яких є низька швидкість збіжності і можливість попадання в локальні мінімуми прийнятого критерія навчання.

Таким чином, ці нейромережі і багато інших, які використовують рекурентні процедури навчання і об'єднуються загальною назвою «мережі, основані на оптимізації», можуть виявитися неефективними в задачах адаптивної ідентифікації, прогнозування та управління реального часу, коли інформація на обробку надходить з досить високою частотою. Мережа в цьому випадку просто не встигає навчитися, не кажучи вже про можливість спостереження за змінами параметрів об'єкту. Крім того, у багатьох реальних задачах навчальна вибірка може бути настільки мала, що оцінки, одержувані на основі тих чи інших процедур оптимізації, не є ефективними. Тому слід передбачити можливість використання інших форм навчання, відмінних від тих, що засновані на вирішенні задачі оптимізації.

2.3 Нейронні мережі, що миттєво навчаються

Ефективною альтернативою мережам, заснованим на оптимізації, є так звані нейромоделі, які навчаються миттєво, в їх основі лежить принцип «нейрони в точках даних» [12]. В рамках парадигми навчання штучних нейронних мереж подібні моделі також називають «лінивими» моделями навчання. Важливою характерною рисою миттєвих нейромереж вважається те, що етап формування фактичної моделі триває до того моменту, поки існує потреба в даній моделі. Іншими словами, це триває до тих пір, поки для цього входу необхідний вихідний сигнал, а до цього здійснюється тільки збір і зберігання даних.

Найчастіше навчання класичних моделей відбувається в режимі offline і для оцінки застосовується виключно фіксована модель. Тобто, обробка всіх даних, необхідних для навчання, спочатку здійснюється в пакетному режимі. Подібна навчальна процедура відрізняється обчислювальною складністю і в окремих ситуаціях є практично неможливою. У зв'язку з цим використовують методи стиснення даних. Можна зробити ще один висновок на базі цього класичного підходу: для процедури навчання необхідна велика кількість обчислювальних етапів, в той час як оцінювання здійснюється дуже швидко або моментально.

Більш того, в якості додаткового варіанту може застосовуватися online адаптація, яка служить для налаштування і адаптації моделі за допомогою зміни її характеристик (параметрів) протягом фази оцінки. Необхідність в цьому обумовлена важливістю забезпечення наступних властивостей.

Для нейромоделей, які навчаються миттєво, фаза навчання відбувається одночасно з фазою накопичення даних. З цього випливає, що розрахунок параметрів моделі здійснюється в період оціночної фази.

Нейронним мережам, які навчаються миттєво, притаманні локальні параметри, що описують вихідний сигнал, зокрема, його поточний стан. Отже є можливість вибрати найбільш примітивну структурну моделі, наприклад, лінійну. Необхідно відзначити, що етап оптимізації моделі і вибір даних проходить в індивідуальному порядку щодо кожного вхідного образу. В результаті з'являється шанс модифікувати архітектуру моделі, складність її алгоритму та чинники, які необхідно враховувати для вибору даних, зберігаючи послідовність та враховуючи поточну ситуацію. Можна враховувати характерну якість і кількість даних, відповідні обмеження, стан і положення об'єкта. Очевидно, миттєві моделі спочатку мають адаптивність. Отже, оновлені дані, що надходять на обробку в почерговому порядку, далі зважуються і зберігаються в базі даних, а попередні по закінченню часу забуваються.

Узагальнена регресійна нейронна мережа (УРНМ), яку запропонував Д. Шпехт [13], є найяскравішим представником моделей, які навчають на основі цього принципу. Вона базується на принципах ядерних оцінок Надарая-Ватсона

[14], непараметричних моделей [15] і парзеновських вікон [16], а процес її навчання, в кінцевому рахунку, зводиться до одноразової установки багатовимірних радіально-базисних функцій (РБФ) в точках одиничного центрованого гіперкуба, які задаються в однозначному порядку за допомогою навчальної вибірки. Це говорить про те, що подібні мережі можна цілком віднести до тих самих нейронних мереж, що навчаються миттєво [17], [18], настройка яких проводиться за допомогою єдиного проходу навчального алгоритму. За рахунок збігу в плані архітектури з НРБНМ, навчання УРНМ відбувається в більш швидкому темпі, при одночасному встановленні центрів РБФ в точках, координати яких визначаються за допомогою вхідних сигналів об'єкта відповідно до принципу «нейрони в точках даних». При цьому «висота» РБФ повністю збігається з певними параметрами вхідного сигналу об'єкта. Завдяки оперативній швидкості навчання УРНМ стало можливим їх ефективне застосування в реальних задачах.

Основним «недоліком» УРНМ є необхідність великих обсягів пам'яті і використання складних обчислювальних дій, які потрібні для оцінки. Подібні труднощі можуть мати місце для online режиму оцінювання, до їх числа відносяться пропущені дані, час, необхідний для оцінки відгуку. Однак, в рамках підвищення можливостей в області обробки даних, моделі, що навчаються миттєво, можуть стати більш привабливою альтернативою.

2.4 Конкурентні нейронні мережі

Деякі методи навчання, які використовуються для традиційних архітектур штучних нейронних мереж, вимагають значення оптимальних вихідних сигналів мережі в навчальній вибірці інформації, тобто відносяться до класу навчання з учителем.

Але, іноді мають місце реальні задачі, для цілей яких в навчальній вибірці немає необхідних бажаних значень, а присутній виключно набір спостережень $x(k)$.

Тому, для навчання мережі, слід витягувати важливі дані, які ґрунтуються виключно на даному наборі *x(k)*. У більшості випадків процедури навчання без вчителя для ШНМ базуються на конкуренції між нейронами.

Охарактеризувати вищезгадані задачі найкращим чином допоможуть наступні приклади:

– кластеризація – в разі, коли вхідні дані об'єднуються в кластери, їх необхідно виділити для подальшої обробки, як чіткої, так і нечіткої при істотному перетині кластерів;

– векторне квантування – дана задача має місце, коли вхідний векторний простір потрібно відобразити в дискретному вигляді оптимальним способом, за допомогою його розбиття на неперетинні підпростори;

– зниження розмірності – в ситуаціях, коли навчальні дані знаходяться в меншому за розмірністю підпросторі, в порівнянні з вхідною вибіркою. Для зменшення розмірності необхідно побудувати оптимальне відображення R^n в менший за розмірністю простір з мінімальними втратами даних (найчастіше інформативними ознаками вважаються ознаки з найбільшою дисперсією даних);

– вибір ознак – в порівнянні з завданням зниження розмірності в даному випадку необхідно знайти кілька найбільш важливих ознак без їх перетворення.

Головна відмінність методу кластеризації від векторного квантування полягає в тому, що під час кластеризації визначаються області вхідного простору, які включають «схожі» спостереження [19], а в іншому випадку − основне завдання полягає в розбитті всього входового простору.

Конкурентне навчання. В даному випадку вивчаються процедури самонавчання, спрямовані на вирішення завдань кластеризації інформації в класичному розумінні [19].

Процедура самоорганізації заснована на методах конкурентного навчання, а її робота починається з ініціалізації синаптичних ваг мережі, які обираються або у випадковому порядку, або за допомогою будь-якого широко поширеного методу. Як правило, реалізація процедури самоорганізації включає три важливих етапи: конкуренція, кооперація і синаптична адаптація.

Стандартна архітектура конкурентної мережі (competitive neural network) являє собою повнозв'язний шар з m нейронів, які містять по m рецепторів, що характеризуються n–вимірними векторами синаптичних ваг

$$w_j(k) = \left(w_{j,1}(k), w_{j,2}(k), ..., w_{j,n}(k)\right)^T, j = 1,2,...,m. \quad (2.1)$$

На вхід мережі подається вхідний вектор-спостереження $x(k)$, k = 1,2,…. На етапі конкуренції, під час подачі вхідного вектора, відбувається активація лише одного нейрона, який має назву «нейрон-переможець». У випадку з коректно навченою мережею для всіх векторів $x(k)$, які належать єдиному кластеру, активується один і той же нейрон-переможець $w^*(k)$.

В якості функцій активації нейронів конкурентної мережі в більшості випадків використовуються лінійні функції. Графічно архітектура конкурентної мережі зображена на рисунку 2.3.

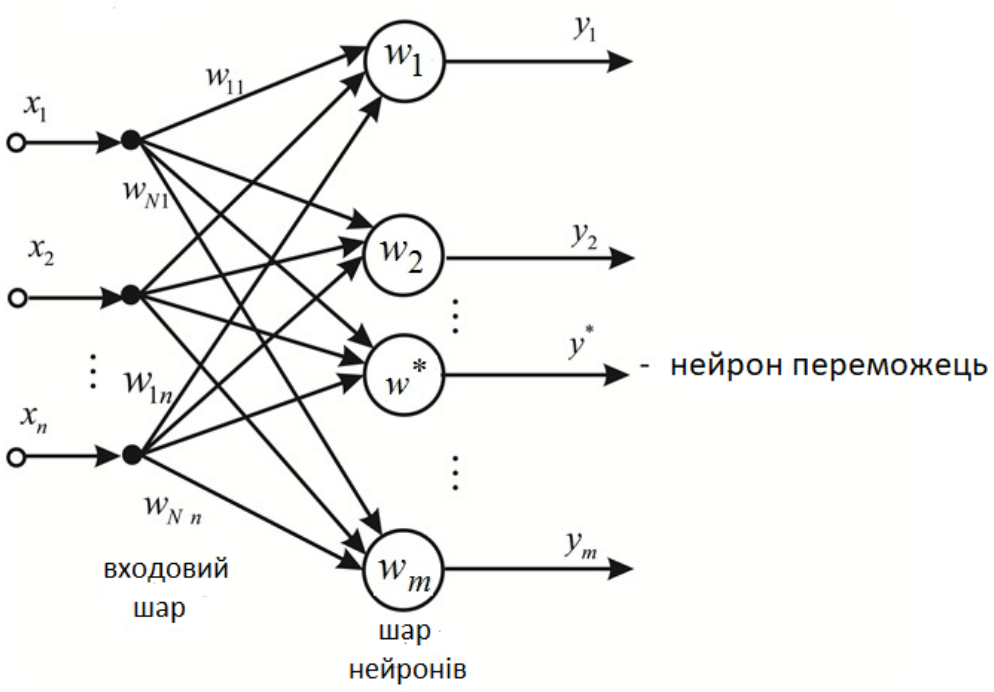

Рисунок 2.3 – Архітектура стандартної конкурентної нейронної мережі

На даний момент відомо два основні методи знаходження нейрона-переможця як «найближчого» до вектору поточного вхідного спостереження *x(k)* відповідно до прийнятої метрики. Розглянемо кожен з них більш детально.

Здійснення вибору нейрона-переможця відповідно до відстані в евклідовому просторі. Скалярний добуток векторів не відображає ступінь їх близькості у випадку з ненормованими даними і векторами синаптичних ваг. Таким чином, щоб вибрати переможця, доцільно застосовувати метрику Евкліда:

$$D_E(a,b) = \|a-b\| = \sqrt{(a-b)^T(a-b)}, \tag{2.2}$$

де *a, b* − координати об'єктів, відповідно до якої

$$w^*(k) = w_O(k), \; k = \arg\max_{j=1,\ldots,m} \|w_j(k) - x(k)\|. \tag{2.3}$$

Налаштування синаптичних ваг. У загальновідомих випадках вибір «ближнього» для вхідного спостереження нейрона-переможця згідно з прийнятою метрикою записується так:

$$D\left(w^*(k), x(k)\right) = \min_i D\left(w_j(k), x(k)\right). \tag{2.4}$$

Наступним етапом після завершення фази вибору переможця є налаштування синаптичних ваг, згідно подальших процедур синаптичної адаптації.

Для нормованого простору ваг:

$$w^*(k+1) = w^*(k) + \eta(k)(x(k) - w^*(k)). \tag{2.5}$$

де $\eta(k)$ − параметр кроку пошуку.

Для ненормованого простору ваг:

$$w^*(k+1) = \frac{w^*(k) + \eta(k)(x(k) - w^*(k))}{\left\| w^*(k) + \eta(k)(x(k) - w^*(k)) \right\|} \quad . \tag{2.6}$$

В даному випадку процедури налаштування ваг (2.5), (2.6) «підтягують» вектор синаптичних ваг нейрона-переможця до вхідного вектору *x(k)* на відстань, що визначається величиною параметру кроку пошуку *η(k)*. У разі нормованого простору налаштування ваги виражається в його повороті в бік *x(k)*.

2.5 Глибинне навчання

У сучасному світі, починаючи з охорони здоров'я і закінчуючи мануфактурним виробництвом, використовуючи глибинне навчання. Компанії звертаються до цих технологій для вирішення складних проблем, таких як розпізнавання мов і об'єктів, машинного перекладу і таке інше.

Звісно, що глибинне навчання ще далеко від досконалості, але воно вже близьке до того, щоб приносити комерційну користь. Наприклад, самокеруючі машини. Відомо, що такі компанії, як Google, Tesla і Uber вже намагаються впровадити автономні автомобілі на вулиці міста.

Нейронна мережа (штучна нейронна мережа) − це спроба відтворення роботи людського мозку на комп'ютері за допомогою шарів нейронів.

Штучний інтелект − здатність машини чи програми знаходити рішення за допомогою обчислень. Під час перших досліджень в області штучного інтелекту вчені намагалися відтворити людський інтелект для вирішення конкретних завдань, наприклад, гри з людиною. Було введено велику кількість правил, яким має слідувати комп'ютер. На основі цих правил комп'ютер приймав рішення відповідно конкретному списку можливих дій [20].

Машинне навчання − це спроба навчити комп'ютери самостійно навчатися на великій кількості даних замість правил. Машинне навчання дозволяє комп'ютерам самостійно навчатися. Це можливо завдяки обчислювальній потужності сучасних комп'ютерів, які можуть легко обробляти великі набори даних.

Взаємодія таких відомих областей як штучний інтелект, машинне навчання та глибинне навчання зображена на рисунку 2.4.

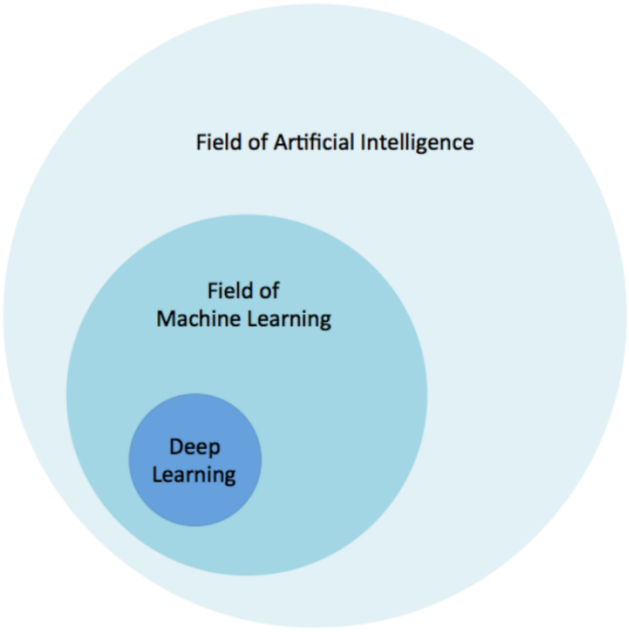

Рисунок 2.4 – Візуалізація взаємодії глибинного навчання з машинним навчанням

Зовнішнє коло − це штучний інтелект в цілому (наприклад, комп'ютери). Трохи далі − машинне навчання, а зовсім в центрі − глибинне навчання і штучні нейромережі. Слово «Глибинне» в словосполученні глибинне навчання означає ступінь складності (глибини) нейромережі, яка часто може бути дуже поверхневою.

Творці першої нейромережі надихалися структурою кори головного мозку. Базовий рівень мережі, перцептрон, є по суті математичним аналогом біологічного нейрона. Перший шар нейромережі називається вхідним. Кожен вузол цього шару отримує на вхід будь-яку інформацію і передає її на наступні вузлі в інших шарах. Найкращий між вузами одного шару немає зв'язків, а останній ланцюжок

виводиться з результатів роботи нейромережі. Архітектура глибинної нейронної мережі зображена на рисунку 2.5.

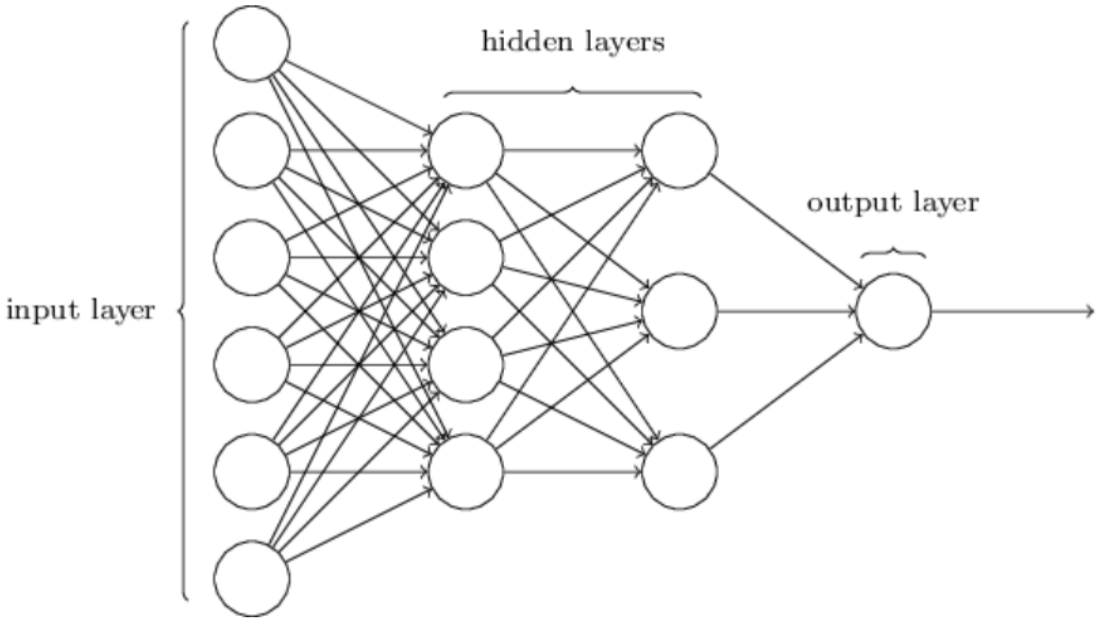

Рисунок 2.5 – Архітектура глибинної нейронної мережі

Внутрішній шар називаються прихованими, оскільки не мають зв'язків із зовнішніми свідченнями, як вузли виведення і введення. Вони включаються тільки в разі активації попередніх шарів.

Глибинне навчання – це техніка навчання нейромережі, яка використовує безліч шарів для вирішення складних проблем (наприклад, розпізнавання мови) за допомогою шаблонів. У вісімдесятих роках більшість нейромереж були одноразовими в силу високої вартості та обмеженості можливостей даних [21].

Якщо розглядати машинне навчання як відгалуження або варіант роботи штучного інтелекту, то глибинне навчання − це спеціалізований тип такого відгалуження. Машинне навчання використовує комп'ютерний інтелект, який не дає відповіді одразу. Замість цього код буде запускатися на тестових даних і, виходячи з правильності їх результатів, корегувати свій хід. Для успішності цього процесу зазвичай використовуються різноманітні техніки, спеціальне програмне забезпечення і інформатика, яка описувала статичні методи і лінійну алгебру.

Методи глибинного навчання діляться на два основних типи:

– навчання з вчителем;

– навчання без вчителя.

Перший спосіб використовує спеціально відібрані дані, щоб домогтися бажаного результату. Він вимагає досить багато людського втручання, адже дані доводиться вибирати вручну. Однак він зручний для класифікації і регресії.

Уявіть, що ви власник компанії і хочете визначити вплив премій на тривалість контрактів з вашими підлеглими. При наявності заздалегідь зібраних даних, метод навчання з учителем був би незамінний і дуже ефективний. Другий же спосіб не має на увазі заздалегідь заготовлених відповідей і алгоритмів роботи. Він спрямований на виявлення в даних прихованих шаблонів. Зазвичай його використовують для кластеризації і асоціативних задач, наприклад, для групування клієнтів по поведінці.

Основною метою навчання є отримання нейронної мережі, яка здатна у найкращий спосіб відтворювати попередньо невідоме відображення $R^n \rightarrow R^m$. Коректне налаштування не тільки синаптичних коефіцієнтів, а й архітектури нейронної мережі, зокрема налаштування кількості шарів та кількості нейронів у кожному шарі, дозволяє суттєво покращити показники такої мережі. Серед підходів до налаштування архітектури нейронної мережі виділяють:

– деструктивний підхід: за основу береться заздалегідь надлишкова модель, до неї застосовуються різні процедури, що видаляють із початкової мережі елементи;

– конструктивний підхід: за основу береться максимально проста модель до неї застосовуються процедури, що додають початковій мережі нові елементи до певного моменту, в залежності від методу, що використовується.

3 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ПРОГНОЗУВАННЯ ВИХІДНИХ ПОКАЗНИКІВ ОБ'ЄКТІВ

3.1 Нейромережеві методи прогнозування фінансових часових рядів

Завдяки своїй можливості виявляти нелінійні математичні закономірності часових рядів, швидко адаптуватися до змін ринкових тенденцій, нейронні мережі є на даний момент одним з найбільш перспективних інструментів прогнозування фінансових часових рядів. Такий підхід має ряд переваг. По-перше, нейромережевий аналіз, на відміну, наприклад, від технічного, не передбачає ніяких обмежень на характер вхідної інформації. Це можуть бути як індикатори даного часового ряду, так і відомості про поведінку інших ринкових інструментів (наприклад, останні зміни цін на нафту, зміни курсів провідних світових індексів, обсяги продажів на біржі). По-друге, нейромережевий аналіз не містить штучного підгону ринкових взаємозв'язків під маску стандартного розподілу, а тому видається більш універсальним і перспективним.

Для вирішення задачі прогнозування необхідно знайти таку нейронну мережу або комітет нейроекспертів, який би найкращим чином будував відображення $F:X \to y$, узагальнююче сформований на основі цінової динаміки набору прикладів $\{x_t,y_t\}$. Пошук такої нейронної мережі або комітету нейроекспертів здійснюється за допомогою одного або декількох алгоритмів навчання. При цьому, однак, дослідження в галузі прогнозування часових рядів за допомогою мереж тривають і в даний час. В нейронній мережі численні фактори взаємодіють досить складним чином, і успіх поки приносить тільки евристичний підхід. Сучасні методи навчання багатошарових штучних нейронних мереж мають на увазі випадкове формування первинних значень вагових коефіцієнтів. У зв'язку з цим передбачення мереж, навчених на одній і тій же вибірці даних, можуть відрізнятися. Крім того, як вже зазначалося, нейромережеве моделювання може використовувати в якості вхідних даних ще й різні фінансові та інші показники, значення яких впливають на зміну прогнозованого ряду. Оскільки виявити всі такі

чинники (і ступінь їх впливу) однозначно, як правило, неможливо, це є аргументом на користь використання не єдиної нейронної мережі, а комітету нейронних мереж.

3.1.1 Комітети мереж

Найчастіше в літературі пропонується використовувати в якості підсумкового значення комітета середнє арифметичне значення прогнозів всіх мереж-експертів [22]. Але цю ідею можна істотно розвинути. Розглянемо кілька можливих способів організації комітетів прогнозуючих нейромереж.

По-перше, можна, наприклад, навчити ще одну нейронну мережу («керівника комітету»), входами якої будуть прогнози всіх нейроекспертів, а виходом − підсумковий прогноз комітету (див. рис. 3.1).

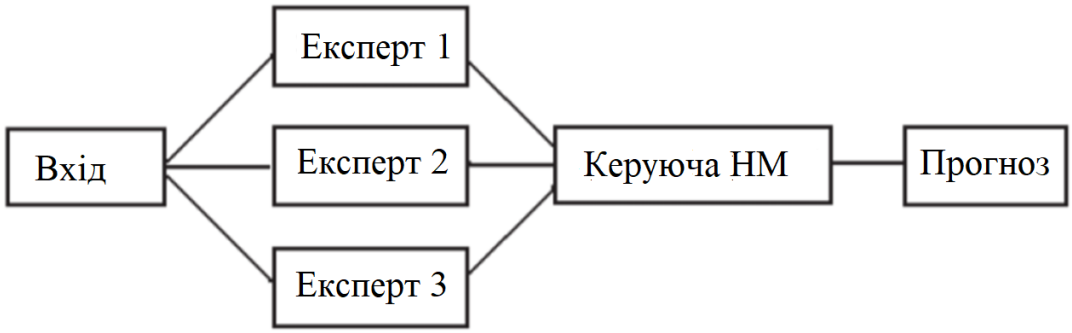

Рисунок 3.1 – Архітектура моделі прогнозування комітета нейроекспертів

По-друге, іншим підходом може бути введення поняття «спеціалізації» експертів. З цією метою пропонується провести попередню кластеризацію вхідних образів навчальної вибірки, тобто розбити вихідну вибірку на кілька (2-5) груп схожих вхідних наборів. Наприклад, в деякі групи даних можуть потрапити навчальні набори, які характеризуються зростаючим трендом, в інші − спадним і т.п. Для такої кластеризації може бути використана система, що самоорганізується,

нейронна мережа, яка називається картою Кохонена. Далі для кожного кластера виділяється, як мінімум, два нейроексперта, які навчаються тільки на даних, які потрапили в цей кластер. Таким чином, створюються підкомітети нейроекспертів, що спеціалізуються на прогнозуванні в умовах тієї чи іншої ринкової ситуації. У режимі функціонування комітету вхідний образ спочатку аналізується картою Кохонена, щоб визначити, до якого з наявних кластерів він відноситься. Потім підсумковий прогноз здійснюється тим підкомітетом, спеціалізацією якого є даний кластер. Нейромережева карта спеціалізованих експертів може використовуватися не тільки в процесі прогнозування, а й для аналітичних цілей. Зокрема, на основі карти в процесі надходження запитів можна зробити висновок про проблемні області комітету (наприклад, виявити кластери, експерти яких відрізняються гіршою якістю прогнозу). Це дає нову стратегію навчання і поповнення комітету новими моделями.

При використанні третього підходу всі нейронні мережі навчаються на одних і тих же даних, але в комітет спочатку включаються мережі, що відрізняються принципово різною архітектурою (багатошарові перцептрони, рекурентні мережі, радіально-базисні мережі і будь-які інші, які добре зарекомендували себе при вирішенні задач прогнозування) . Кожна з цих мереж по-своєму вирішує задачу нелінійної апроксимації відображення $F:X \to y$, де X − вхідний вектор мережі, y − вихідне (прогнозоване) значення. Тому різниця в прогнозах, які видаються експертами, буде наслідком не тільки випадковості початкових значень вагових коефіцієнтів, а й принципової різниці цих мереж. Після завершення етапу навчання всі вхідні дані навчальної вибірки кластеризуються, як і при використанні попереднього підходу. А потім для кожного нейроексперта визначається коефіцієнт його компетентності на даних кожного кластера (наприклад, в процесі експериментального тестування було відмічено, що ймовірнісна мережа забезпечує більш високу якість прогнозу в умовах наявності зростаючого тренду курсу акції, а мережа, навчена за алгоритмом зворотного поширення, навпаки, спадної). В процесі функціонування мережі коефіцієнти компетентності корегуються в залежності від величини помилок прогнозу нейроексперта на даних цього кластера.

Об'єднання експертів в ансамбль при вирішенні завдання підсумкового прогнозування проводиться з вагами, відповідними коефіцієнтами компетентності нейроекспертів для того кластера, в який потрапляє аналізований вхідний вектор. Тобто в якості значення комітету береться зважене середнє арифметичне значення прогнозів всіх мереж-експертів

У четвертому варіанті як відповідь комітету використовується прогноз тієї з *P* наявних в розпорядженні мереж, яка в даний момент *t* є найкращою в тому сенсі, що для неї сума квадратів помилок прогнозів за *k* періодів, мінімальна:

$$\sum_{s=1}^{k}(y_i(t-s)-y(t-s))^2 \to \min_{1\le i\le P}. \qquad (3.1)$$

де $y_i(t-s)$ − прогноз мережі з номером і в момент часу $(t-s)$,

$y(t-s)$ − справжнє значення ряду в цей момент.

Величина *k* задає швидкість забування передісторії і, як показують чисельні експерименти, її найкращі значення знаходяться в проміжку $k \in [3,5]$.

3.2.2 Використання методів BOOSTING

Ідея бустінга (посилення ансамблю класифікаторів) вперше була запропонована І. Френдом і Р. Шапіром [23]. Сенс постановки проблеми полягає в наступному. Нехай є деякий алгоритм класифікації об'єктів за двома класами, точність якого для заданого набору даних лише незначно перевищує точність вгадування:

$$p = 1/2 + \varepsilon. \qquad (3.2)$$

Чи можлива практична реалізація програми навчання цього класифікатора, таким чином, що для будь-яких $0<\gamma, \delta<1/2$ з ймовірністю більше $1-\gamma$ помилка класифікатора виявиться менше δ?

В перших варіантах бустінга (BOOST1) розглядалися трійки послідовно навчаючихся алгоритмів, що досягають рівня помилки $\alpha = (1-p)<1/2$[24]. Нижче наводиться алгоритм застосування методу BOOST1 для керування ансамблем, що складається з трьох нейронних мереж:

- перша нейронна мережа навчається на множині з m прикладів;
- друга мережа також навчається на m прикладах, які обирають так, що перша мережа рівно на половині з них дає точну відповідь;
- третя мережа навчається на таких m прикладах, на яких відповіді першої та другої мереж розходяться;

Прогнозом комітету є відповідь перших двох мереж, якщо він однаковий, і відповідь третьої мережі, якщо відповіді першої та другої мереж розійшлися.

Якщо, як було зазначено, рівень помилки на тестовій множині кожної з нейронних мереж дорівнює α, то ймовірність помилки для комітету буде дорівнює $3\alpha^2-2\alpha^3 < \alpha$ (при $0<\alpha<1/2$), тобто менше помилки кожної з мереж. Відзначимо, що метод BOOST1 вимагає, щоб кожна з мереж комітету здійснювала класифікацію навчальної множини виключно на два класи, що знижує його застосовність для вирішення задачі прогнозування. Однак, наприклад, в системах трейдингу буває достатньо, щоб алгоритм прогнозування видавав сигнал до покупки (якщо очікується зростання котирування) або до продажу (якщо очікується падіння).

Метод AdaBoost, також запропонований Френдом і Шапіро [24], дозволяє об'єднувати в комітет довільну кількість мереж, але їх навчання і в цьому випадку має проводитися послідовно. При використанні методу AdaBoost для управління комітетом всі мережі можуть навчатися на одній й тій самій множині, але навчальні вектори набувають вагові коефіцієнти, які змінюються з плином часу. А саме, для кожної мережі комітету, починаючи з другої, ваги (тобто значимість) навчальних векторів перераховуються так, щоб вона точніше налаштовувалася на тих векторах, на яких частіше помилялися всі попередні мережі.

Як і в методі BOOST1, кожна з мереж комітету повинна здійснювати класифікацію навчальної множини на два класи. Відповіддю комітету є зважене середнє значення прогнозів всіх мереж-експертів, округлюються в бік найближчої допустимої відповіді мережі. Ваги мереж-експертів обчислюються, виходячи з числа допущених ними помилок. Узагальнююча здатність бустінга (стосовно алгоритмів класифікації) досліджена досить добре. У багатьох випадках якість класифікації на тестовій вибірці може продовжувати поліпшуватися навіть після досягнення безпомилкового розпізнавання навчальної вибірки. Бустінг будує опуклу комбінацію класифікаторів, яка проявляє властивість стабільності, тобто невеликі варіації навчальної вибірки призводять до незначних змін одержуваного результату.

Як показують обчислювальні експерименти, використання комітетів нейроекспертів здатне на 20-30% збільшити точність базових прогнозів.

Найкращі результати поки показують 3-й і 4-й із вище описаних методів формування нейромережевих комітетів.

3.2 Використання нейронних мереж для прогнозування часових рядів

До проблем аналізу часових рядів відноситься те, що вони часто зашумлені, не стаціонарні, мають високу розмірність, не завжди достатньо інформації для аналізу і, нарешті, прогноз майбутніх значень ряду повністю залежить від ознак, що змінюються в часі. В даний час великою популярністю користуються так звані мережі третього покоління − глибинні нейронні мережі (Deep Neural Network), до них відносяться: обмежена машина Больцмана (Restricted Boltzmann Machine), глибинні мережі довіри (Deep Belief Network), автоенкодери (Autoencoder), згорткові нейронні мережі (Convolutional Neural Networks) і їх модифікації [25]. Ці мережі добре себе показали у вирішенні складних сучасних задач штучного інтелекту. Однак увага дослідників приділялася розробці моделей для статичних даних і не так багато для часових рядів, зокрема, фінансових [26].

Першим кроком до розуміння того, як працює глибинне навчання, є розуміння відмінностей між важливими термінами.

Коли вперше почалося вивчення штучного інтелекту (ШІ), дослідники намагалися відтворити людський інтелект для конкретних задач: наприклад, грати в гру. Було введено величезну кількість правил, яким повинен був слідувати комп'ютер. ШІ надавався конкретний перелік можливих дій, і комп'ютер приймав рішення на основі цих правил.

Machine Learning (ML) відноситься до здатності машини вчитися з використанням великих наборів даних замість жорстко закодованих правил. ML дозволяє комп'ютерам самостійно вчитися. Цей тип навчання використовує переваги обчислювальної потужності сучасних машин, які можуть легко обробляти великі набори даних.

Кероване навчання включає в себе використання відмічених наборів даних, які містять входи і очікувані результати. Коли ви тренуєте ШІ за допомогою керованого навчання, ви даєте йому уведення і повідомляєте очікуваний результат. Якщо результат, що генерується штучним інтелектом, помилковий, він скоректує свої обчислення. Цей процес виконується інтерактивно для набору даних, поки ШІ не перестане робити помилки.

Прикладом керованого навчання є передбачення погоди. ШІ вчиться прогнозувати погоду з використанням історичних даних. Дані навчання мають входи (тиск, вологість, швидкість вітру) та виходи − результат (температура).

Некероване навчання − це задача машинного навчання з використанням наборів даних, але без готових відповідей. Виходів просто немає. Коли ви тренуєте ШІ з використанням некерованого навчання, ви дозволяєте йому робити логічні класифікації даних. Прикладом неконтрольованого навчання є ШІ, який прогнозує поведінку користувача для веб-сайту електронної комерції. Він не буде навчатися, використовуючи позначений набір даних входів і виходів. Замість цього він створить свою власну класифікацію даних і визначить, які користувачі найчастіше купують ті чи інші продукти.

Глибинне навчання − це метод ML, який дозволяє навчати ШІ для прогнозування виходів з урахуванням набору вхідних даних. Для цих цілей можна використовувати як кероване, так і не кероване навчання.

Для розуміння як працює Deep Learning, створимо гіпотетичну послугу оцінки вартості квитка на літак (див. рис. 3.2).

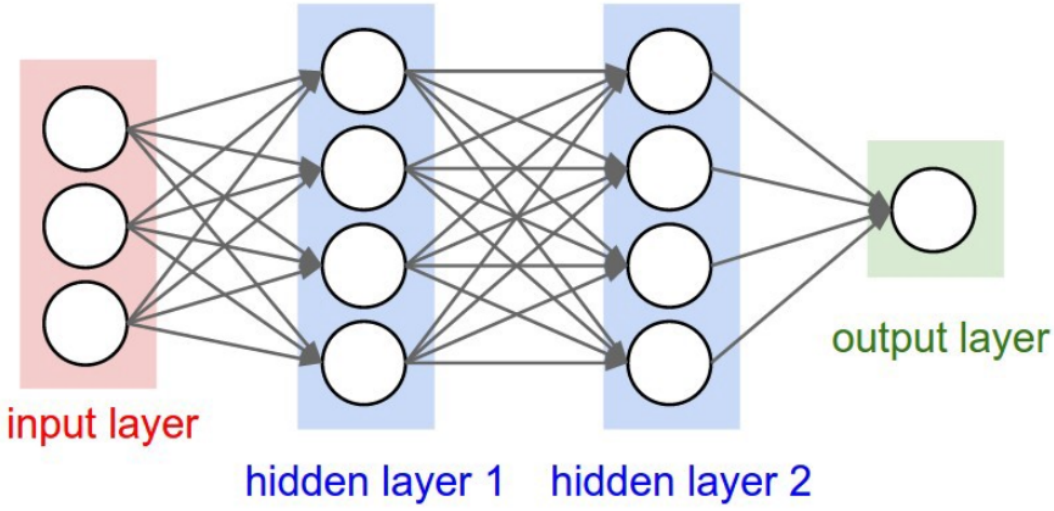

Рисунок 3.2 – Архітектура нейронної мережі для оцінки вартості квитка на літак

Ми будемо навчати його за допомогою керованого методу навчання. Оцінювач прайсів буде розраховувати вартість, використовуючи наступні входи: початковий аеропорт, аеропорт призначення, дата від'їзду, авіакомпанія.

Як і у тварин, в мозку задіяні нейрони. Вони представлені колами. Такі нейрони є взаємопов'язаними. Нейрони згруповані в три різних типи шарів: вхідний шар, прихований шар і вихідний шар.

Вхідний шар приймає спочатку надані дані. У нашому випадку тут є чотири нейрона: початковий аеропорт, аеропорт призначення, дата відбуття і авіакомпанія. Вхідний шар передає входи в перший прихований шар.

Приховані шари виконують математичні обчислення на входах. Однією з проблем при створенні нейронних мереж є визначення кількості прихованих шарів,

а також кількості нейронів для кожного такого шару. «Глибина» в Deep Learning відноситься до наявності більш ніж одного прихованого шару.

Вихідний шар повертає результат (вихідні дані). У нашому випадку це прогноз вартості квитка. Архітектура нейронної мережі для прогнозу вартості квитка зображена на рисунку 3.3

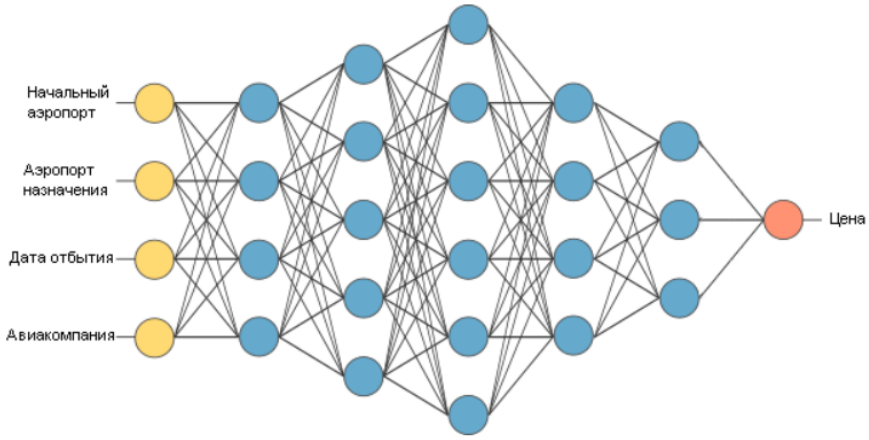

Рисунок 3.3 – Архітектура нейронної мережі для прогнозу вартості квитка

Кожне з'єднання між нейронами пов'язано з вагою (див. рис. 3.4).

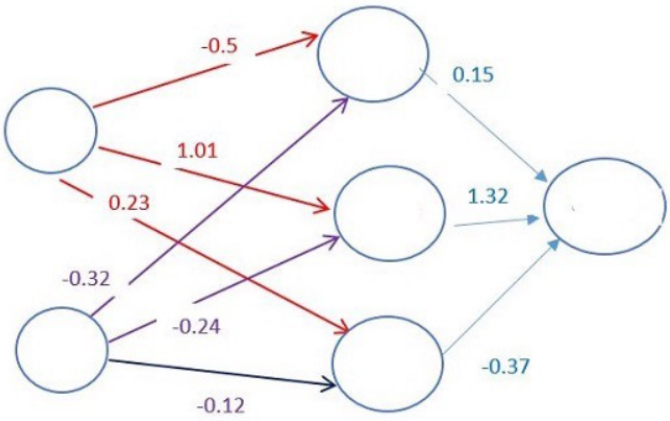

Рисунок 3.4 – Архітектура нейронної мережі зі значеннями синаптичних ваг

Ця вага визначає важливість вхідного значення. Вихідна вага задається випадковим чином. При прогнозуванні ціни на авіаквиток дата вильоту є одним з

найбільш значущих факторів. Отже, з'єднання нейронів відправлення матимуть велику вагу (значення).

Кожен нейрон має функцію активації. Коли набір вхідних даних пройшов всі шари нейронної мережі, він повертає вихідні дані через відповідний шар.

Навчання ШІ − це найскладніша частина Deep Learning, тому що потрібен великий набір даних, а також потрібна велика кількість обчислювальної потужності.

Для оцінки вартості авіаквитка потрібно знайти історичні дані про ціни на квитки. Через велику кількість можливих аеропортів і комбінацій дати вильоту потрібно величезний список цін.

Щоб навчити ШІ, необхідно надати йому дані з набору і порівняти його виходи з виходами з набору даних. Оскільки ШІ ще не навчений, його виходи будуть помилковими. Як тільки буде пройдений весь набір даних, ми зможемо створити функцію, яка показує, як сильно відрізняються результати ШІ від реальних результатів. Ця функція називається функцією витрат або вартості.

Метою є зменшення значення цієї функції до нуля. Для того, щоб зменшити функцію вартості треба змінювати вагу, це можна було б зробити в випадковому порядку, поки функція вартості не знизиться максимально, але це не дуже ефективний спосіб.

Для досягнення зменшення функції вартості необхідно використовувати метод градієнтного спуск. Метод градієнтного спуску − це метод, який дозволяє знайти мінімум функції, як це показано на рисунку 3.5.

У нашому випадку це функція вартості. Метод працює, трохи змінюючи вагу після кожної ітерації для набору даних. Обчислюючи похідну (або градієнт) функції вартості при певному наборі ваги, ми можемо бачити, в якому напрямку знаходиться мінімум. Щоб звести до мінімуму функцію вартості, доведеться багаторазово перебирати дані. Ось чому потрібна велика обчислювальна потужність.

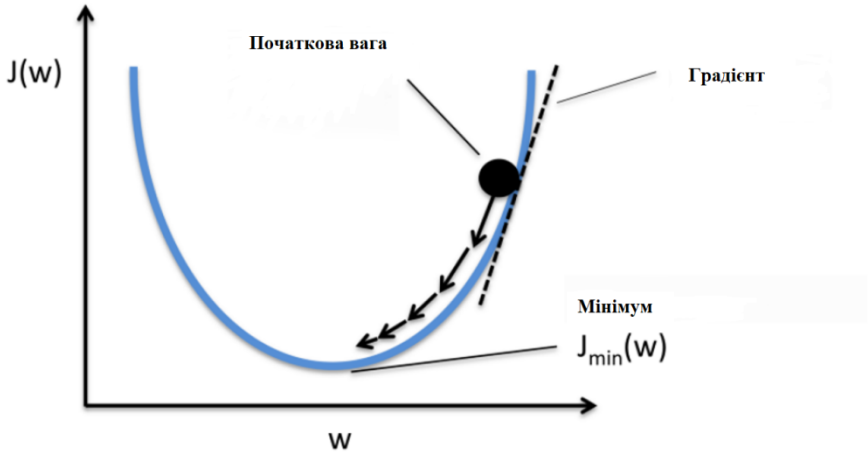

Рисунок 3.5 – Візуалізація методу градієнтного спуску

Оновлення ваги за допомогою градієнтного спуску виконується автоматично. Саме це є перевагами Deep Learning. Після обчислення оцінки вартості авіаквитків, можна використовувати її для прогнозування майбутніх цін.

3.3 Критерії та методика оцінювання ефективності. Порівняльний аналіз показників ефективності нейромережевих моделей прогнозування

Методика оцінювання ефективності нейромережевих методів і моделей полягає у наступній послідовності дій:

– здійснити вибір критеріїв оцінювання ефективності нейромережевих методів прогнозування;

– виконати моделювання точності прогнозування для моделей, обраних за визначеними критеріями;

– здійснити вибір тієї моделі, яка демонструє найбільшу точність прогнозування;

Для оцінювання ефективності нейромережевих моделей прогнозування були визначені наступні критерії:

– точність відноситься до близькості двох або більше вимірювань один до одного. Це повторюваність або відтворюваність вимірювання;

– чутливість посилається на частку релевантних екземплярів, котрі були отримані з загальної кількості екземплярів;

– ROC-крива (Receiver operating characteristic, робоча характеристика приймача) – графік, що дозволяє оцінити якість бінарної класифікації, відображає співвідношення між часткою об'єктів від загальної кількості носіїв ознаки, вірно класифікованих і часток об'єктів від загальної кількості об'єктів помилково класифікованих. Кількісну інтерпретацію ROC дає показник AUC (Area under ROC curve, площа під ROC-кривої) – площа, обмежена ROC-кривої і віссю частки помилкових позитивних класифікацій. Чим вище показник AUC, тим якісніше класифікатор, при цьому значення 0,5 демонструє непридатність обраного методу класифікації. Значення менше 0,5 показує, що класифікатор діє з точністю до навпаки: якщо позитивні назвати негативними і навпаки, класифікатор буде працювати краще.

Для підтвердження працездатності розробленої моделі прогнозування, на основі нейронних мереж (НМ), була вирішена задача прогнозування на основі тестової вибірки «Iris» з UCI-репозиторія [27].

Тестова вибірка «Iris». Вибірка складається з даних про 150 примірників ірису, по 50 примірників з трьох видів − ірис щетинистий ірис віргінський і ірис різнокольоровий. Для кожного екземпляра вимірювалися чотири характеристики: довжина чашолистки, ширина чашолистки, довжина пелюстки та ширина пелюстки.

З метою оцінки ефективності розробленої моделі прогнозування, на основі багатошарового перцептрона (MLP), результати прогнозування були порівняні з згортковою нейронною мережею GoogLeNet [28], Inception-V3 [29], ResNet [30]. GoogLeNet є згортковою нейронною мережею, яка не тільки показує високу точність, але і має порівняно малу кількістю обчислень. Важливо відзначити, що така мережа збільшується не тільки в глибину, але і в ширину, шляхом паралельного використання згорток різного масштабу. Архітектура Inception-V3

використовує ту ж концепцію, що і GoogLeNet, намагаючись збільшувати глибину. Нейромережа ResNet є набагато глибшою в порівнянні з попередніми архітектурами. Результати прогнозування приведені у таблиці 3.1:

Таблиця 3.1 − Порівняння критеріїв ефективності прогнозування на тестових вибірках

Модель прогнозування	Точність	Чутливість	AUC
MLP	89.44%	88.73%	0.93
GoogLeNet	86.11%	87.10%	0.92
Inception-V3	84.44%	86.44%	0.91
ResNet	82.78%	85.51%	0.89

Як бачимо з цієї таблиці, результати прогнозування запропонованої моделі на основі багатошарового перцептрона мають вищу точність, ніж такі архітектури, як GoogLeNet, Inception-V3, ResNet.

Ми застосували найпростішу архітектуру нейронних мереж для прогнозування вихідних показників об'єктів. В якості базової моделі використовували багатошаровий перцептрон, в якості фреймворка для імплементації використовували нейромережеву бібліотеку Keras. На основі порівняння результатів графіків похибок і точності навчання нейронної мережі при вирішенні задач класифікації та регресії, було правильно обрано алгоритм нормування даних, визначено архітектуру мережі, оцінено якість роботи алгоритму.

3.4 Розробка нейромережевої моделі для прогнозування цін акцій компаній

Основними завданнями цієї роботи було розроблення моделі, яка може не тільки передбачити наступний крок часу, але й створити послідовність прогнозів і

використовувати множинні часові ряди разом з набором статичних (скалярних) функцій в якості вхідних даних. На високому рівні ця модель використовує досить стандартну послідовність-до-послідовності архітектуру нейронної мережі. Її вхідні дані є минулими значеннями прогнозованих часових рядів, пов'язаних з іншими значеннями часових рядів і вставками часової мітки. Якщо доступні статичні особливості, то модель може використовувати їх і для визначення прогнозу.

Кодер використовується для кодування входів часових рядів з їх відповідними вставками часової мітки до векторного представлення фіксованого розміру. Вона також виробляє латентні вектори для окремих кроків часу, які використовуються пізніше у декодері. Для цього я використав багатошарову однонаправлену нейронну мережу, де всі шари, крім першого, є залишковими.

У деяких випадках у вас можуть бути вхідні послідовності, які є надто великими і можуть призвести до помилки навчання через проблеми з пам'яттю або істотного уповільнення. Щоб вирішити цю проблему, модель згортає вхідну послідовність з одновимірною згорткою, яка має той же розмір ядра і просувається, перш ніж подавати її в кодер нейронної мережі. Це зменшує вхід нейронної мережі на коефіцієнт n, де n - розмір ядра згортки.

Контекстний рівень розташований між кодером входів і шаром декодера. Він об'єднує кінцевий стан кодера зі статичними властивостями і статичними вбудовуваннями і видає вектор фіксованого розміру, який потім використовується як початковий стан для декодера (див. рис. 3.6).

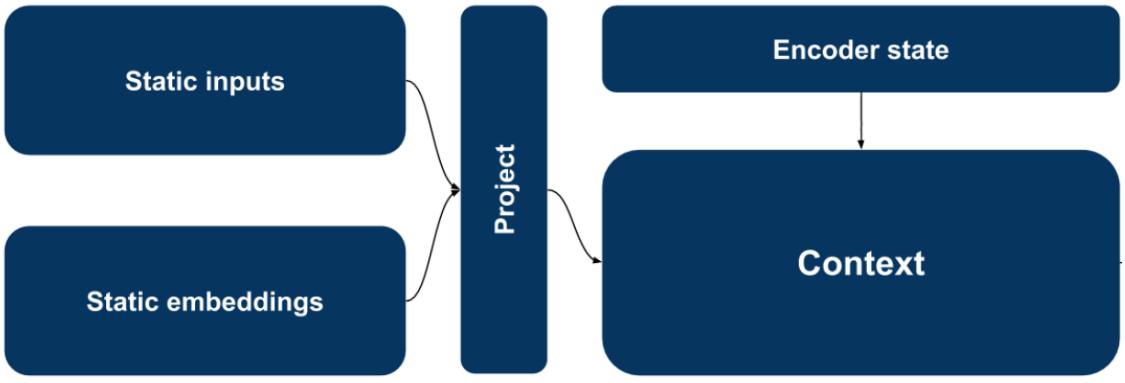

Рисунок 3.6 – Структура взаємодії компонентів

Декодерний шар реалізований як авторегресивна нейронна мережа. Вхідні дані на кожному кроці - це конкатенація попереднього значення послідовності і тимчасова мітка, вбудована для цього кроку. Подача тимчасової мітки в декодер допомагає моделі вивчати шаблони в сезонних даних.

На першому кроці кодер приймає контекст як початкове значення і конкатенацію початкового значення послідовності і першу мітку часу, вбудовує в якості вхідного елемента. Потім перший шар видає запит, який подається на модуль, який виводить стан, який потім використовується як стан на наступному етапі. Виходи декодера є необробленими прогнозованими значеннями, які потім подаються на наступний крок разом з тимчасовою міткою, вбудованою для цього кроку.

Проміжок функції часового ряду повинен не перетинатися з діапазоном цілей і що остання мітка показу безпосередньо перед першою міткою часу. Крім того, якщо у вас є які-небудь статичні функції (наприклад, сукупні статистичні дані), їх потрібно об'єднати до останньої мітки часу. Дані повинні бути нормалізовані у відповідний діапазон для моделі нейронної мережі.

Для цільового значення існують кілька варіантів нормалізації. Можна, наприклад, прогнозувати відносну зміну від останнього вхідного значення (може бути проблемою у випадку, якщо це 0) або нормалізувати абсолютні значення, використовуючи аналогічний підхід, описаний вище для функцій.

4 ПРОГРАМА ВИЗНАЧЕННЯ ЕФЕКТИВНОСТІ МОДЕЛЕЙ ПРОГНОЗУВАННЯ ВИХІДНИХ ПОКАЗНИКІВ ОБ'ЄКТІВ

Для візуалізації ефективності моделей прогнозування вихідних показників об'єктів було розроблено web-додаток з використанням мови програмування Python, фреймворку Django та нейромережевої бібліотеки Keras.

Keras − відкрита нейромережева бібліотека, написана на мові Python [31]. Вона являє собою надбудову над фреймворками Deeplearning, TensorFlow і Theano. Націлена на оперативну роботу з мережами глибинного навчання.

Підготовка даних. Для прикладу були взяті ціни акцій компанії Shopify з 2015 року, які зображено на рисунку 4.1. Їх можна завантажити на Yahoo Finance в форматі .csv.

Рисунок 4.1 – Графік змінення цін акцій компанії Shopify з 2015 року

Графік має вигляд випадкового процесу. Для того, щоб вирішити задачу прогнозування на день або декілька днів вперед, необхідно задачу прогнозування описати ближче до задач машинного навчання. Метою є прогнозування значення ціни на наступний день (або через пару днів), прогнозування змінення ціни на

наступний день в порівнянні з останнім днем або логарифм цієї різниці, все це є задачею регресії. При вирішенні задач регресії виникає проблема нормалізації даних.

У разі класифікації або у разі задачі регресії, на вхід ми візьмемо вікно часового ряду (30 днів) і передбачимо рух ціни на наступний день (класифікація) або значення зміни ціни (регресія).

Після вирішення задачі нормалізації графік значення змінення цін має наступний вигляд (див. рис. 4.2):

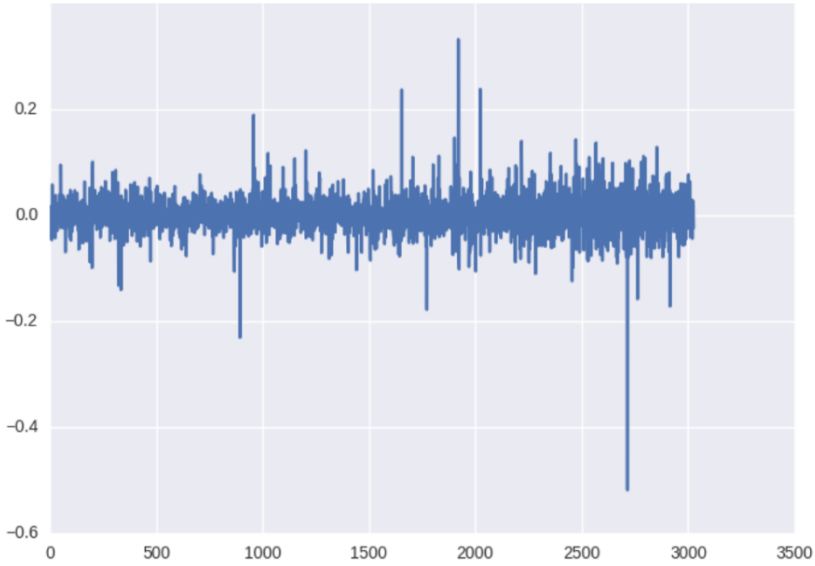

Рисунок 4.2 – Графік нормалізації значень змінення цін

Значення знаходяться в інтервалі [-0.5; 0.5]. Для поділу на навчальну і тренувальну вибірку візьмемо перші 85% вікон у часі для навчання і останні 15% для перевірки роботи нейронної мережі. Для навчання нейронної мережі ми отримаємо наступні пари X, Y: ціни в момент закриття ринку за 30 днів і [1, 0] або [0, 1] в залежності від того, зросло або впало значення ціни для бінарної класифікації; процентна зміна цін за 30 днів і зміна на наступний день для регресії.

В якості базової моделі використовується багатошаровий перцептрон. В якості фреймворка для імплементації використовується нейромережева бібліотека Keras − проста, інтуїтивно зрозуміла, за допомогою якої можна реалізовувати

досить складні обчислювальні графи. Реалізуємо сітку − вхідний шар з 30 нейронами (довжина вікна), перший прихований шар з 64 нейронами, після нього шар нормалізації, потім активаційна функція, далі вихідний шар з 1 нейроном (у разі задачі регресії), з 2 нейронами (у разі задачі класифікації).

Для задачі регресії в кінці параметр активації повинен бути лінійним. Далі визначаються функції помилки і алгоритм оптимізації. Довжина кроку градієнтного спуску є 0.001. Для нормалізації параметру втрат в задачах класифікації потрібно поставити крос-ентропію, а для задачі регресії − середню квадратичну помилку. Слід зазначити, що Keras дозволяє контролювати процес навчання за рахунок зменшення шагу градієнтного спуску, у разі, якщо результати навчання не покращуються ми використовуємо функцію ReduceLROnPlateau для досягнення кращих результатів навчання. Приклад коду з використанням функції для зменшення кроку градієнтного спуску наведено на рисунку 4.3.

```
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.9, patience=5, min_lr=0.0000
01, verbose=1)
model.compile(optimizer=opt,
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Рисунок 4.3 – Функція для зменшення кроку градієнтного спуску

Важливим моментом є навчання. Вчити алгоритми на таких даних треба довше 50-100 епох. Це пов'язано з тим, що якщо навчати на 5-10 епохах, точність буде 55%. Якщо провести аналіз тренувальних даних, буде видно, що 55% вікон були для одного патерну (підвищення, наприклад), а решта 45% − для іншого (зниження). У нашому випадку 53% вікон класу "зниження", а 47% − "підвищення", тому нашою метою є отримання точності вище 53%.

На рисунках 4.4, 4.5 представлені графіки які відображаються у web-додатку для відображення змін значень функції похибки та точності навчання нейронної мережі.

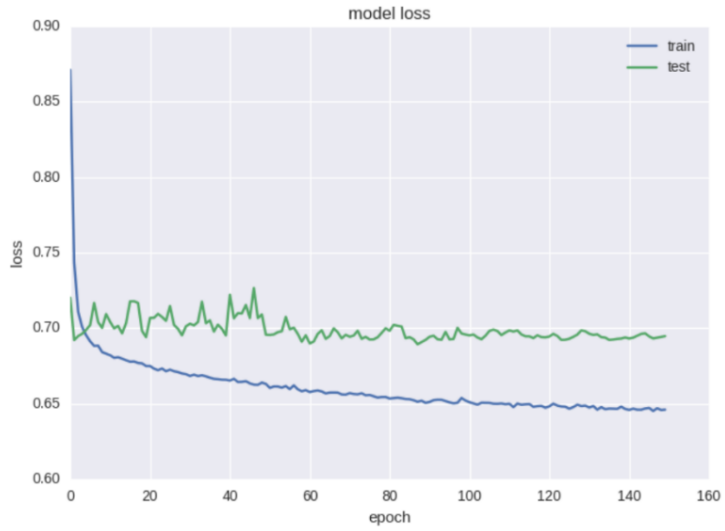

Рисунок 4.4 – Графік зміни значень функції похибки навчання нейронної мережі

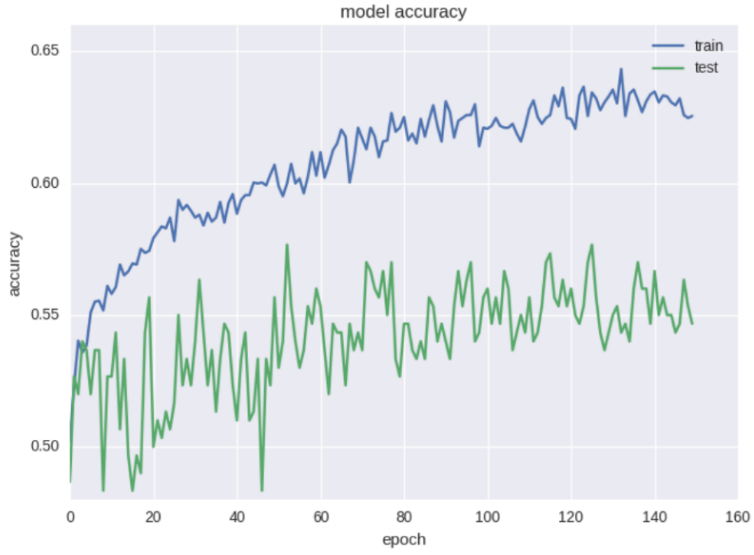

Рисунок 4.5 – Графік зміни значень точності навчання нейронної мережі

Як видно з рисунків 4.4 і 4.5, похибка і точність для тестової вибірки весь час залишається на приблизно одному значенні. При навчанні на тренувальній виборці, похибка падає, а точність зростає, що говорить про перенавчання.

Якщо з'являється ефект перенавчання, потрібно додати регуляцію. Під час перенавчання будується модель, яка "запам'ятовує" тренувальні дані і не дозволяє узагальнити знання на нові дані. В процесі регуляції накладаються певні обмеження на ваги нейронної мережі, щоб не було великого розкиду в значеннях і,

незважаючи на велику кількість параметрів (тобто ваг мережі), частина з них перетворюється на нуль для спрощення. Почнемо з самого поширеного способу − додавання до функції похибки додаткової складової з L2 нормою за сумою ваг, в Keras це робиться за допомогою keras.regularizers.activity_regularizer. Приклад коду для регуляції ваг нейронної мережі наведено на рисунку 4.6.

```
model = Sequential()
model.add(Dense(64, input_dim=30,
                activity_regularizer=regularizers.l2(0.01)))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(16,
                activity_regularizer=regularizers.l2(0.01)))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(2))
model.add(Activation('softmax'))
```

Рисунок 4.6 – Регуляція ваг нейронної мережі

З точки зору функції похибки, така нейронна мережа навчається краще але точність все ще страждає. Варто додати ще більше регуляції за допомогою популярної в останні роки техніки Dropout (метод вирішення проблеми перенавчання в нейронних мережах) − це випадкове "ігнорування" деяких ваг в процесі навчання, щоб уникнути ко-адаптації нейронів (щоб вони не вивчали однакові ознаки) [32].

Dropout зазвичай не додають між вхідним шаром і першим прихованим, так як в цьому випадку ми будемо навчатися лише на зашумлених даних, і також не додається безпосередньо перед виходом. Графік зміни значень функції похибки та точності навчання нейронної мережі з використанням техніки Dropout зображено на рисунках 4.7, 4.8.

Як бачимо, графіки помилки і точності вже мають кращі результати, якщо зупинити навчання мережі трохи раніше, можна отримати 58% точності передбачення руху ціни.

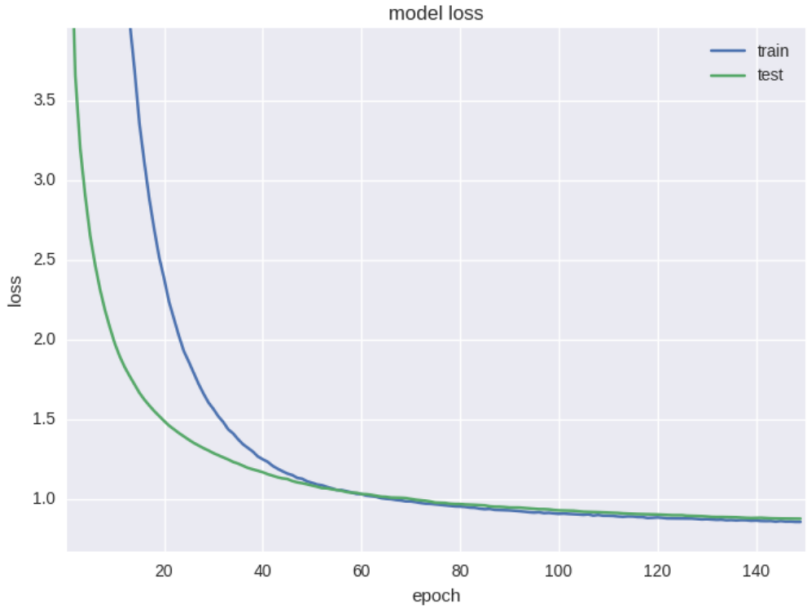

Рисунок 4.7 – Графік зміни значень функції похибки навчання нейронної мережі з використанням техніки Dropout для регуляції ваг нейронної мережі

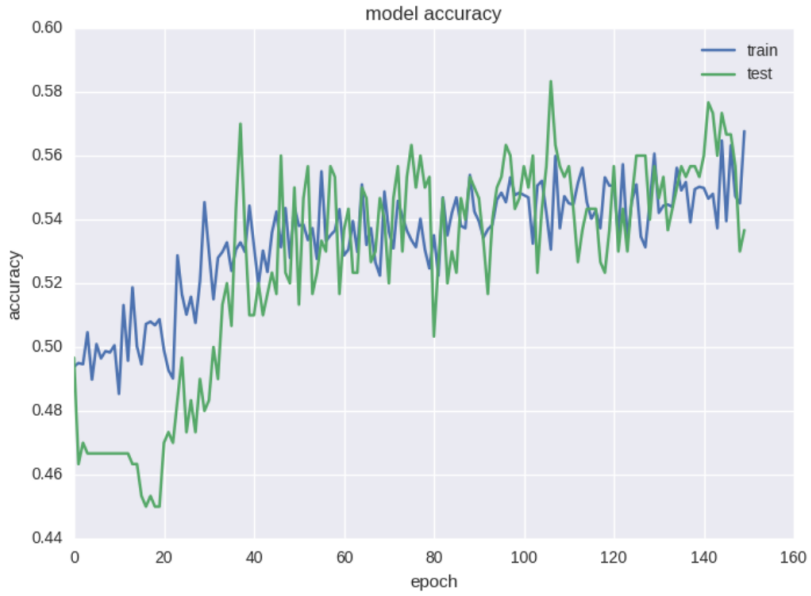

Рисунок 4.8 – Графік зміни значень точності навчання нейронної мережі з використанням техніки Dropout для регуляції ваг нейронної мережі

Ще один цікавий і інтуїтивно зрозумілий момент прогнозування фінансових часових рядів полягає в тому, що коливання в наступний день має випадкову

природу, але коли ми дивимося на графіки, ми все-таки можемо помічати тренд на наступні 5-10 днів.

Якщо зупинити навчання досить рано, то можна отримати 60% точності, що є дуже непогано. Для задачі регресії візьмемо нашу останню успішну архітектуру для класифікації (вона вже показала, що вміє вивчати необхідні ознаки), приберемо Dropout і навчимо на більшій кількості ітерацій. Також в даному випадку ми можемо дивитися вже не тільки на значення похибки, а й візуально оцінити якість прогнозування (див. рис. 4.9):

Рисунок 4.9 – Результати навчання нейронної мережі

Тестування програмного забезпечення – це процес дослідження програмного забезпечення з метою виявлення помилок і перевірки його якості. Також тестування ПЗ можна описати як процес валідації та верифікації того чи іншого програмного продукту, щоб дізнатися, на скільки точно він задовольняє всім технічним вимогам.

Тестування ПЗ може проводитися на будь-якому етапі розробки, але найчастіше це відбувається по закінченню процесу кодування. Верифікація – це процес оцінки системи або її компонентів з метою визначити чи задовольняють результати поточного етапу розробки умовам, сформованим на початку цього

етапу. Тобто чи виконуються цілі, терміни, завдання з розробки проекту, визначені на початку поточної фази.

Повинен бути здійснений контроль якості web-додатка в цілому, а також його окремих частин (інтерфейс головної сторінки, інтерфейс тестування, функціонал пошуку сумісних користувачів), і серверної частини системи.

Було проведено конфігураційне та навантажувальне тестування системи. Для конфігураційного тестування була проведена перевірка працездатності програмного продукту для різних видів браузерів, перевірка розташування елементів при різних розширеннях екрану/браузеру, результати якої наведено в таблиці 4.1.

Таблиця 4.1 – Тестування крос-браузерності web-додатку

Браузер	Результат
Google Chrome 50 (50.0.2661.102 m)	Відображення елементів web-додатку без якихось помилок і повний робочій функціонал інтерфейсу
Mozilla Firefox 44.0	
Opera 37 (37.0.2178.32)	
Internet Explorer 11 (11.0.9600.18321)	

З таблиці 4.1 можна зробити висновок, що коректна робота додатку не залежить від браузера, що використовується для роботи.

Завдяки використанню фреймворка Bootstrap, що надає інструменти для створення адаптивного дизайну, для побудови користувацького інтерфейсу при зміні розширення екрану суттєвих порушень та зміщень елементів інтерфейсу не відбувається, що зрозуміло з таблиці 4.2.

Метою тестування розробленого web-додатку є позбавлення помилок та недоліків, виявлення недоліків інтерфейсу і забезпечення коректного відображення функціоналу на усіх можливих конфігураціях комп’ютера.

Таблиця 4.2 – Результати тестування різних розширень екрану

Пропорції екрану	Важливість	Результат
16:10 (1280x800)	Критично	Коректне відображення усіх елементів web-додатку
16:9 (1920x1080)	Критично	
5:3 (1200x720)	Важливо	
4:3 (1024x768)	Критично	Деякі елементу не можливо розмістити на екрані через недостатню ширину, тому необхідно скористатися горизонтальною панеллю прокрутки
3:2 (960x640)	Важливо	

Після внесення змін до чергову версію програми, регресивні тести підтверджують, що зроблені зміни не вплинули на працездатність решті функціональності додатку. Регресійне тестування може виконуватися як вручну, так і засобами автоматизації тестування.

Під час проведення тестування програмного забезпечення ніяких критичних дефектів не виявлено.

ВИСНОВКИ

Атестаційна робота присвячена актуальній темі – дослідженню ефективності використання нейромережевих методів і моделей для прогнозування вихідних показників об’єктів. В процесі виконання атестаційної роботи проведено аналіз предметної галузі та виявлено існуючі проблеми та шляхи їх вирішення.

Розглянуті нейромережеві методи і моделі прогнозування, виявлені їх недоліки та переваги.

Розроблена методика оцінювання ефективності нейромережевих моделей, сформульовані критерії ефективності та їх вплив на вибір типу моделі.

Розроблена програма для визначення показників ефективності моделей прогнозування вихідних показників об’єктів. Виконано порівняльний аналіз ефективності нейромережевих моделей прогнозування MLP, GoogLeNet, Inception-V3 та ResNet відповідно до обраних критеріїв оцінювання, що дозволяє обрати найбільш точну модель прогнозування.

Базуючись на результатах проведеного дослідження, розроблена нейромережева модель прогнозування руху цін акцій компаній на ринку. В якості базової моделі використовувався багатошаровий перцептрон, в якості фреймворка для імплементації використовувалась нейромережева бібліотека Keras. На основі порівняння результатів графіків похибок і точності навчання нейронної мережі при вирішенні задач класифікації та регресії, було правильно обрано алгоритм нормування даних, визначено архітектуру мережі, оцінено якість роботи алгоритму.

За результатами досліджень опублікована стаття в міжнародному науковому електронному журналу «Наука онлайн».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. R. Kruse, C. Borgelt, F. Klawonn. Computational Intelligence. A Methodological Introduction. Springer, 2013. – 383 с.

2. R.J. Schalkoff. Artificial Neural Networks. NY : The McGraw-Hill Comp., Inc., 1997. – 544 с.

3. R. M. Golden. Mathematical Methods for Neural Network Analysis and Design. Cambridge, Massachussetts: The MIT Press., 1996. – 287 с.

4. H. Braun. Neuronale Netze. Optimierung durch Lernen und Evolution. Berlin: Springer-Verlag., 1997. – 345 с.

5. D.C. Dracopoulos. Evolutionary Learning Algorithms for Neural Adaptive Control. Evolutionary Learning Algorithms for Neural Adaptive Control. – Berlin: Springer-Verlag, 1997. – 183 с.

6. A. J. Shepherd. Second-Order Methods for Neural Networks. London: Springer–Verlag, 1997. – 232 с.

7. S. Haykin. Neural Networks. A Comprehensive Foundation. Upper Saddle River, N.J.: Prentice Hall, Inc, 1999. – 383 с.

8. R. Rojas. Neural Networks. A Systematic Introduction. Berlin: Springer-Verlag, 1996. – 127 с.

9. J. Moody, C.J. Darken. Fast learning in networks of locally-tuned processing units. Neural Computation, 1989. – 221 с.

10. D. Zahirniak, R. Chapman, S. Rogers, B. Suter, M. Kabritsky, V.Piati. Pattern recognition using radial basis function. Proc 6th Ann. Aerospace Application of Artificial Intelligence Conf. – Dayton, 1990. – 196 с.

11. J. Park, J. I.W. Sandberg. Universal approximation using radial-basis-function networks. Neural Computation, 1991. – 151 с.

12. R.J. Schilling, J.J. Carrol, A.F. Al-Ajlouni. Approximation of nonlinear systems with radial basis function neural networks. IEEE Trans. on Neural Networks, 2001. – 383 с.

13. D.E. Specht. A general regression neural network. IEEE Trans. on Neural Networks, 1991 – 576 с.

14. E.A. Nadaraya. About nonparametric probability density and regression estimates. Probability Theory and its Application, 1965. – 199 с.

15. В. П. Живоглядов, А.В. Медведев. Непараметрические алгоритмы адаптации. Фрунзе: Илим, 1974. – 203 с.

16. Медведев, А.В. Адаптация в условиях непараметрической неопределенности. Адаптивные системы и их приложения. – Новосибирск: Наука, 1978. – 236 с.

17. R.J. Schilling, J.J. Carrol, A.F. Al-Ajlouni. Approximation of nonlinear systems with radial basis function neural networks. IEEE Trans. on Neural Networks, 2001. – 12 с.

18. O. Nelles. Nonlinear System Identification. Berlin: Springer, 2001. – 23 с.

19. Y. Bodyanskiy, N. Teslenko. Generalized regression neuro-fuzzy network. Information Research and Applications i.TECH-2007: Proc. Fifth Int. Conf.-Varna, 2007. – 43 с.

20. Mladen Dalto, Jadranko Matusko, Mario Vasak Deep neural networks for time series prediction with applications in ultra-short-term wind forecasting. – IEEE Internatiol Industrial Technology, Seville, Spain, March 17–19 th, 2015. – 53 с.

21. Martin L¨angkvist, Lars Karlsson, Amy Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. Pattern Recognition Letters, 2014. – 148 с.

22. А. А. Ежов, С. А. Шумский. Нейрокомпьютинг и его применения в экономике и бизнесе – М.: МИФИ, 1998. – 232с.

23. Y. Freund , R. E. Schapire. Experiments with a new boosting algorithm. International Conference on Machine Learning, 1996. – 166с.

24. С. А. Терехов. Гениальные комитеты умных машин. IX Всероссийская научно-техническая конференция «Нейроинформатика-2007»: Лекции по нейроинформатике. Часть 2, М., МИФИ, 2007. – 127 с.

25. M. Dalto. Deep neural networks for time series prediction with applications in ultra-short-term wind forecastin. Spain: IEEE Internatiol Industrial Technology, Seville, March 17–19 th. 2015. – 198 c.

26. T. Hastie, R. Tibshirani, J. Friedman. The Elements of Statistical Learning, 2nd edition. Springer, 2009. – 126 c.

27. Murphy P. M. UCI Repository of machine learning databases // CA : University of California, Department of Information and Computer Science. 1994. URL : http://www.ics.uci.edu/mlearn/MLRepository.html (дата звернення 15.05.2018).

28. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent 26 Vanhoucke, Andrew Rabinovich. Going deeper with convolutions. ArXiv, 2014. – 136c.

29. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens. Rethinking the Inception Architecture for Computer Vision. ArXiv, 2015. – 43c.

30. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition. ArXiv, 2015. – 164c.

31. Keras: The Python Deep Learning library [Електронний ресурс]. – Режим доступу: https://keras.io/.

32. Paolo Galeone Dropout — Метод решения проблемы переобучения в нейронных сетях [Електронний ресурс]. – Режим доступу: https://habr.com/ru/company/wunderfund/blog/330814/.