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 нейромережевих методів аналізу медичних зображень
(тема)

Виконала: студентка 2 курсу, групи ІПЗм-17-1.
спеціальності 121- Інженерія програмного забезпечення
(код і повна назва спеціальності)
Освітньо-наукової програми
Інженерія програмного забезпечення
(повна назва освітньої програми)

Сердюк Д.О.
(прізвище, ініціали)
Керівник проф. Дудар З.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

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-професійна програма Інженерія програмного забезпечення
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентці Сердюк Дар'ї Олексіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження нейромережевих методів аналізу медичних зображень

затверджена наказом по університету від "___" _____2019р. № _____
заповнюється вручну після отримання наказу

2. Термін подання студентом роботи до екзаменаційної комісії

3. Вихідні дані до роботи: *бібліотека для Python, методи прийняття рішень, пояснювальна записка. Використовувати середовище розробки PyCharm та операційну систему Windows або Linux.*

4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз проблемної галузі, постановка задачі, огляд сучасних засобів медичної діагностики, аналіз методів розпізнавання зображень, огляд методів штучного інтелекту, огляд та налаштування штучних нейронних мереж, дослідження засобів покращення існуючих методів.*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *титул, мета роботи, наукові та практичні*

завдання роботи, вирішення проблеми розпізнавання медичних зображень для конкретної задачі, запропонований метод аналізу даних, платформа реалізації, програмна реалізація алгоритму та результати дослідження, висновки

6 Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	проф. Дудар З.В		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка *
1.	Аналіз предметної галузі	19.04.2019	
2.	Огляд існуючих методів	23.04.2019	
3.	Алгоритми штучного інтелекту для аналізу даних та прийняття рішень	3.05.2019	
4.	Підготовка пояснювальної записки	5.06.2019	
5.	Спецчастина	7.06.2019	
6.	Підготовка презентації та доповіді	11.06.2019	
7.	Попередній захист	13.06.2019	
8.	Нормоконтроль, рецензування	14.06.2019	
9.	Занесення диплома в електронний архів	19.06.2019	
10.	Допуск до захисту у зав. кафедри	20.06.2019	
* заповнюється вручну після виконання чергового пункту			

Дата видачі завдання _____ _____2019 р.

Студент _____
(підпис)

Керівник роботи _____ Дудар З.В_____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до атестаційної роботи: 71 с., 29 рис., 1 табл., 11 джерел, 2 додатки.

НЕЙРОМЕРЕЖІ, ШТУЧНИЙ ІНТЕЛЕКТ, РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ, МЕДИЧНІ ЗОБРАЖЕННЯ, АВТОМАТИЗАЦІЯ МЕДИЧНОЇ ДІАГНОСТИКИ, TENSORFLOW, PYTHON

Головним об’єктом дослідження є використання штучного інтелекту, а саме нейромереж для аналізу медичних зображень.

Метою роботи є дослідження роботи нейромереж та їх налаштування для навчання і подальшого розпізнавання уражених ділянок шкіри на злоякісні та доброякісні на основі близької фотографії шкіри людини.

Методи розробки використовують мову програмування Python та допоміжні бібліотеки для обчислень, такі як NumPy, SciPy та бібліотека для машинного навчання і добування та аналізу даних TensorFlow.

NEURAL NETWORKS, ARTIFICIAL INTELLIGENCE, COMPUTER VISION, AUTOMATED MEDICAL DIAGNOSIS, PYTHON, TENSORFLOW

The main objective of the research is appliance of artificial intelligence, particularly artificial neural networks.

The goal of this piece of work is to conduct a research about how neural networks algorithms can be applied to distinguish between malignant and benign formations on the human skin, based on the close-up photographs.

The developed methods are based on Python programming language and additional computing libraries, e.g. NumPy, SciPy, and a machine learning, data analysis and data mining library TensorFlow.

.

ЗМІСТ

ВСТУП

Штучні нейронні мережі – це один з розділів машинного навчання, який був започаткований ще у 1940-х роках. Цей підхід бере за основу біологічні нейронні мережі у тварин, які складаються з нейронів головного та спинного мозку центральної нервової системи. Звичайно, штучні нейронні мережі є набагато спрощеною інтерпретацією біологічних та лише прагнуть до вирішення задач, які наш мозок вирішує кожен день.

Нейроподібні мережі пройшли довгий шлях становлення і розвитку, від повного заперечення можливості їх застосування до втілення в багато сфер діяльності людини. Найперші дослідження у цієї галузі, основні положення теорії діяльності головного мозку і власне математичну модель нейрона, яка включала механізм навчання, були розроблені У. Маккалоком і Ч. Піттсом в 1943 році [1]. Така модель мала назву формального нейрона, та її надалі будуть використовувати інші дослідники штучних нейронних мереж.

Вивчення механізмів функціонування нервової системи людини та взаємодії нейронів є дуже важливим завданням, адже для виконання складніших задач треба ще далі вдосконалювати модель нервової системи. Але ще до сьогоднішнього дня людство не має точної та повної інформації про те, як працює людський мозок.

Нервова система людини побудована з нервових клітин – нейронів, мозок містить приблизно 10^{11} нейронів, які створюють близько 10^{15} синапсів (місце передачі інформації між клітинами). Самі по собі нервові клітини розділяються на декілька видів, мають досить складну структуру. Використання навіть найпростішої моделі нейрона не дозволяє побудувати модель мозку, що наближалася б за своїми глобальними показниками до реального об’єкта

моделювання, тому ми кажемо, що штучні мережі є набагато більш простою структурою за реальні. До того ж, штучні нейронні мережі містять набагато меншу кількість нейронів та відтак і синапсів.

Для збільшення або зменшення активності певного нейрона іншими нейронами існують так звані синапси. Вони переносять величину активності від нейрона-відправника до нейрона-одержувачу. Якщо синапс є збудливим, то величина активності нейрона-відправника збільшує активність нейрона-одержувача. Якщо синапс є гальмівним, то величина активності нейрона-відправника зменшує активність нейрона-одержувача. Синапси відрізняються не тільки за ознакою гальмування або збудження нейрона-одержувача, але також і за сумарним впливу (синаптична потужність). Вихідний сигнал кожного нейрона передається по так званому аксону, який закінчується більш ніж 10000 синапсами, що впливають на інші нейрони.

Існує певна класифікація задач, з найпростіших до найскладніших. Найпростіші завдання, як от вирішення простих рівнянь, роздрукування документів, побудування графіків, можуть бути розв'язані простою комп'ютерною програмою.

Більш складні задачі, пов'язані з простим прогнозуванням або вирішенням більш складних рівнянь, також мають відомі алгоритми вирішення. Нейромережі використовують для задач, для яких дуже побудувати алгоритм вирішення – дуже складна та тривала для людини задача. Приклади таких задач:

– аналіз зображень, в тому числі розпізнавання облич;
– розпізнавання мови;
– складне прогнозування у великих масштабах;
– класифікація інформації в реальному масштабі часу.

Метою даної роботи є дослідження та аналіз застосування нейронних мереж для розпізнавання та подальшої класифікації медичних зображень. Прогнозування

діагнозу на основі аналізів (в тому числі зображень) є наразі реалістичною задачею. Але її виконання потребує досить великих обчислювальних потужностей та вдосконалення існуючих алгоритмів.

Розробка автоматизованих систем медичної діагностики є надзвичайно важливим завданням, адже для постановки діагнозу лікарю необхідно опрацювати величезну кількість інформації та постійно тримати її на увазі. Збільшення цієї інформації призводить до втомленості та можливих розладів психіки у людини, які у свою чергу призводять до помилкових або недостатньо точних аналізів.

Тоді, об’єктом даного дослідження є медичні зображення (знімки тканин або клітин людини). За допомогою використання комп’ютерних технологій у медичній сфері можна збільшити кількість на швидкість поставлених діагнозів, підвищити надійність та точність діагностики, автоматизувати цей процес тощо.

Предмет дослідження – методи та алгоритми реалізації нейронних мереж для розпізнавання зображень.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Загальна інформація про нейромережі

Враховуючи те, що нейромережі є так званим аналогом біологічної нейронної мережі у тварин, то перші застосовують певні принципи роботи других. Найважливішими особливостями діяльності біологічних нейромереж є навчання, узагальнення та абстрагування. Саме ці принципи лежать у основі проектування штучних нейронних мереж.

Однією з найважливіших особливостей штучний нейромереж є паралельна обробка даних, адже нейронна мережа містить величезну кількість міжнейронних зв’язків. Крім того, при великому числі міжнейронних з'єднань мережу набуває стійкість до помилок, які виникають на деяких лініях. Функції пошкоджених зв'язків беруть на себе справні лінії, в результаті чого діяльність мережі майже не зазнає істотних збурень.

Інша не менш важлива властивість – здатність до навчання і узагальнення накопичених знань. Нейронна мережа має риси штучного інтелекту. Натренований на обмеженій множині даних мережу, здатна узагальнювати отриману інформацію і показувати хороші результати на даних, що не використовувалися в процесі навчання. Властивість узагальнення дає можливість нейронній мережі знижувати чутливість до незначних відхилень вхідних сигналів від норми. Ця властивість дуже важлива для об’єктів, які існують у реальному середовищі.

Є два основних підходи для навчання нейронних мереж: з учителем та без. Навчання з учителем означає використання такої тестової вибірки даних, яка містить пари вхідного значення та очікуваного вихідного: наприклад фото заздалегідь визначені фото котів та собак. Далі на основі визначених ознак та

множини класів, для нового вхідного фото нейромережа визначає найбільш приближений для нього клас. Така задача також має назву задачі класифікації.

Навчання без учителя у свою чергу є складнішим процесом, так, при цьому підході, вибірка не містить «правильного» класу, тоді нейромережа повинна самостійно розділити дані тестової вибірки на класи за подібними ознаками. Така задача називається кластеризацією.

Властивість абстрагування дозволяє нейронній мережі створювати нову сутність, виходячи з аналізу вхідної інформації. Особливо ця властивість проявляється для задач розпізнавання образів. Завдяки їй нейромережа може створювати деякий ідеальний образ, керуючись вхідною інформацією, яка має деякі властивості цього образу.

Також нейромережі є дуже стійкими до похибок у роботі. Так, якщо штучна нейронна мережа складається з 1000 нейронів, то похибка у одному або навіть десяти нейронах не призведе до тотальної помилка результату. Для цієї властивості є важливим і той факт, що нейромережа зазвичай не дає відповіді на 100 відсотків, а лише видає процент вірогідності.

Недоліки використання нейромереж:

– нейромережі не можуть дати однозначну та точну відповідь (як і мозок людини);

– неможливість вирішувати обчислювальні задачі;

– неможливість вирішувати задачу поетапно (крок за кроком).

Нейромережі використовуються для виконання ряду різноманітних задач, таких як задачі пов’язані з комп'ютерним баченням (computer vision), розпізнаванням мовлення, грою у відеоігри, машинним перекладом, соціально-мережевим фільтруванням та зокрема медичним діагностуванням.

1.2 Теорія розпізнавання образів

Розпізнавання (класифікація) об’єктів – одна з найважливіших задач в області комп’ютерного зору. Хоча це задача з якою кожна людина справляється автоматично (розпізнавання лиць, тварин, будинків тощо), вона є надзвичайно складною для комп'ютера. Але, враховуючи те, що людина вчиться впізнавати (виділяти) певні предмети з великої групи інших все своє життя, то можна примінити подібний підхід у створенні систем комп'ютерного зору. Розпізнавання може ускладнюватись нечітким зображенням, наявність шумів, нерівним освітленням і т.д. Прикладами таких задач є: розпізнавання облич, розпізнавання рукописного тексту, розпізнавання штрихкодів.

Необхідність систем розпізнавання виникає у багатьох сферах життя: медицині, військовій справі, охоронних системах, системах безпеки.

Власне, процес розпізнавання складається з наступних етапів:

– знаходження тренувального датасету (бажано, щоб він був доволі великим);

– попередня обробка поточного зображення: зменшення шумів, вирівняння контрасту, зміна розміру, сегментація та інші;

– виділення найбільш інформативних ознак для подальшої класифікації;

– розробка правил класифікації;

– власне, навчання системи (нейронна мережа, метод опорних векторів, лінійна регресія);

– налаштування параметрів системи після оцінки якості роботи класифікатора (кількість помилок, чутливість);

– класифікація нового зображення вже з тестового датасету.

1.3 Види нейронних мереж

Роздивимось структуру простішої нейронної мережі. Вона складається з вхідного та вихідного шарів. Дані, які подаються на вхід, обробляються нейронами мережі та передаються на наступний шар. Кожен нейрон має рівень активації, що лежить в діапазоні між максимумом і мінімумом, отже, на відміну від булевої логіки, існує більш ніж два рівня активації. При кожному з цих переходів, вхідне число збільшується або зменшується в залежності від ваги (аналог синапсів у біологічній нейромережі), яка налаштовується під час навчання.

На виході отримаємо результуюче число – відгук мережі. Далі це число подається на вхід функції-активатора, яка перетворює значення на вихід нейрона.

1.3.1 Одношарові мережі

В одношарових нейронних мережах сигнали зі вхідного шару одразу передаються на вихідний (рис. 1.1).

Одношаровий перцептрон здатний розпізнавати найпростіші образи. Окремий нейрон обчислює зважену суму сигналів вхідних елементів, віднімає значення зсуву і пропускає результат через жорстку порогову функцію, вихід якої дорівнює +1 чи -1. В залежності від значення вихідного сигналу приймається рішення: сигнал належить до класу А чи Б.

Перцептрон – одна з перших моделей нейромереж, запропонована Френком Розенблатом у 1957 році [7].

Алгоритм навчання одношарового перцептрона:

– ініціалізація синаптичних ваг і зсуву: синаптичні ваги приймають малі випадкові значення;

– пред'явлення мережі нового вхідного і бажаного вихідного сигналів: вхідний сигнал $x=(x_1, x_2, ..., x_n)$ пред'являється нейрону разом з бажаним вихідним сигналом d.

– обчислення вихідного сигналу нейрона:

$$y(t) = f(\sum_{i=1}^{n} w_i(t)x_i(t) - b)$$

– налаштування значень ваг:

$$w_i(t+1)=w_i(t)+r[d(t)-y(t)]x_i(t),\ i=1, ..., N$$

$$d(t) = \begin{cases} +1, \text{вихідний клас А} \\ -1, \text{вихідний клас Б'} \end{cases}$$

де $w_i(t)$ - вага зв'язку від i-го елемента вхідного сигналу до нейрона в момент часу t,

r - швидкість навчання (менше 1),

$d(t)$ - бажаний вихідний сигнал.

Якщо мережа приймає правильне рішення, синаптичні ваги не модифікуються.

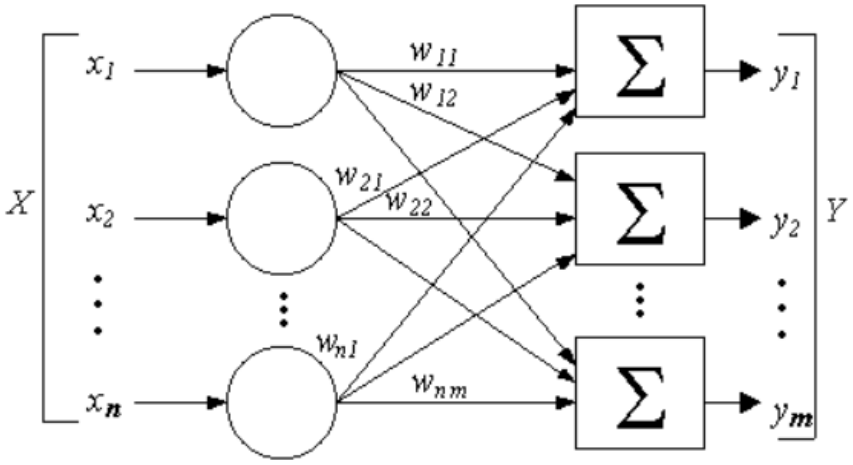

Рисунок 1.1 – Одношарова нейронна мережа

– Перехід до кроку 2.

Розмірності входу і виходу обмежені при програмній реалізації тільки можливостями обчислювальної системи, на якій моделюється нейронна мережа, при апаратній реалізації – технологічними можливостями.

Області застосування таких мереж зазвичай більш прості задачі, як от розпізнавання образів або класифікація.

1.3.2 Багатошарові мережі

Багатошарові мережі (рис. 1.2) володіють значно більшими можливостями, ніж одношарові. На відміну від одношарових, вони містять один або звичайно більше прихованих шарів.

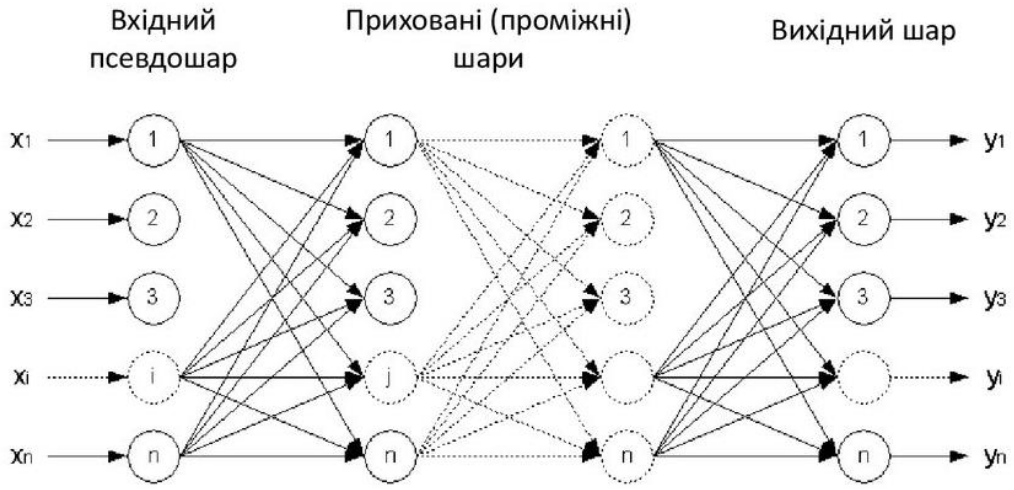

Рисунок 1.2 – Багатошарова нейромережа

Проте багатошарові мережі можуть призвести до збільшення обчислювальної потужності в порівнянні з одношаровими лише в тому випадку, якщо активаційна функція між шарами буде нелінійною.

Багатошарові мережі модна у свою чергу поділити на мережі прямого та зворотного поширення.

Мережі прямого поширення (Feedforward neural network) – штучні нейронні мережі, в яких сигнал поширюється строго від вхідного шару до вихідного. У зворотному напрямку сигнал не поширюється. Такі мережі широко використовуються і цілком успішно вирішують певний клас завдань: прогнозування, кластеризація і розпізнавання.

А в мережах із зворотними зв'язками виходи нейронів можуть повертатися на входи. Це означає, що вихід якогось нейрона визначається не тільки його вагами і вхідним сигналом, але ще і попередніми виходами (так як вони знову повернулися на входи).

1.4 Модель штучного нейрона

Штучний нейрон є спрощеною моделлю біологічного нейрона, складова частина штучної нейронної мережі. Першу математичну модель такого нейрона було представлено у статті Мак-Калока і Піттса [2]. Головним принципом цієї теорії є те, що довільні явища, які стосуються вищої нервової діяльності, можуть бути проаналізовані і зрозумілі, як деяка активність в мережі, що складається з логічних елементів, які приймають тільки два стани «все або нічого».

Це означає, що в найпростіших моделях нейронів вихідний сигнал приймає двійкові значення: 0 або 1. Значення 1 відповідає перевищення порога збудження нейрона, а значення 0 – збудження нижче порогового рівня.

Математично нейрон являє собою зважений суматор (рис. 1.3), єдиний вихід якого визначається через його входи і матрицю ваг наступним чином:

$$y = f(u), y = \sum_{i=0}^{n} w_i x_i + w_0 x_0,$$

де x_i – сигнали на входах нейрона,

w_i – ваги входів.

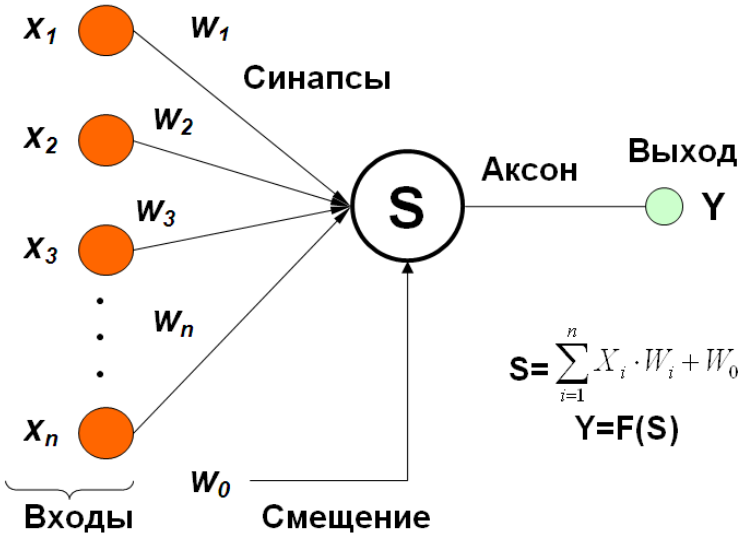

Рисунок 1.3 – Модель штучного нейрону

Функція u називається індукованим локальним полем, а f(u) - передавальної функцією або функцією активації нейрона. Можливі значення сигналів на входах нейрона вважають заданими в інтервалі [0,1]. Вони можуть бути або дискретними (0 або 1), або аналоговими. Додатковий вхід x_0 і відповідна йому вага w_0 використовуються для ініціалізації нейрона. Під ініціалізацією мається на увазі зсув активационной функції нейрона по горизонтальній осі, тобто формування порогу чутливості нейрона. Крім того, іноді до виходу нейрона спеціально додають якусь випадкову величину, яка називається збудження. Збудження можна розглядати як сигнал на додатковому, завжди навантаженому, синапсі.

Передавальна функція нейрона f(u) визначає залежність сигналу на виході нейрона від зваженої суми сигналів на його входах. У більшості випадків вона є монотонно зростаючою і має область значень [0,1], однак існують винятки. Також

для деяких алгоритмів навчання мережі необхідно, щоб вона була безперервно диференційованою на всій числовій осі. Штучний нейрон повністю характеризується його передавальною функцією [5]. Використання різних передавальних функцій дозволяє вносити нелінійність в роботу нейрона і в цілому нейронної мережі.

Найпоширенішою нелінійною аналоговою активаційною функцією є сигмоїдальна логічна функція [6], її область значень знаходить у інтервалі (0, 1) (рис. 1.4). З параметрами, b c і d, функцію задають виразом:

$$y = \frac{b}{c+e^{dv}},$$

де b, c, d – вільні параметри функції.

При одиничних значеннях параметрів b =1, c =1, d = −1 одержуємо:

$$y = \frac{1}{1+e^{-v}}$$

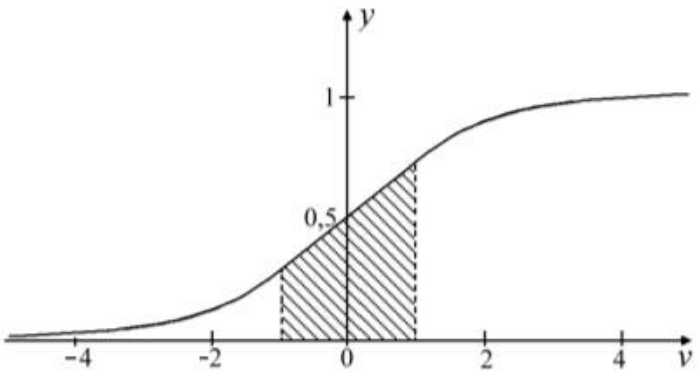

Рисунок 1.4 – Сигмоїдальна логічна функція

Більш простою є лінійна передавальна функція (рис. 1.5), для якої сигнал на виході нейрона лінійно пов'язаний із ваговою сумою сигналів на його вході. Обчислюється за наступною формулою:

$$f(x) = tx,$$

де t — параметр функції.

В штучних нейронних мережах із шаруватою структурою, нейрони з передавальними функціями такого типу, як правило, складають вхідний шар.

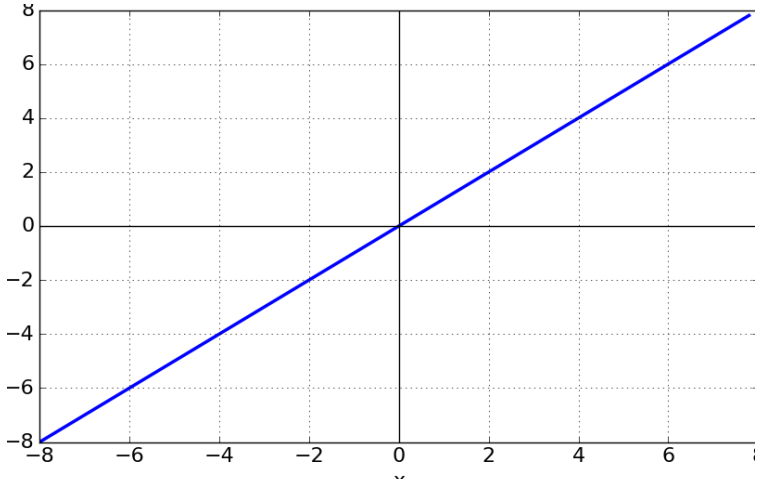

Рисунок 1.5 Лінійна функція

Крім простої лінійної функції можуть використовуватись також її модифікації. Наприклад, напівлінійна функція (якщо її аргумент менше нуля, то вона дорівнює нулю, а в інших випадках, поводить себе як лінійна) або крокова (лінійна функція з насиченням), яку можна виразити формулою:

При цьому можливий зсув функції по обох осях (як зображено на малюнку).

Недоліками крокової і напівлінійних активаційних функцій щодо лінійної можна назвати те, що вони не є диференційованими на всій числовій осі, а отже не можуть бути використані при навчанні за деякими алгоритмам.

Порогова передаточна функція (рис. 1.6) являє собою перепад. До тих пір поки зважений сигнал на вході нейрона не досягає певного рівня T– сигнал на виході дорівнює нулю. Як тільки сигнал на вході нейрона перевищує зазначений рівень, вихідний сигнал стрибкоподібно змінюється на одиницю. Власне, найперший представник багатошарових штучних нейронних мереж – перцептрон [7] складався виключно з нейронів такого типу.

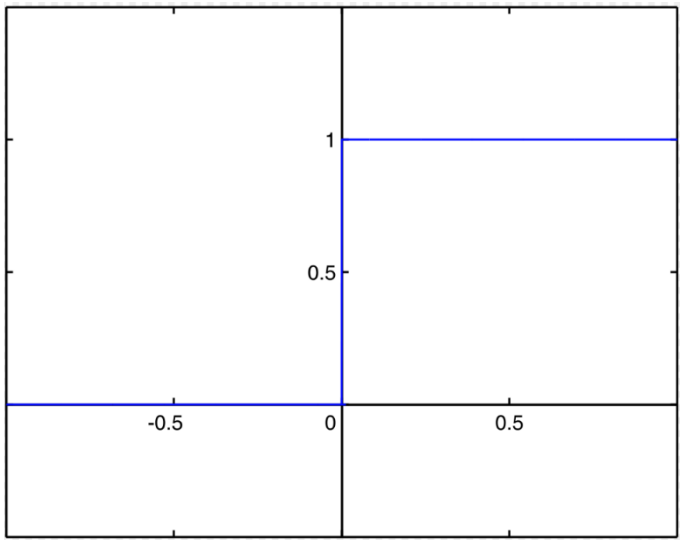

Рисунок 1.6 – Порогова функція активації

Математичний запис цієї функції виглядає так:

$$f(x) = \begin{cases} 1 & if\ x\ \geq T \\ 0 & else \end{cases},$$

де $T=-w_0*x_0$ – зсув функції активації щодо горизонтальної осі,

x – зважена сума сигналів на входах нейрона без урахування цього додатку.

З огляду на те, що ця функція не є диференційованою на всій осі абсцис, її не можна використовувати в мережах, що навчаються за алгоритмом зворотного поширення помилки.

До того ж, порогову функції активації можна використовувати з іншими алгоритмами, які вимагають, щоб передавальної функції була диференційована, як зворотнє поширення помилки.

Іншою часто використовуваною функцією активації є гіперболічний тангенс (рис. 1.7). Ця функціє графіком нагадує сигмоїдальну функцію та має такі самі переваги, область значень (-1, 1).

Варто відзначити, що на відміну від сигмоїдальної фукнції, градієнт гіперболічного тангенса є більшим, тоді вибір між ними двома буде залежати від вимог до амплітуди градієнту.

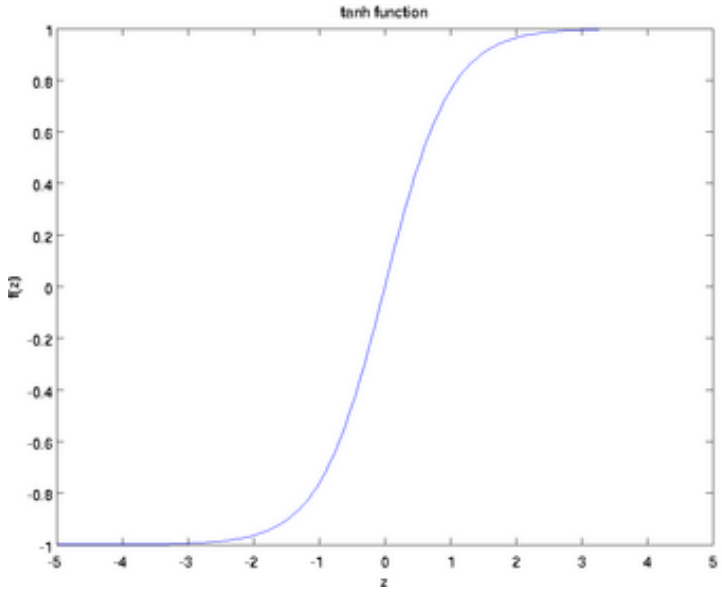

Рисунок 1.7 – Гіперболічний тангенс

Останнім часом також стає популярною функція випрямляч або ReLU – rectified linear unit (рис. 1.8).

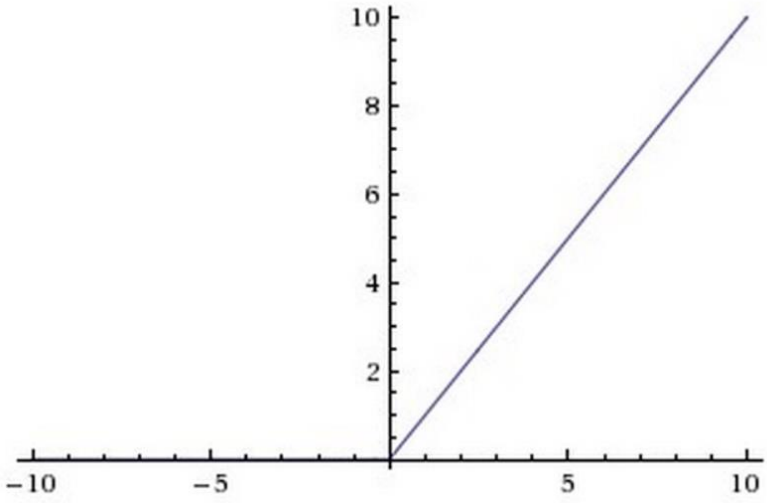

Рисунок 1.8 – Функція-випрямляч

Важливою перевагою цієї функції перед сигмоїдальною та гіперболічним тангенсів, є те, що очислення значень цієї функції не є таким ресурсоємким, тож відбувається швидше за наступною формулою:

$$f(x) = \max(0, x)$$

З недоліків цієї функції активації можна відзначити можливе відмирання великої частини нейронів, тобто внаслідок визначення вагів, деякі нейроні можуть перестати активуватися. Таку проблему можна вирішити встановленням коректної швидкості навчання. Існує також декілька модифікацій цієї функції: параметрична рандомізована та нещільна, кожна з них застосовується для певного виду задач.

Нейрон зміщення (bias нейрон) – це вид нейронів, що використовується в більшості нейромереж. Особливість цього типу нейронів полягає в тому, що його вхід і вихід завжди дорівнюють 1 і він ніколи не має вхідних синапсів. Нейрони зміщення можуть, або бути присутнім в нейронної мережі по одному на шар, або бути повністю відсутніми. З'єднання у нейронів зміщення такі ж, як у звичайних нейронів – з усіма нейронами наступного рівня, за винятком того, що синапсів між двома нейронами зміщення бути не може.

Отже, їх можна розміщувати на вхідному шарі і всіх прихованих шарах, але ніяк не на вихідному шарі, так як їм просто нема з чим буде формувати зв'язок. По суті, нейрон зміщення дозволяє класифікатору зміщувати межу рішення ліворуч або праворуч.

1.5 Конволюційні нейронні мережі

Конволюційні, або згорткові, нейронні мережі є модифікацією багатошарових перцептронів, які успішно застосовуються для роботи з зображенням: класифікації, кластеризації та проведення ідентифікації об’єктів на зображенні. Згорткові мережі були спроектовані також на основі мозку тварини, а саме процесах, які відбуваються у нейронах зорової кори мозку.

Інновацією згорткових нейронних мереж є можливість автоматичного паралельного навчання великої кількості фільтрів, специфічних для навчальних даних за обмежень конкретної задачі прогнозування, наприклад, класифікації зображень. Результатом є дуже специфічні ознаки, які можуть бути виявлені в будь-якому місці на вхідних зображеннях.

Тобто, мета конволюційної мережі полягає у зменшенні зображення у форму, яку легше обробляти, не втрачаючи особливості, які є критичними для отримання хорошого прогнозу. До того ж, незважаючи на великий розмір, ці мережі мають невелику кількість параметрів, що настроюються.

Конволюційні нейронні мережі мають відмінну від звичайних нейронних мереж архітектуру [2]. Регулярні нейронні мережі перетворюють вхід, провевши його через ряд прихованих шарів. Кожен шар складається з набору нейронів, де кожен шар досі повністю з'єднаний з усіма нейронами шару – повноз’єдані шари. Нарешті, є останній повноз’єднаний шар – вихідний шар – який представляє прогнози.

Тобто, для обробки зображень, на вхідному шарі повинно бути стільки нейронів, скільки є пікселів у зображенні, тому у випадку великих зображень такі мережі дуже погано масштабуються. Так, якщо розмір зображення 20 х 20 пікселів та містить 3 канали кольору, у першому прихованому шарі нейрон буде мати вагу 20*20*3 = 1200. До того ж, велика кількість згенерованих мережею параметрів може швидко призвести до перенавчання.

Очевидно, що багатошаровий персептрон не є найкращою ідеєю для обробки зображень. Однією з головних проблем є те, що просторова інформація втрачається, коли зображення розгортається в одновимірний масив в персептроні. Тому, якщо бажаний об'єкт для ідентифікації знаходиться в різних вуглах на різних зображеннях, або якщо зображення перевернуте догори ногами, багатошаровий персептрон не буде корректно розпізнавати об’єкт. Вузли, які знаходяться близько,

важливі, тому що вони допомагають визначити особливості зображення (ознаки). Таким чином, нам потрібен спосіб використання просторової кореляції особливостей зображення (пікселів) таким чином, щоб ми могли бачити об’єкт на знімку, незалежно від того, де він може з'явитися.

Перш за все, шари в конволюційних мережах організовані в 3 виміри: ширина, висота і глибина. Далі, нейрони в одному шарі не з'єднуються з усіма нейронами в наступному шарі, але тільки з невеликою його областю. Нарешті, кінцевий прогноз буде зведений до одного вектора оцінок ймовірностей, організованих по глибинному вимірі.

Згорткові мережі також мають вхідний, вихідний та приховані шари. Але саме першим прихований шар – шар згортки – це те, що робить цю мережу особливою (рис. 1.9).

Згортка – це просте застосування фільтра до входу, що призводить до активації. Повторне застосування одного і того ж фільтра до входу призводить до карти активацій, які називаються картою об'єктів, що вказує на розташування і силу виявленої функції на вході, наприклад зображення.

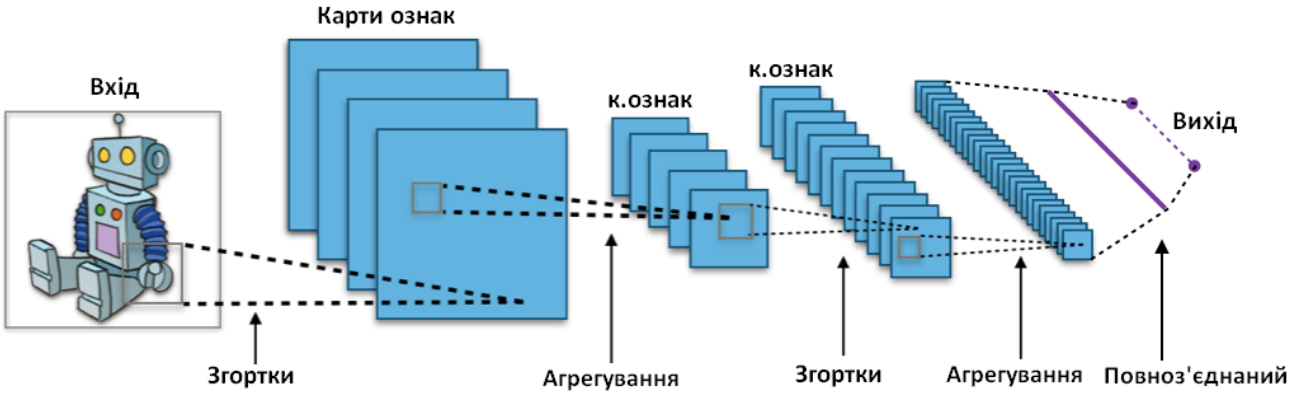

Рисунок 1.9 – Архітектура згорткової нейронної мережі

На згортковому шарі кожен фільтр здійснює згортку за шириною та висотою вхідної ємності, обчислюючи скалярний добуток даних фільтру та входу, і формуючи 2-вимірну карту збудження (карту ознак) цього фільтру. Усі карти збудження формують повну ємність виходу згорткового шару.

Після шару згортки зазвичай наступним буде шар агрегування (pooling layer). Цей шар відповідає за зменшення просторового розміру знайдених на етапі згортки знак. Мета в тому щоб зменшити обчислювальну потужність, необхідну для обробки даних, шляхом зменшення розмірності. Крім того, це корисно для вилучення домінуючих ознак, які є ротаційними та позиційними інваріантами, таким чином підтримуючи процес ефективної підготовки моделі.

Агрегувальний шар базується на тому, що грубе положення ознаки відносно інших є важливішим, ніж її точне положення на зображенні, тому можна зменшувати просторовий розмір ознак для покращення продуктивності. Таким чином, агрегувальний шар можна вставляти поміж послідовних шарів згортки.

Виділяють декілька видів агрегування: максимізаційне (max pooling), усередньювальне (average pooling) та L^2-нормове агрегування. Максимізаційне агрегування залишає максимальне значення пікселя з групи (рис. 1.10), усередньювальне – відповідно середнє значення пікселів групи.

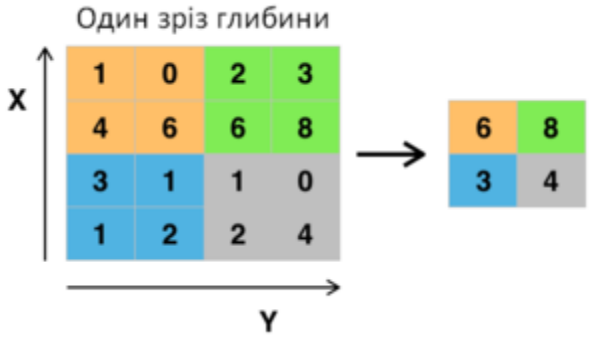

Рисунок 1.10 – Максимізаційне агрегування з фільтром 2x2 та кроком 2

При L^2-нормовому агрегуванні значення обчислюється за формулою норми L2 або Евклідової норми:

$$\|\bar{x}\| = \sqrt{\sum_{i=0}^{n} {x_i}^2},$$

де вектор $\bar{x}$ задається як $\bar{x} = (x_1, \dots, x_n)$.

Таким чином, гіперпараметрами цього шару (якщо він використовується у мережі, що розробляється, є тип фільтру (хоча, зазвичай, найбільш ефективним є максимізаційний [8]), розмір ядра фільтру та значення кроку.

Однак, багато розробників не полюбляють операцію агрегації і думають, що можна обійтися і без неї на користь архітектури, яка складається тільки з повторюваних шарів згортки. Щоб зменшити розмір подання, вони пропонують використовувати більший шаг в шарах згортки час від часу. Також виявлено, що відкидання шарів агрегації є важливим у навчанні хороших генеративних моделей, таких як варіаційні автокодери (VAE) або генеративні змагальні мережі (GAN). Ймовірно, що майбутні архітектури будуть мати лише небагато шарів агрегації або не використовувати їх взагалі.

Після шару активації може знаходитись шар нормалізації, який використовує одну з функцій, описаних у попередньому підрозділі. Але, з часом шар нормалізації перестають включати до мережі, бо частіше його внесок є дуже малим, або навіть нульовим.

Останнім шаром є повноз’єданий шар, який здійснює високорівневі судження у мережі. Нейрони у цьому шарі мають з'єднання з усіма збудженнями попереднього шару, як це можна бачити у звичайних нейронних мережах.

1.6 Алгоритми навчання нейронних мереж

Навчання нейромережі є власне основним процесом її розробки. У цьому розділі буде розглянуто методи найбільш ефективного навчання та оптимізації мережі.

1.6.1 Еволюційний метод

Цей метод базується на механізмах спадкування ознак у природних популяціях організмів. Гіпотеза селекції може добре бути застосована до мережі штучних нейронів. Чим вище пристосованість особини, тим вище ймовірність, що в її потомстві ознаки, які визначають пристосованість, будуть виражені ще більше.

Для визначення ефективності проведеного навчання використовуються так звана функція втрати. Якщо алгоритм працює добре, тобто, після навчанні з вчителелем на тестових даних прогнозування є коректним, значення функції є малим. При зміні частин алгоритма саме за допомогою функції втрати можна стежити за ефективністю алгоритму. Тоді, навчання мережі є задачею оптимізації, метою якої є зменшення значення функції втрати.

Одна з найпростіших функцій втрати – середньоквадратична похибка (MSE – Mean Squared Error). Задається наступною формулою:

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2,$$

де n – кількість тестових даних,

y – значення, прогнозоване алгоритмом,

$\hat{y}$ – реальне значення.

Частіше для обчислення втрат використовується функція перехресної ентропії (cross entropy, log loss), яка є логарифмічною модифікацією функції правдоподібності та для випадку бінарної класифікації має наступну формулу:

$$H = -(y * \log(p) + (1 - y)\log(1 - p)),$$

де p – значення, прогнозоване алгоритмом,

y – реальне значення.

Функцію можна записати простіше наступним чином:

$$H = -\begin{cases} \log(p), & y = 1 \\ \log(1-p), & y = 0 \end{cases}$$

Функція перехресної ентропії використовується для вихідних значень у діапазоні від 0 до 1. Великою перевагою цієї функції є те, що за малу помилку (0,1-0,2) штраф функціх буде малим, а за велику похибку (0,9-1,0) – дуже великим (рис. 1.11).

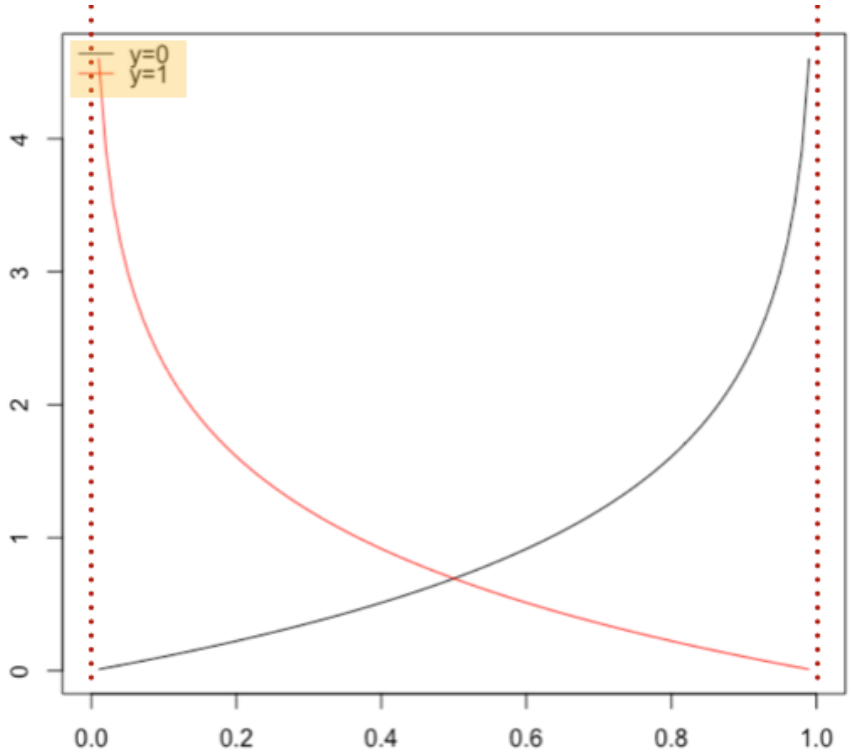

Рисунок 1.11 – Графік функції перехрестної ентропії

1.6.2 Метод градієнтного спуску

Градієнтний спуск – один із методів алгоритм навчання, застосовується у банатьох моделях машинного навчання, зокрема при навчанні глибоких нейронних мереж, також відомий як метод зворотного поширення помилки (backpropagation).

Це алгоритм оптимізації, заснований на опуклої функції, який ітераційно змінює параметри, щоб мінімізувати задану функцію до її локального мінімуму.

Сутність методу градієнтного спуску – пошук мінімального значення функції втрати для визначення оптимальних параметрів створюваної моделі (ваги та зсуви), використовуючи градієнт функції втрати [9]. Враховуючи, що похідна функції відносного однієї ваги – це те, куди потрібно змістити вагу, щоб мінімізувати її для цього прикладу навчання, то власне, градієнт допомагає визначити вплив кожної ваги з'єднань нейронної мережі на значення функції втрати.

Щоб почати пошук правильних значень, ми ініціалізуємо значення W (ваги) і B (зсув) деякими випадковими числами, після чого градієнтний спуск починається з цієї точки (десь у верхній частині на рис. 1.12). Потім він займає один крок за іншим у найкрутішому напрямку вниз (наприклад, зверху вниз), поки він не досягне точки, де функція витрат J є якомога меншою.

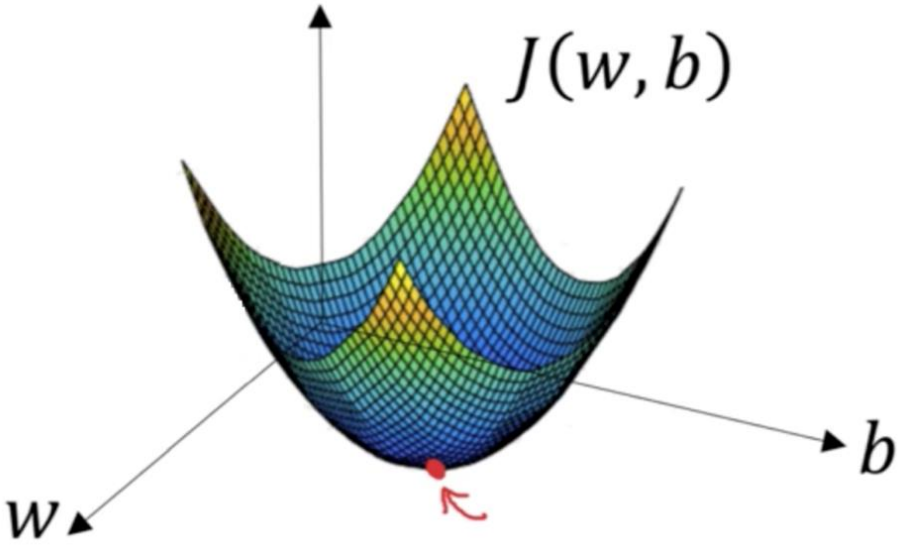

Рисунок 1.12 – Значення функції, до якого прагне градієнтний спуск

Наскільки великими є крок, на який кожний раз сходить алгоритм в бік місцевого мінімуму, визначається так званий коефіцієнт швидкості навчання. Він визначає, наскільки швидко або повільно ми рухатимемося до оптимальних ваг.

Для того, щоб спуск градієнта досягав локального мінімуму, ми повинні встановити швидкість навчання до певного значення. Адже якщо цей коефіцієнт занадто великий, ми подолаємо мінімум: тобто пройдемо повз мінімуму. Якщо занадто малий, то буде виконано надто багато ітерацій, щоб дістатися до мінімуму. Тобто, необхідно емпіричним методом на конкретних даних встановити значення таким чином, що воно не буде занадто низьким чи занадто високим.

Часто використовуються різноманітні модифікації методу градієнтного спуску, один із них – стохастичний градієнтний спуск (SGD – Stochastic Gradient Descend). Цю модифікацію часто використовують, коли тренувальний сет містить дуже багато даних. Головна його особливість у порівнянні з вище описаним алгоритмом є використання одного випадкового тренувального даного, а не суми градієнтів, викликаних кожним елементом навчання. Тоді стандартному градієнтному спуску потрібен один прохід по тренувальним даним до того, як він зможе змінювати параметри, а при стохастичному – параметри моделі змінюються після кожного об'єкта навчання.

Тому для великих масивів даних стохастический градієнтний спуск може дати значну перевагу в швидкості в порівнянні зі стандартним градієнтним

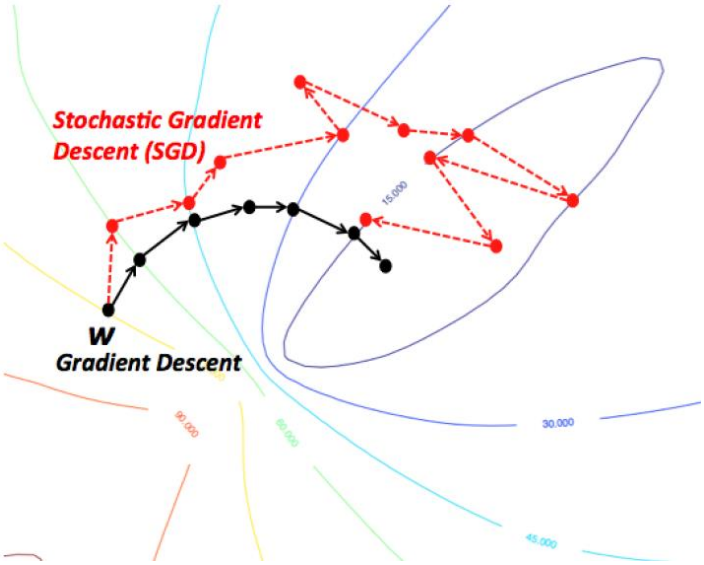

Рисунок 1.13 – Криві знаходження мінімуму функції

спуском. Однак, при стохастичному варіанті лінія спуску буде більш переривчастою внаслідок вибору випадкових даних (рис. 1.13).

Частіше використовується певну комбінацію обох методів, який містить їх переваги, стохастичний градієнтний спуск маленькими партіями (mini-batch). Він полягає в обчисленні градієнта проти більш ніж одного прикладу навчання (власне, партії) на кожному кроці. Цей метод може працювати значно краще, ніж справжній стохастичний градієнтний спуск, оскільки код може використовувати бібліотеки векторизації, а не обчислювати кожен крок окремо.

1.6.3 Метод зворотного поширення помилки

Зворотне поширення, або зворотне поширення помилок – алгоритм навчання штучних нейронних мереж з використанням градієнтного спуску [10]. Враховуючи штучну нейронну мережу та функцію втрат, метод обчислює градієнт функції втрат щодо ваг нейронної мережі. Це узагальнення правила дельти для персептронів до багатошарових нейронних мереж.

Нейронна мережа поширює сигнал вхідних даних вперед (feed-forward propagation) через свої параметри до моменту прийняття рішення, а потім розповсюджує інформацію про помилку у зворотньому напрямі через мережу, для зміни одного параметру за раз.

Функція зворотного розповсюдження приймає помилку, пов'язану з неправильним передбаченням нейронною мережею, і використовує цю помилку, щоб налаштувати параметри нейронної мережі в напрямку меншої помилки, а саме фактичний вихід мережі віднімається з бажаного, в результаті чого формується

сигнал помилки. Цей сигнал згодом поширюється мережею в напрямку, протилежному напрямку синаптичних зв'язків, звідси і назва.

Алгоритм зворотного поширення помилки наступний:

а) ініціалізувати синаптичні ваги маленькими випадковими значеннями;

б) вибрати чергову навчальну пару з навчальної множини; подати вхідний вектор на вхід мережі;

в) обчислити вихід мережі;

г) обчислити різницю між виходом мережі і необхідним виходом (цільовим вектором навчальної пари);

д) підкоригувати ваги мережі для мінімізації помилки;

е) повторювати кроки з б) по д) для кожного вектора навчальної множини до тих пір, поки помилка на всій множині не досягне прийнятного рівня.

Корекція вагів виходить нейронів за формулою:

$$\Delta w_{j,i}(n) = -\eta \frac{\partial E_{av}}{\partial w_{j,i}},$$

де $w_{j,i}$ – вага і-го зв’язку j-го нейрону,

η – параметр швидкості навчання,

E – значення вихідної помилки нейромережі.

З огляду на те, що вихідна сума j-го нейрона дорівнює:

$$S_j = \sum_{i=1}^{n} w_{ij} x_i,$$

можна показати, що:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial S_j} \frac{\partial S_j}{\partial w_{ij}} = x_i \frac{\partial E}{\partial S_j}$$

З цього виразу випливає, що диференціал ∂S_j функції активації мережі f(s) повинен існувати і не бути рівним нулю в будь-якій точці, тобто функція активації повинна бути диференційована в кожній точці числової осі. Тому для застосування

методу зворотного поширення використовують сигмоїдальні активаційні функції, такі як логістична або гіперболічний тангенс.

На практиці, навчання продовжують не до точної настройки мережі на мінімум функції помилки, а до тих пір, поки не буде досягнуто досить точне його наближення. Це дозволить з одного боку, зменшити число ітерацій навчання, а з іншого – уникнути перенавчання мережі.

1.6.4 Перенавчання та недонавчання

Метою хорошої моделі машинного навчання є узагальнення від навчальних даних до будь-яких даних з проблемної області. Це дозволяє робити прогнози в майбутньому щодо даних, яких модель ніколи не бачила. Існує термінологія, яка використовується в машинному навчанні, коли говорять про те, наскільки добре модель машинного навчання вивчає і узагальнює нові дані, а саме так звані перенавчання (overfitting) і недонавчання (underfitting) – вони є двома найважливішими причинами для поганої роботи алгоритмів машинного навчання (рис. 1.14).

Рисунок 1.14 – Різні ступені навчання моделі

Недонавчання відбувається, коли все ще є можливості для поліпшення результатів тестового прогону. Це може статися з ряду причин: якщо модель недостатньо потужна, вона надмірно регулюється або просто недостатньо підготовлена. Це означає, що мережа не вивчила відповідних шаблонів у навчальних даних.

Якщо тренувати мережу занадто довго, вона почне переповнювати і вивчатиме шаблони з навчальних даних, які не підходять до тестових даних. Це означає, що шум або випадкові флуктуації в навчальних даних підібрані і засвоєні за допомогою моделі. Проблема полягає в тому, що ці поняття не поширюються на нові дані і негативно впливають на здатність моделей до узагальнення.

Тобто, при навчанні потрібно досягти ідеального балансу. З часом навчання алгоритму, помилка для моделі на тренувальних даних знижується, так само як і помилка на тестовому наборі даних. Якщо навчання проходить занадто довго, продуктивність тренувального набору продовжує зменшуватися, оскільки модель перенавчається і вивчає нерелевантні деталі та шум у навчальному наборі даних. У той же час помилка для тестового набору знову починає зростати, оскільки здатність моделі узагальнюватися зменшується.

Ідеальний баланс є власне точкою перед тим, як помилка на тестовому наборі даних починає збільшуватися, тобто коли модель має хороші навички як для навчального набору даних, так і для невидимого раніше тестового.

Найпростішим способом запобігання перенавчання є зменшення розміру моделі, тобто кількості змінюючихся параметрів (яка визначається кількістю шарів і кількістю нейронів на шар). У процесі глибокого навчання кількість ознак, які можна дізнатися в моделі, часто називають «розмірністю» моделі. Інтуїтивно, модель з більшою кількістю параметрів буде мати більше "можливостей запам'ятовування" і, отже, зможе легко вивчити зв’язок між навчальними зразками

та їхніми цілями, зв’язок без будь-якої здатності до узагальнення, але це буде недоцільно для прогнозів на нових даних.

Потужною запобіжною мірою проти перенавчання є для запобігання перенавчання є крос-валідація. Ідея у тому, щоб генерувати з тренувальних даних декілька невеликих сетів даних та використовувати ці сети для налаштування моделі.

У стандартній крос-валідації k-fold дані розділяють на k підвибірки, на яких потім ітеративно тренуємо алгоритм на k-1 підвибірках, використовуючи останній у якості тестового.

1.7 Аналіз існуючих засобів медичної діагностики

Медична діагностика включає в себе засоби та методи для допомоги у винесенні медичного діагнозу або ж повного встановлення діагнозу на основі певних аналізів. Діагностика базується на величезній кількості даних: анамнезі (інформація, отримана у результаті опитування пацієнта або його оточуючих), безпосередньо багатьох дослідженнях стану організму, лабораторних аналізах, рентгені, біопсії та інше.

Розробка автоматизованих систем медичної діагностики є надзвичайно важливим завданням, адже для постановки діагнозу лікарю необхідно опрацювати величезну кількість інформації та постійно тримати її на увазі. Збільшення цієї інформації призводить до втомленості та можливих розладів психіки у людини, які у свою чергу призводять до помилкових або недостатньо точних аналізів. Згідно підрахунків Національної академії наук, інженерії та медицини (NASEM) [3]

близько 10% пацієнтів помирають внаслідок неправильно поставленого діагнозу. З цим завданням може дуже добре впоратися комп'ютерне програмне забезпечення.

Перші прості автоматизовані системи працюють за певним алгоритмом, побудованим лікарями. Так звані медичні алгоритми (рис. 1.15) застосовуються для величезного ряду задач: від вирахування індексу маси тіла до прогнозування клінічних результатів. Найчастіше такі алгоритми представлені у виді блок-схеми, та відтак можуть бути запрограмовані. На вхід таким алгоритмам подаються готові та зазвичай текстові результати аналізів та анамнезу.

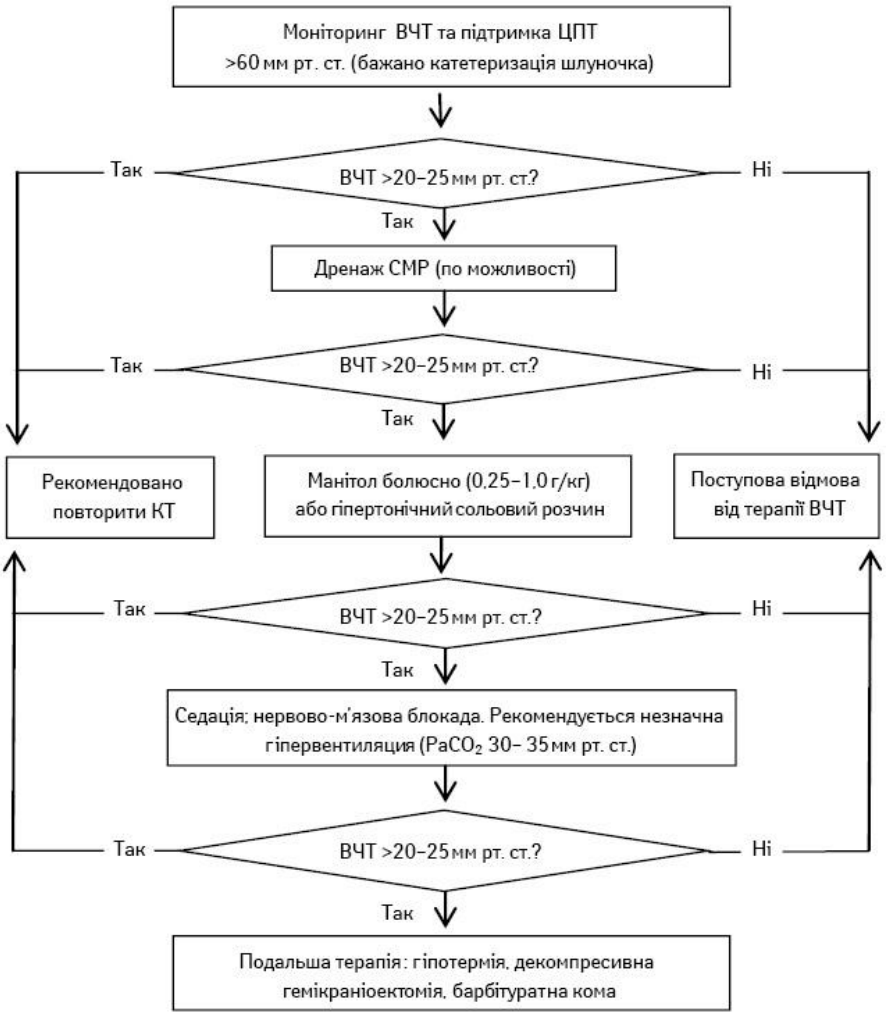

Рисунок 1.15 – Приклад медичного алгоритму

Більш вдосконалене обладнання використовує методи штучного інтелекту (рис. 1.16). Найголовнішою перевагою таких алгоритмів буде здатність

до навчання. Але треба враховувати, що для навчаннях таких програм необхідна досить велика база аналізів та очікуваних результатів. Багато медицинських клінік самостійно розробляють таке програмне забезпечення для подальшого його використання у своїх цілях.

Використання систем штучного інтелекту пов’язане зі зменшенням вартості лікування для населення, покращення точності постановки діагнозу, та більш рання постановка діагнозу, власне прогнозування, завдяки якому можна попередити певні хвороби. З розвитком технологій розпізнавання мови та технологій комп’ютерних помічників (питально-відповідальних систем) анамнез також можна буде проводити повністю автоматизовано.

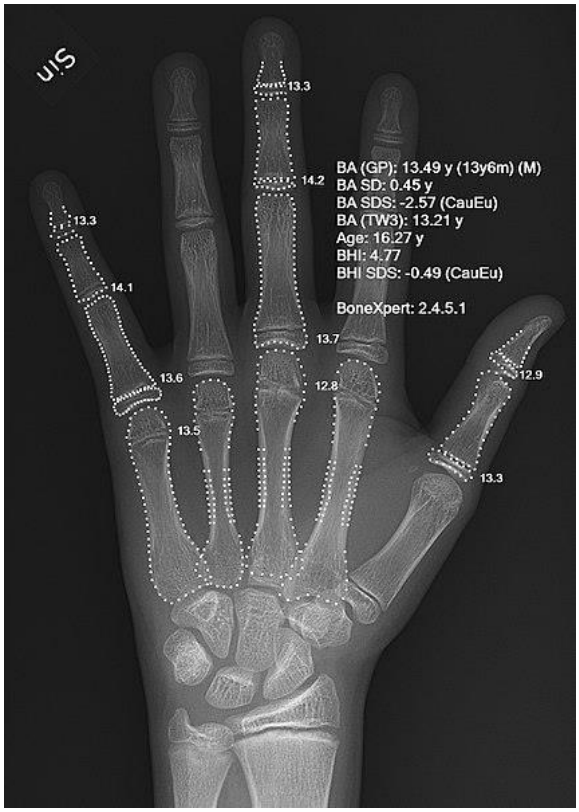

Рисунок 1.16 – Визначення віку кістки людини на основі рентгенівського знімку

Області, для яких використовується комп’ютерне устаткування:

– додатки і програмні продукти для розпізнавання медичних зображень: знімків МРТ, кардіограм, результатів комп'ютерної томографії або УЗД;

– стартапи для вивчення ефективності препаратів, пошуку ефективних вакцин;

– проектування та розробка штучних протезів з урахуванням анатомічних особливостей людини;

– додатки для віддаленої допомоги пацієнту, для надання рекомендацій для лікування простудних хвороб або інших станів, що не загрожують життю, у віддаленому режимі;

– стартапи з лікування ракових захворювань (наприклад, SOPHiA AI - додаток по діагностиці раку, який привернув 30 млн.дол. інвестицій, яке вміє аналізувати клінічну картину стану пацієнта і пропонувати ефективну схему лікування)

1.8 Вибір датасету

У якості датасету була вибрана база Міжнародної спілки зображень шкіри (ISIC – International Skin Imaging Collaboration) [4]. Проект «Меланома» у цій організації – це наукове і галузеве партнерство, спрямоване на полегшення застосування цифрових зображень шкіри для зменшення смертності від меланоми. При діагностуванні і лікуванні на ранніх стадіях меланома легко виліковується.

Цей архів містить близько 23 тисячі зображень класифікованих уражень шкіри, які були зібрані у провідних клінічних центрах по всьому світу і скановані з різних пристроїв у кожному центрі. Зображення знаходяться у відкритому доступі та можуть бути завантажені у цілях навчання. Датасет містить як злоякісні, так і доброякісні приклади.

Незважаючи на те, що ураження шкіри видно неозброєним оком, меланоми на ранній стадії можуть бути важко відрізнити від доброякісних уражень шкіри з подібними виступами. Це призвело до багатьох пропущених меланом, незважаючи на епідемію біопсій шкіри. Кількість непотрібних біопсій залежить від клінічних умов, досвіду експерта та застосування технології. Наприклад, у дітей, у яких частота меланоми низька, а зміна родимок є поширеною, на рік існує понад 500 000 біопсій для діагностики приблизно 400 меланом.

Прийняття технологічних засобів для виявлення меланоми дерматологами в загальній практиці та лікарями первинної медичної допомоги - які становлять лінію фронту у виявленні меланоми - повільне через вартість та незручності. Це змінюється внаслідок мобільної революції, оскільки на професійний ринок виходять цифрові фотоапарати на базі смартфонів і дерматоскопи.

У кожному прикладі міститься зображення ураження, інформацію щодо ураження (включаючи класифікацію і сегментацію) і метадані щодо пацієнта.

Зображення можна завантажити у форматі JPG, разом з метаданими у форматі JSON. Приклад зображень та предоставлених метаданих на рис. 1.17, 1.18.

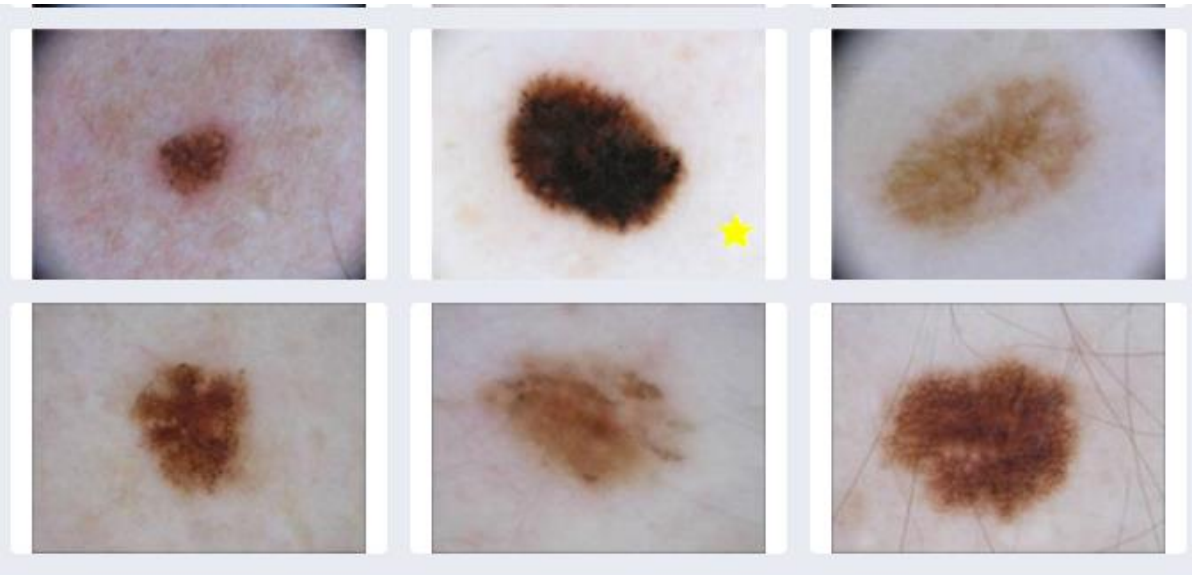

Рисунок 1.17 – Приклад зображень датасету

Крім прямого завантаження з сайту спілки, також предоставлений скрипт на Python, за допомогою якого можна роботи автоматизоване завантаження через API.

Усі зображення мають однакові розміри: 1022 на 767 пікселів. Властивість meta.clinical.benign_malignant позначає на те, чи є утворення доброякісним (benign) або злоякісним (malignant).


```
{
  "_id":"5436e3acbae478396759f0d9",
  "name":"ISIC_0000005",
  "meta":{
    "acquisition":{
      "image_type":"dermoscopic",
      "pixelsX":1022,
      "pixelsY":767
    },
    "clinical":{
      "benign_malignant":"benign",
      "age_approx":40,
      "sex":"female",
      "diagnosis":"nevus",
      "diagnosis_confirm_type":null,
      "anatom_site_general":"anterior torso",
      "melanocytic":true
    }
  }
```


Рисунок 1.18 – Приклад метаданих зображення

В данному досліджені ми не будемо використовувати такі параметри, як стать (‘sex’), вік (‘age_approx’) та заключний діагноз (‘diagnosis confirm type’), так як цей параметр для більшості даних не є інформативним або має значення null. Тобто, для визначення, чи є утворення злоякісним, будемо використовувати, власне, зображення, у якому нейронна мережа буде шукати певні ознаки.

1.9 Постановка задачі

Метою даної атестаційної роботи є створення нейромережевої системи для класифікації фотографій шкірних утворень: є утворення доброякісним чи злоякісним. Під час створення нейронної мережі необхідно дослідити вплив параметрів моделі на кінцеву ефективність роботи мережі на тренувальних та тестових даних.

У якості тренувальних та тестових даних необхідно використовувати дані Міжнародної спілки зображень. Для отримання найкращих результатів роботи нейронної мережі, треба брати до уваги необхідність попередньої обробки зображень та правильно розподілити дані між тренувальним та тестовим сетом, тобто з рівним розподілом добро- та злоякісних утворень.

У результаті необхідно отримати таку архітектуру нейромережі, яка буде найкраще пристосована до вибраних даних: матиме найвищі показники ефективності, найменший час тренування, займати найменше операційної пам’яті.

2 РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ

2.1 Формування вимог до програмної системи

У ході формування вимог був проведений ретельний аналіз поставленої задачі, зокрема можливості програмних засобів та можливі обмеження, вирішення яких не входить до обсягів цього дослідження.

Отже, розроблена система повинная бути представлена у вигляді бібліотеки (модулю) з застосуванням мови програмування Python та інших додаткових бібліотек машинного навчання.

Вимоги до бібліотеки:

– повинна бути гнучкою та вільною до сторонніх налаштувань;

– надавати можливість перегляду результатів пронозування, тобто, наприклад, з якою ймовірністю (у процентах) родимка є злоякісною;

– нейронна мережа у ядрі біблиотеки повинна показувати точність не менше 70% на тестових даних;

– мінімізувати процент хибно позитивних прогнозів, тобто коли злоякісна родимка визначається системою як доброякісна;

Обмеження при розробці:

– система приймає на вхід зображення якими вони надаються на вибранному ресурсі, лише з мінімальною попередньою обробкою, адже проблематика комп'ютерного зору не є центральною для данного дослідження;

– для покращення коректності прогнозування, можна використовувати лише параметри нейронної мережі, та не використовувати високорівневі засоби комп'ютерного зору;

– нейронна мережа повинна працювати на комп'ютері з 6 гігабайтами операційної пам'яті, потужністю процесора 2.5 гігагерц (Intel i5);

– нейронна мережа навчається з учителем, наявна лише задача класифікації.

Для того, щоб система задовольняла поставленим вимогам, необхідно після первісної розробки системи провести дослідження з приводу налаштування параметрів нейронної мережі на роботу з даним датасетом.

Параметри, які необхідно підбирати експериментальним шляхом, називаються гіперпараметрами. У даному контексті вони включають в себе: кількість прихованих шарів мережі та кількість нейронів на кожному з них, швидкість навчання, співвідношення тренувальної дати до тестової, вид навчання мережі, функція активації, необхідність використання нейронів зміщення.

2.2 Огляд засобів програмного забезпечення

TensorFlow – бібліотека з засобами машинного навчання, яка розроблена компанією Google та використовується у тому числі самою компанією для розробки та проведення досліджень. Забезпечує програмний інтерфейс для Python, Java, C++ та інших.

Keras – відкрита бібліотека, яка працює на мові Python, надає високорівневий API для роботи зі штучними нейронними мережами. За допомогою цієї бібліотеки можна конструювати власні нейромережі з різноманітних елементів: оптимізаторів, цільових та передавальних функцій, шарів. Хоча бібліотека може функціонувати як розширення для TensorFlow або Theano, з 2016 після того, як вона була придбана компанією Google, і стала адаптованою як основне надлаштування над TensorFlow.

Keras надає можливість легко і швидко робити прототипування (через повну модульність, мінімалізм і розширюваність). Він підтримує як згорткові мережі, так і рекурентні мережі, а також комбінації цих двох.

Бібліотека Torch є науковою обчислювальною базою з широкою підтримкою алгоритмів машинного навчання, що в першу чергу базується на використанні GPU. Вона пропонує простий у використанні та ефективний інтерфейс своїм користувачам, завдяки легкій та швидкій скриптовій мові, LuaJIT, та лежачій в основі C / CUDA. Користувачі Torch зможуть скористатися основними функціями, такими як потужний N-мірний масив, багато процедур для індексації, нарізки, транспонування та іншими.

Theano – це бібліотека Python, яка дозволяє користувачам встановлювати, оптимізувати і оцінювати математичні вирази, особливо ті, що використовують багатовимірний масив. Можна досягти швидкості, що конкурує з ручними C реалізаціями для проблем, пов'язаних з великими обсягами даних за допомогою Theano. Вона також може перевершити продуктивність C на CPU у декілька разів, використовуючи переваги новітніх GPU. Theano поєднує аспекти системи комп'ютерної алгебри з аспектами оптимізуючого компілятора. Вона також може генерувати індивідуальний код C для багатьох математичних операцій.

Python – широко використовувана мова програмування високого рівня для програмування загального призначення. Крім мови програмування з відкритим вихідним кодом, Python є чудовою об'єктно-орієнтованою, інтерпретованою та інтерактивною мовою програмування. Python поєднує в собі чудову потужність з дуже чітким синтаксисом.

Що робить цю мову програмування надзвичайно привабливою для розробки застосування глибинного навчання, це перш за все дуже простий синтаксис (у порівнянні з Java або C++), що сприяє низькому порогу входження. Також, Python може взаємодіяти з багатьма іншими бібліотеками. Наразі

основними бібліотеками, які використовуються у машинному навчанні, окрім згаданих вище, є NumPy, SciPy, MatPlotLib, Scikit-learn, Pandas та інші.

2.3 Розробка програмної системи

У цьому підрозділі буде розглянутий процесс розробки системи, а саме причини прийняття певних рішень.

2.3.1 Попередня підготовка даних

ISIC архів надає зображення у наступному форматі для кожного даного: зображення у форматі JPG або PNG та JSON-файл з мета інформацією про зображення. Власне, з мета інформації нас цікавить тільки остаточний діагноз: чи є родимка злоякісною або доброякісною. Тобто, принаймні для первинної імплементації мережі, єдиною вхідною ознакою буде фото родимки з поблизу.

Відмінною характерною рисою злоякісною родимки, яка може призвести до ракового утворення, є її «неправильна» форма, тобто, якщо родимка не є симетричною, а також, якщо цю форму складають різні кольори. Тобто, безпечна родимка буде мати більш менш однорідне забарвлення.

Відомо, що усі зображення мають різні розміри, але пропорції фото є однаковими. Щоб розміри не були досить великим для опрацювання звичайними нейронними мережами, тому у якості попередньої обробки необхідно зменшити розмірність зображення, перевести його у чорно-біле (для того, щоб кожний

піксель можна було представити у значення від 0 до 255) ті збільшити контраст зображення (для усунення несуттєвих ознак, які можуть лише знизити ефективність роботи мережі).

Усього з сайту ISIC [4] було завантажено 2774 файли, тобто 1387 зображення та їх метадані. З них 908 доброякісних та 479 злоякісних (сайт також видає родимки з неоднозначним діагнозом (‘benign_malignant’), але в області даного дослідження ми не будемо приймати до уваги ці дані.

Так як, зображення, які йдуть по порядку, є з одного й того ж ресурсу (клініки, людини і т.д.) і є схожими за масштабом, кольорами та іншими ознаками, їх необхідно попередньо перемішати, щоб різні типи зображень були наявні у тренувальних та тестових даних.

Наступним кроком необхідно коректно розподілити дані на тренувальну та тестову вибірку. Розподілимо дані таким чином, що 80% даних будуть використовуватися як тренувальні та 20% як тестові. При цьому обидві вибірки повинні мати однакову кількість зображень доброякісних та злоякісних родимок (табл. 2.1).

Таблиця 2.1 – Співвідношення тренувальних та тестових даних

	Доброякісні	Злоякісні	Всього
Всього	908	479	2774
Тренувальний набір	375 (41.3%)	390 (81.4%)	765 (80%)
Тестовий набір	103 (11.3%)	89 (18.6%)	192 (20%)

Далі перейдемо до нормалізації кожного зображення. Кожне дане потрібно перевести в однакове розширення. Якщо розмір буде занадто великий, то обчислювальна складність підвищиться, відповідно обмеження на швидкість відповіді будуть порушені, визначення розміру в даній задачі вирішується методом

підбору. Якщо вибрати розмір занадто маленький, то мережа не зможе виявити ключові ознаки.

Перед тим, як надавати вхідному шару нейронної мережі, необхідно перетворити зображення у чорно-біле, збільшити його контрастність на 4.0 (за термінологією бібліотеки PIL, рис. 2.1) та перевести значення пікселів в придатні для обробки нейронами. Отже, вхідні дані кожного конкретного значення пікселя нормалізуються в діапазон від 0 до 1, за формулою:

$$f = \frac{p-min}{max-min},$$

де p – значення пікселя (від 0 до 255),

min – мінімальне значення пікселя (0),

max – максимальне значення пікселя (255).

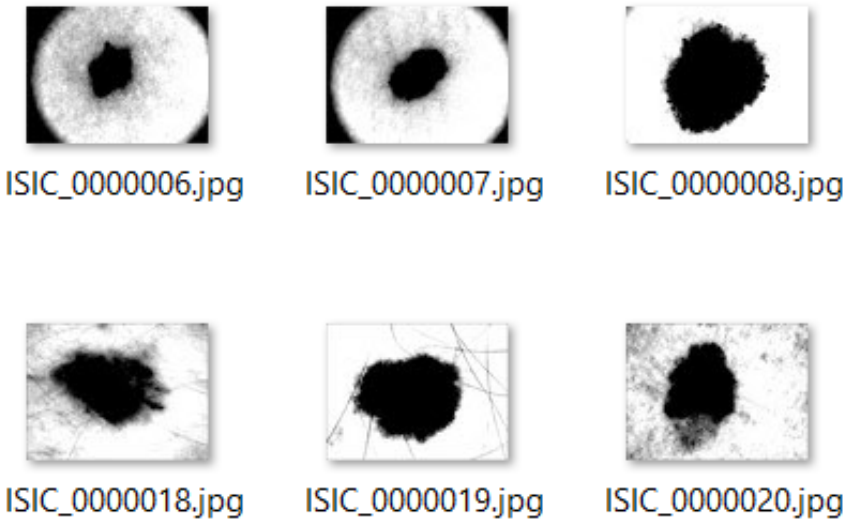

Рисунок 2.1 – Оброблені зображення

Очікуваний результат роботи мережі також треба перевести у придатні для обробки нейронами. Позначимо доброякісні (benign) родимки як 0, злоякісні (malignant) як 1. Таким чином, якщо у вихідному шарі нейрону маємо значення менше за 0.5, то родимка є доброякісною, у противному разі – злоякісною.

Таким чином, для кожного даного на вхід мережі маємо матрицю з нормалізованими значеннями пікселів, та відповідний результат: 0 чи 1.

2.3.2 Проектування нейронної мережі

Першою спробою розробки системи було створення звичайної нейронної мережі з використанням вхідного шару зі стількох нейронів, скільки пікселів було у зображенні. Але, таким чином, модель майже не видавала правильних результатів. Тож, було прийняте рішення використовувати згорткову нейромережу.

Початкова структура мережі представлена на рис. 2.2. вона містить два шари згортки, шар, який перетворює двовимірні масиви у одномірні (операція сплющення) та вихідний повнозв’язаний шар, який на виході може показувати 2 значення: 1 чи 0.

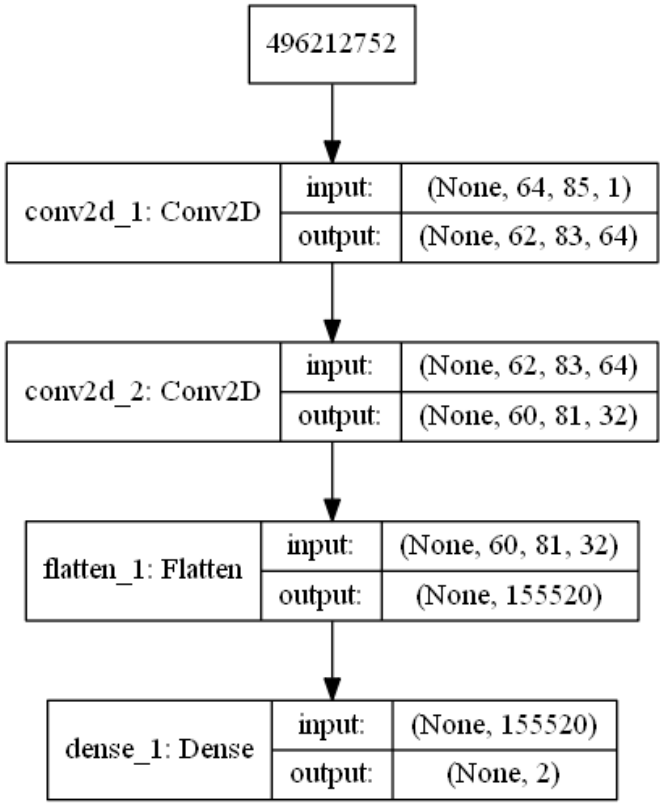

Рисунок 2.2 – Первинна структура мережі

Для перевірки роботи моделі, у метод fit у якості даних для валідації передається тестовий датасет. До того ж, Keras надає можливість зупинити навчання у тому моменті часу, коли метрики моделі не покращуються. Для даної моделі використовуємо наступну умову: зупинити навчання, коли значення функції втрат для даних валідації (у цьому випадку, тестових даних) перестає зменшуватися. Тоді метод тренування моделі виглядатиме наступним чином:

```
early_stopping = EarlyStopping(
        monitor='val_acc',
        patience=10,
        mode='max',
        verbose=1)
history = model.fit(train_images,
                    train_labels,
                    epochs=50,
                    validation_data=(test_images, test_labels),
                    callbacks=[early_stopping])
```

У якості оптимізатору використовується Adam (adaptive moment estimation) – це модифікований метод стохастичного градієнтного спуску. На відміну від градієнтного спуску, який використовує постійну швидкість навчання, Adam змінює швидкість під час навчання. До того ж, зазвичай не потребує налаштування, так як параметри його роботи за замовчування, показують досить гарну ефективність.

Графіки прогресу навчання моделі показані на рис 2.3, рис. 2.4. На осі X відображена кількість епох тренування – тобто ітерацій навчання системи (для кожної епохи всі тренувальні дані проходять через модель та проходить зміна вагів нейронів), на осі У – точність та значення функції втрат.

Результуюча ефективність роботи на тестових даних дорівнює 64% правильних результатів. Цей показник є не достатньо високим, тож будемо покращувати структуру нейромережі для отримання кращих результатів.

На рисунку 2.3 можна побачити, що коли точність передбачень продовжує зростати на тренувальних даних, то точність передбачень на тестових вже починає спадати (після третьої епохи). Підсумкова точність моделі на тестових даних становить близько 64%, на тренувальних – 82%. Тобто відбувається перенавчання системи.

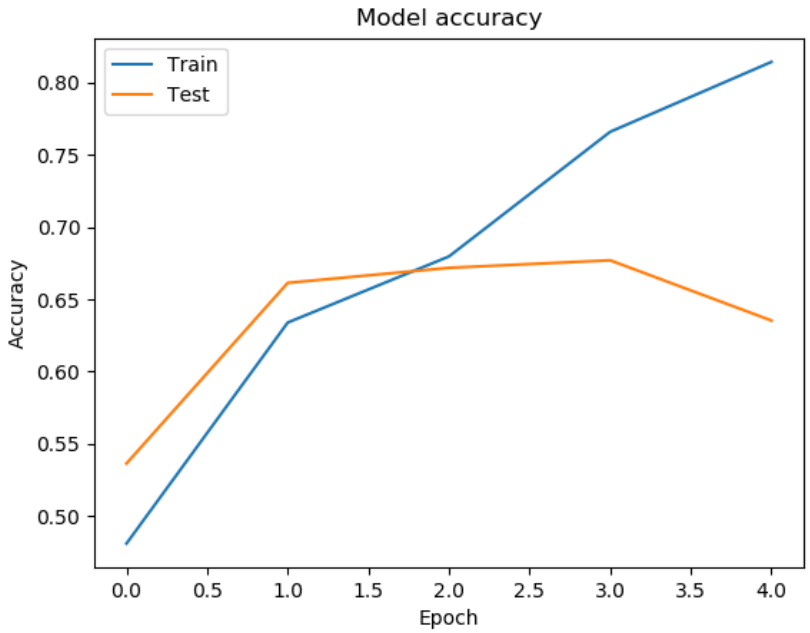

Рисунок 2.3 – Графік точності передбачень

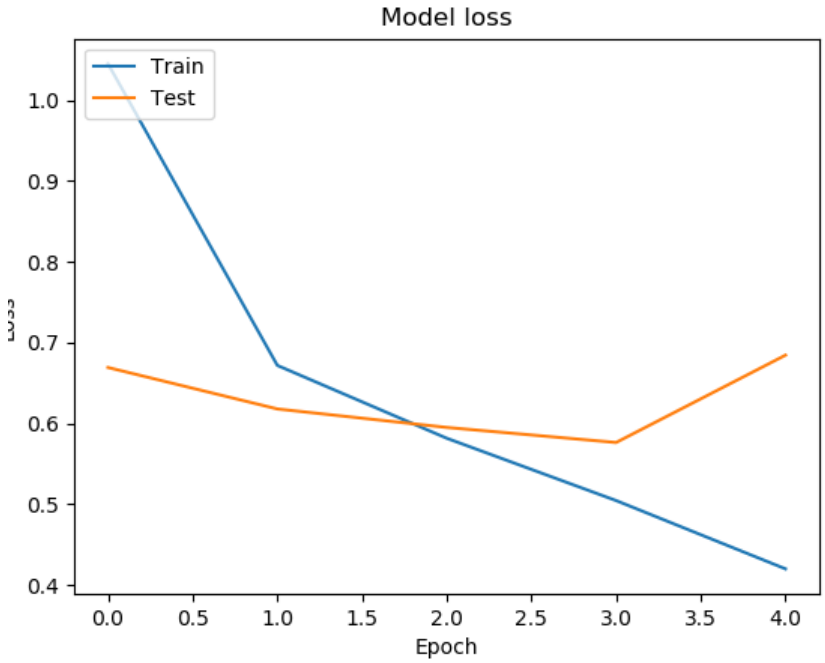

Рисунок 2.4 – Графік втрат

На рисунку 2.4. можна побачити такий самий тренд, коли втрати на тестових даних підвищуються.

Спробуємо додати шари агрегування (MaxPooling2D) після кожного шару згортки. Кожен шар агрегування є максимізаційним агрегування з розміром ядра 2x2, метою якого є зменшення просторового розміру виділеної на попередньому шарі ознаки та, як наслідок, зменшення необхідної обчислювальної потужності. Структура мережі на рис. 2.5, результати роботи на рис. 2.6, рис. 2.7.

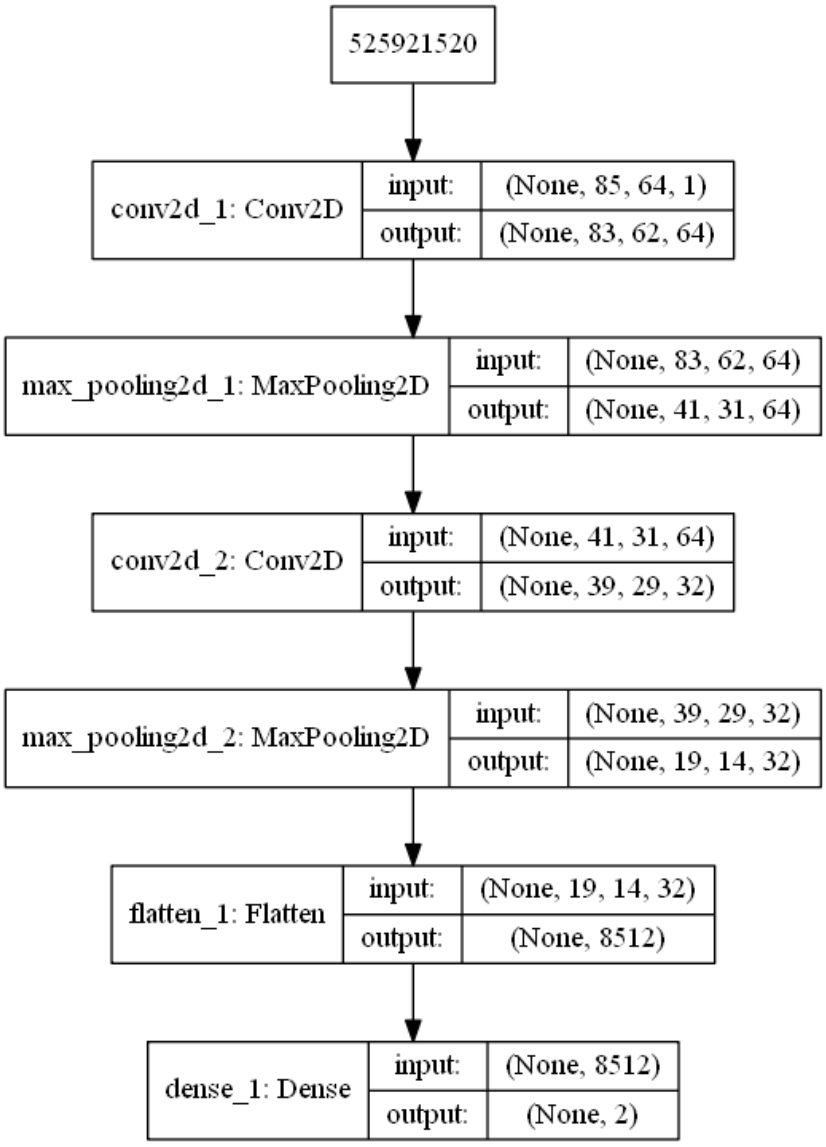

Рисунок 2.5 – Структура нейромережі, етап 2

Результат роботи нейромережі після даних перетворень на тестовому сеті становить 71.9%, на тренувальному – 75%. Все же можна відмітити, що модель краще себе показує на тренувальних даних, ніж на тестових, але розрив між цими показниками скоротився з 18% до 3.1%.

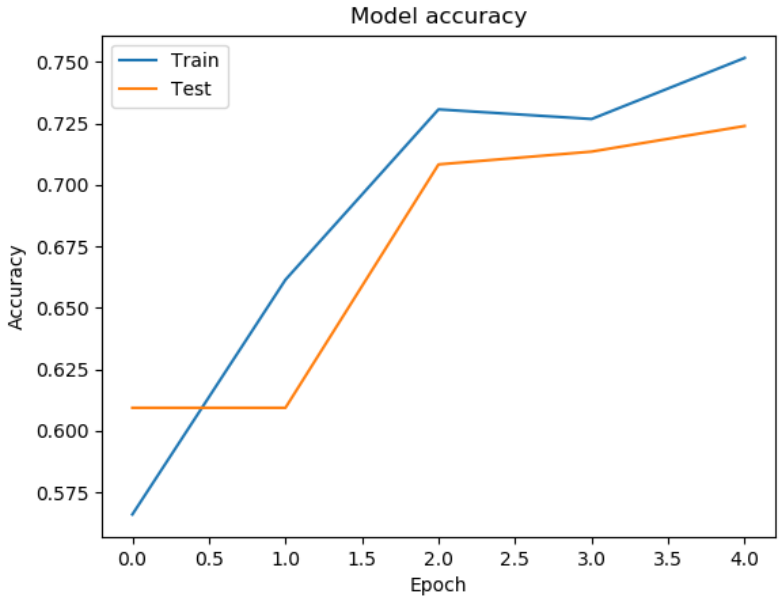

Рисунок 2.6 – Графік точності передбачень, етап 2

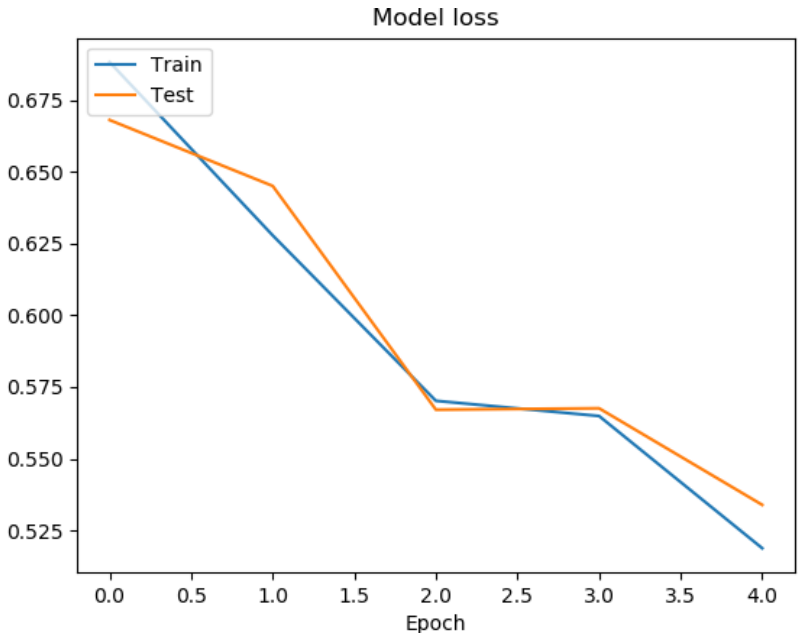

Рисунок 2.7 – Графік втрат, етап 2

Далі спробуємо визначити оптимальну кількість епох тренування, підвищивши кількість ітерацій на попередній моделі до 70. На рис. 2.8 можна побачити, що модель з перебігом часу показує кращі результати на тренувальному сеті, але на тестовому сеті не видно значних покращень десь після двадцятої епохи, тобто перенавчання лише прогресує.

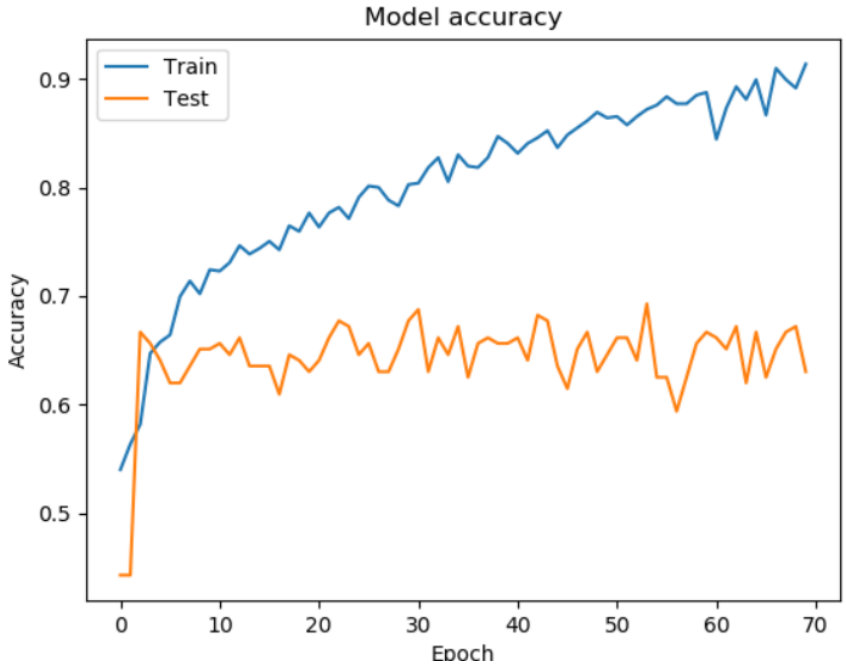

Рисунок 2.8 – Прогрес навчання моделі на великій кількості ітерацій

Після декількох експериментів з методами регуляризації, у тому числі L1- та L2-регуляризація, пакетна нормалізація (Batch normalization) та техніка виключення (Dropout) можна зробити наступні висновки для даних медичних зображень:

– Пакетна нормалізація лише збільшує перенавчання, на тестовому сеті модель показує результат близько 96%, у той час на тренувальному – не більше 65%;

– L1- та L2- регуляризації не показують великого покращення у порівнянні з шарами без застосування методів регуляризації;

– Найбільше покращення помітне при застосуванні техніки Dropout, який додається як додатковий шар після другого та третього шарів згортки, з процентом нейронів, які видаляються, рівному 25%.

Суть Dropout полягає у тому, що при навчанні мережі трапляється виключення з неї нейронів з певною ймовірністю p, таким чином, вірогідність того, що нейрон залишиться в мережі, становить $q = 1 - p$ [11]. "Виключення" нейрона означає, що при будь-яких вхідних даних або параметрах він повертає 0.

Виключені нейрони не роблять свій внесок в процес навчання на жодному з етапів алгоритму зворотного поширення помилки; тому виключення хоча б одного з нейронів рівносильно навчанню нової нейронної мережі.

У якості оптимізатора краще себе показав звичайний стохастичний градієнтний спуск з наступними параметрами: швидкість навчання 0.01 (значення за замовчуванням у Keras), значення імпульсу 0.6 (яке означає, що алгоритм буде використовувати результати попереднього етапу для передбачення) та ослаблення швидкості навчання 0.01 розділене на кількість епох (примірно 20).

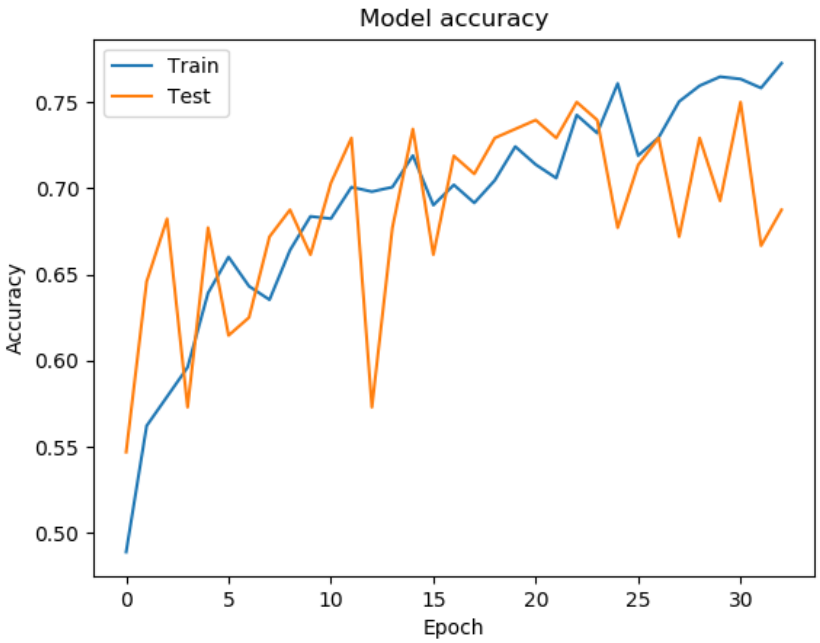

Рисунок 2.9 – Графік точності передбачень, етап 3

До того ж, був доданий ще один шар згортки , але вже без шару агрегування, адже кількість ознак після нього вже достатньо мала. Отримані результати показані на рис. 2.9, рис. 2.10.

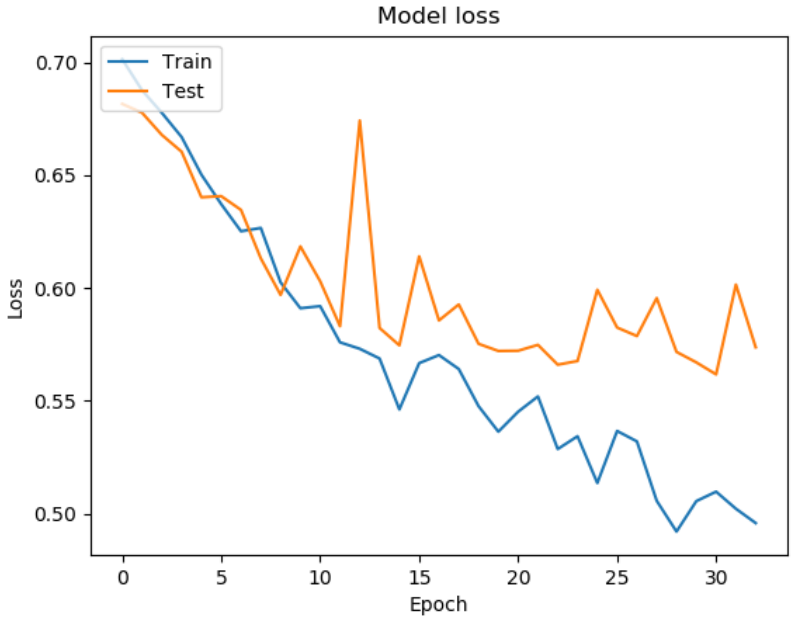

Рисунок 2.10 – Графік функції втрат, етап 3

Найкращий результат роботи моделі (епоха 22) на тренувальних даних становить 74%, на тестових – 75%. Фінальна модель нейронної мережі представлена на рисунку 2.11.

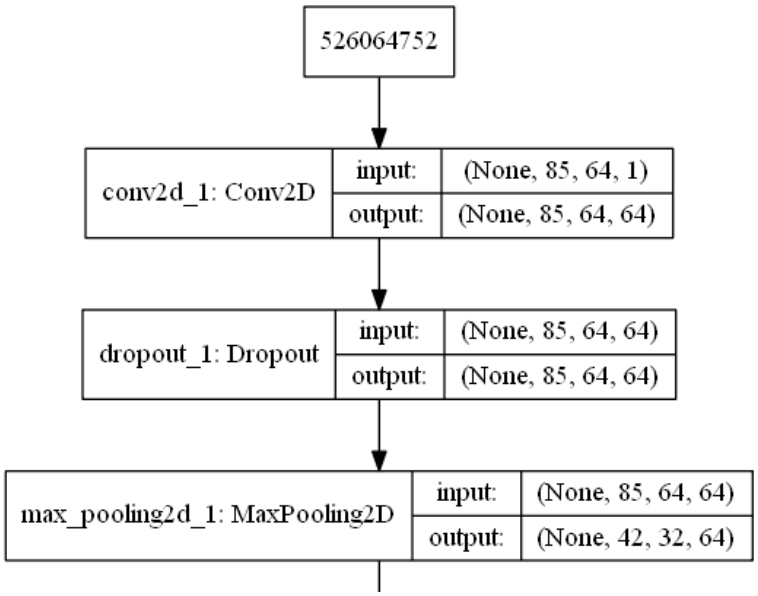

Рисунок 2.11 – Структура мережі на 3-му етапі, аркуш 1

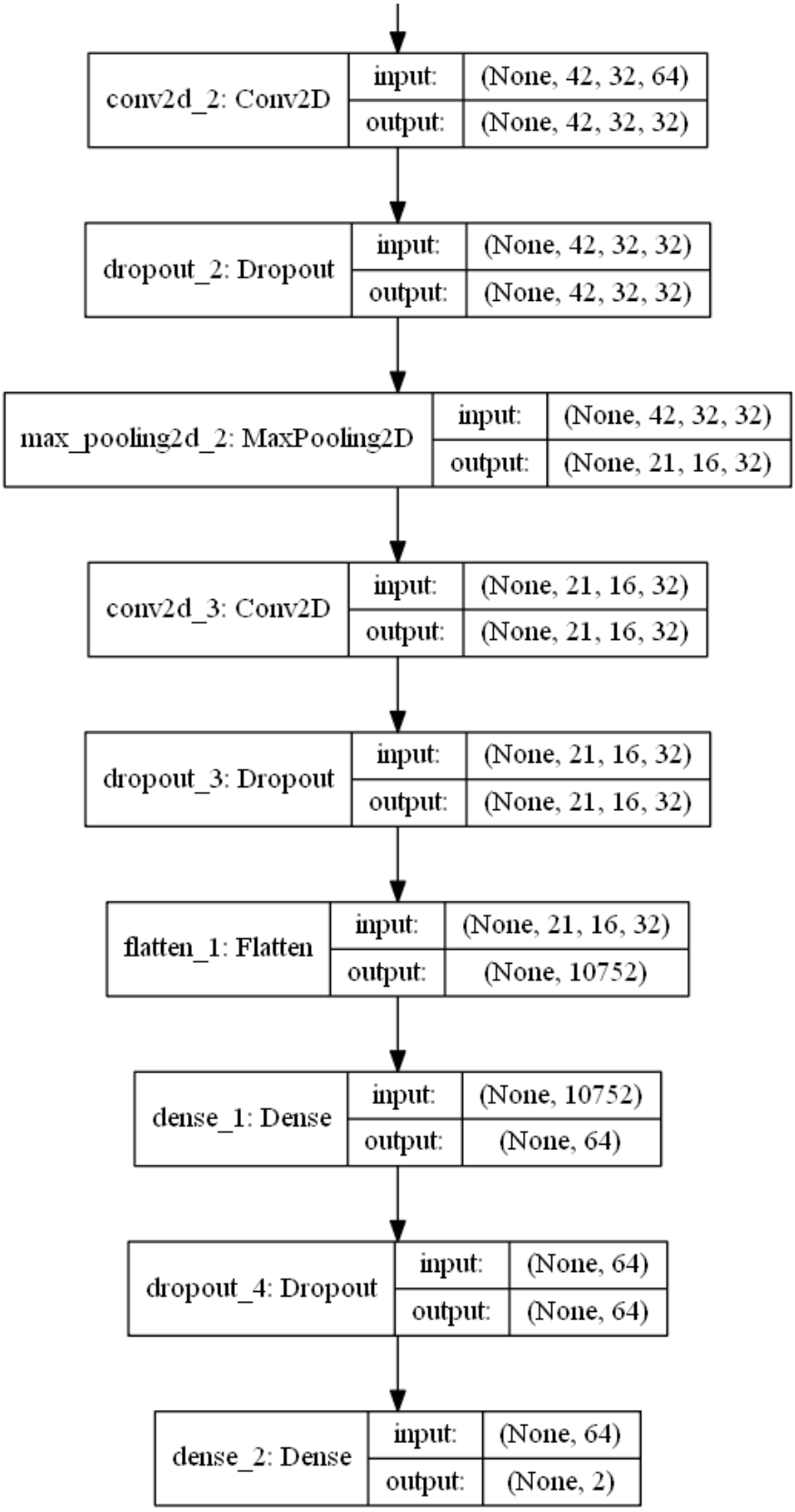

Рисунок 2.11 – Структура мережі на 3-му етапі, аркуш 2

Мережа містить три шари згортки, кожен з яких супроводжується шаром видалення нейронів та шаром агрегування. Останнім йде повнозв’язаний шар, шар видалення нейронів, та вихідний шар. Після шарів згортки є перший повнозв`язний шар та вихідний повнозв’язний. Більш детально структура мережі наведена у додатку А.

Крім EarlyStopping, до колбеків був доданий ModelCheckpoint, який зберігає модель з найкращим показником ефективності на тестових даних у файл HDF5. Цей формат файлів створений для зберігання дуже великих даних, зміст яких організовано як ієрархічна файлова система. Розмір файлу становить 5.5 мегабайт та з нього можна у будь який час завантажити натреновану модель.

ВИСНОВКИ

Під час виконання науково-дослідницької роботи була досліджена структура штучних нейромереж, їх види та задачі, до яких вони можуть бути застосовані.

Нелінійність нейронних мереж дозволяє встановлювати нелінійні залежності між майбутніми та фактичними значеннями процесів. іншими важливими перевагами є масштабованість – паралельна структура штучних нейронних мереж прискорює обчислення, що є вкрай актуальним в промислових масштабах, коли необхідно обробляти терабайти даних.

Був вибраний датасет з ресурсу Міжнародної спілки зображень шкіри (ISIC Archive), а саме проект «Меланома». Через те, що ця база містить велику кількість зображень з відповідями на те, чи є шкірне утворення злоякісним чи доброякісним, можна використовувати підхід навчання з учителем, та одну і ту саму базу для тренування і тестування мережі. Фотографії у базі даних є стандартизованими та більша їх кількість має одинакові розміри.

У результаті виконання роботи було розроблено штучну нейронну мережу мовою програмування Python, використовуючи бібліотеку машинного навчання TensorFlow та Keras. Модель нейронної мережі містить три шари згортки, шари агрегування, шари видалення «зайвих» нейронів. Було проведено дослідження роботи моделі з різними параметрами та вибрані найкращі за ознакою точності показників мережі на тестових даних. Кінцева ефективність роботи дорівнює 75%.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. McCulloch W, Pitts W. A Logical Calculus of the Ideas Immanent in Nervous Activity [Електронний ресурс] / W.S. McCulloch, W.H. Pitts // Bulletin of Mathematical Biophysics. – 1943. – URL: http://www.cse.chalmers.se/~coquand/AUTOMATA/mcp.pdf (дата звернення: 23.03.2019)

2. Huang G., Liu Z., K. Q. Weinberger. Densely connected convolutional network [Електронний ресурс] / G. Huang, Z. Liu, Weinberger K.Q. – 2016. – URL: https://arxiv.org/abs/1608.06993v1 (дата звернення: 12.04.2019)

3. Improving Diagnosis in Health Care [Електронний ресурс] // Institute of Medicine. – 2015. – URL: http://www.nationalacademies.org/hmd/~/media/Files/Report%20Files/2015/Improving-Diagnosis/DiagnosticError_ReportBrief.pdf (дата звернення: 27.03.3019)

4. ISIC Archive – The International Skin Imaging Collaboration – URL: https://www.isic-archive.com/ (дата звернення: 5.04.2019)

5. Каллан Р. Основные концепции нейронных сетей = The Essence of Neural Networks First Edition. — «Вильямс», 2001. — 288 с

6. Новотарський М.А., Нестеренко Б.Б. Штучні нейронні мережі: обчислення [Електронний ресурс] / М.А. Новотарський, Б.Б. Нестеренко // Інститут математики НАН України. – 2004. – URL: http://www.immsp.kiev.ua/postgraduate/Biblioteka_trudy/ShtuchnNejronMeregNester2004.pdf (дата звернення: 5.04.2019)

7. Розенблатт, Ф. Принципы нейродинамики: Перцептроны и теория механизмов мозга. Principles of Neurodynamic: Perceptrons and the Theory of Brain Mechanisms. — М.: Мир, 1965. — 480 с.

8. Krizhevsky A., Sutskever I. ImageNet Classification with Deep Convolutional Neural Networks [Електронний ресурс] / A. Krizhevsky, I. Sutskever, E. Geoffrey. – 2012. – URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf (дата звернення: 15.04.2019)

9. Ruder S. An overview of gradient descent optimisation algorithms [Електронний ресурс] / S. Ruder – 2016. – URL: https://arxiv.org/abs/1609.04747 (дата звернення: 17.04.2019).

10. Rumelhart D., Hinton G. Learning representations by back-propagating errors [Електронний ресурс] / D. Rumelhaft, G.Hinton, R. Williams. – 1986. – URL: https://www.nature.com/articles/323533a0 (дата звернення: 18.04.2019)

11. Srivastava N., Hinton G.E. Dropout: a simple way to prevent neural networks from overfitting [Електронний ресурс] / N. Srivastava , G.E. Hinton – 2014. – URL: https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf (дата звернення: 28.05.2019)