Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи обробки зображень на основі згорткових нейронних мереж

(тема)

Виконав:
студент II курсу, групи СПм-18-3
Доліна К.А.
(прізвище, ініціали)

Спеціальність
123 – Комп’ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: проф. Руденко О.Г.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність (напрям) 6.050102 – Комп’ютерна інженерія
(код і назва)

Освітня програма Системне програмування
(повна назва)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Доліній Карині Андріївні
(прізвище, ім’я, по батькові)

1. Тема роботи Методи обробки зображень на основі згорткових нейронних мереж

затверджена наказом по університету від “____” _____ 2020 р. № _____

2. Термін подання студентом роботи до екзаменаційної комісії 18 травня 2020 р.

3. Вхідні дані до роботи

4. Перелік питань, що потрібно опрацювати в роботі

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) _____

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка

Дата видачі завдання _____

Студент _____
(підпис)

Керівник роботи _____ проф.Руденко О.Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 74 с., 36 рис., 1 дод., 17 джерел.

НЕЙРОННА МЕРЕЖА, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ЗОБРАЖЕННЯ, ОБРОБКА, АЛГОРИТМИ

Метою атестаційної роботи є аналіз існуючих методів обробки зображень та створити свою згорткову нейронну мережу для класифікації зображень рукописних цифр та їх розпізнавання.

У ході виконання атестаційної роботи був здійснен аналіз різних алгоритмів фільтрації зображень і оптимізація їх під завдання магістерської роботи, а саме методи обробки зображень на основі згортальних нейронних мереж. Завдяки цьому можна прийти до зовсім нового напрямку в обробці зображень. Обробка графічних файлів - сфера, яка в сучасному світі комп'ютерних технологій швидко і інтенсивно розвивається, з'являються нові засоби та алгоритми фільтрації зображень. На даний момент вже створено досить багато редакторів, які працюють на персональних комп'ютерах і можуть виконувати дуже складні графічні функції. Однак не всі використовують нейронні мережі, які надають можливість нового рівня якості аналізу зображення цифр. Цей факт надає можливості для розробки алгоритмів згортальних нейронних мереж, які будуть максимально ефективно виконувати аналіз зображень на новому рівні.

ABSTRACT

Master’s thesis: 74 pages, 36 figures, 1 appendices, 17 sources.

NEURAL NETWORK, ROLLED NEURAL NETWORKS, IMAGES, PROCESSING, ALGORITHMS

The purpose of the certification work is to analyze the existing methods of image processing and to create its own convolutional neural network for the classification of images of handwritten numbers and their recognition.

During the certification work the analysis of various algorithms of filtering of images and their optimization under tasks of the master’s work, namely methods of processing of images on the basis of convolutional neural networks was carried out. This allows you to come to a whole new direction in image processing. Graphic file processing is a field that is rapidly and intensively developing in the modern world of computer technology, new tools and algorithms for image filtering are appearing. At the moment, there are already many editors who work on personal computers and can perform very complex graphics functions. However, not everyone uses neural networks, which provide a new level of image image quality analysis. This fact provides opportunities for the development of algorithms for convolutional neural networks that will most effectively perform image analysis at a new level.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГНМ (deep neural network, DNN) – глибока нейронна мережа

ЗНМ – згорткова нейронна мережа

ШНМ (штучна нейронна мережа) – обчислювальна нелінейна модель, на основі котрій є нейронна структура мозга. Вона здатна навчатися виконанню задач класифікації, передбачення, прийняття рішень та візуалізації.

LSTM (Long Short-Term Memory) – мережа довгої короткостроковій пам’яті

ВСТУП

Цифрова обробка та подальше розпізнавання зображень – один з напрямів наукових досліджень, що інтенсивно розвивається. Багато галузей техніки, що мають відношення до отримання, обробці, зберіганню та передачі інформації, в значній мірі орієнтуються на розвиток систем, в котрих інформація має характер зображень.

Сьогодні нейронні мережі набирають все більшу популярність, та все частіше використовуються у різних сферах життя. Особливо часто нейромережі використовуються, коли це стосується задач аналізу даних, кластеризації, розпізнавання об’єктів на зображеннях, виявлення схованих закономірностей, комп’ютерного зору, прогнозування даних.

Согодні налічується багато різних архітектур нейронних мереж, кожна з яких виконує свої функції, має певні алгоритми тренування, та має переваги й недоліки порівняно із іншими архітектурами.

Новий подих розвитку та використанню нейронних мереж дав розвиток згорткових нейронних мереж, особливо що стосується сфери комп’ютерного зору, кластеризації, та аналізу візуалізації зображень. Головною особливістю цієї архітектури є те, що вона використовує операцію згортки для того, щоб зменшити розмір зображення, виділяючи й розпізнаючи характерні ознаки, в залежності від поставленої задачі. Нейронна мережа поступово та ітеративно проходить по зображенню, виконуючи операцію згортки. Поступово вона отримує все більше даних, які необхідні їй для розпізнавання. На якість розпізнання впливає кількість та різні комбінації шарів згортки всередині топології згорткової нейронної мережі.

Згорткова ж нейронна мережа за рахунок застосування спеціальної операції – власне згортки – дозволяє водночас зменшити кількість інформації, що зберігається в пам’яті інформації, за рахунок чого краще справляється з картинками більш високої роздільної здатності, і виділити

опорні ознаки зображення, такі як ребра, контури або грані. На наступному рівні обробки з цих ребер і граней можна розпізнати повторювані фрагменти текстур, які далі можуть скластися в фрагменти зображення.

Для досягнення найкращих результатів роботи згорткових нейронних мереж є безліч засобів. Зокрема це: гістограми датасетів, представлення зображення у полутоновій формі, розпаралелювання процесів нейронної мережі, додання афінних перетворень до зображень з датасетів, різні ядра згортки для зменшення зображень та інше.

В цій атестаційній роботі будуть детально розглянуті й проаналізовані найпоширеніші алгоритми попередньої обробки зображень, та будуть надані практичні рекомендації по використанню цих алгоритмів, в залежності від існуючих умов й поставленої задачі.

1 НЕЙРОННІ МЕРЕЖІ

1.1 Архітектура нейронних мереж

Будь-яка архітектура ШНМ складається з штучних нейронів – елементів обробки, мають структуру трьох пов’язаних один з одним шарів: вхідним, що складається з одного або кількох шарів, прихованим і вихідним (рисунок 1.1).

Вхідний шар складається з вхідних нейронів, які передають інформацію в приховані шари. Прихований шар в свою чергу передає інформацію в вихідний. Кожен нейрон має входи з вагами - синапсами, функцію активації, визначальну вихідну інформацію при заданої вхідної, і один вихід. Синапси - регульовані параметри, що конвертують нейронну мережу в параметризрвану систему.

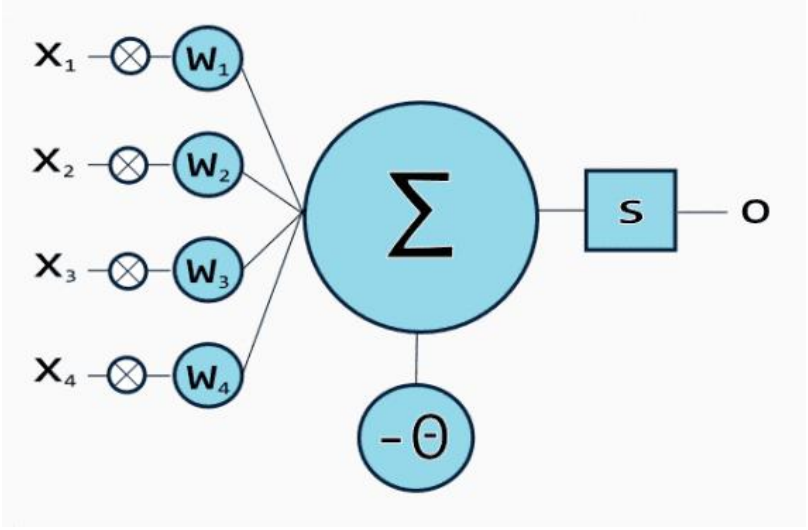


Рисунок 1.1 – Штучна нейронна мережа

Навчання (або тренування) – процес оптимізації ваг, в якому мінімізується помилка передбачення, і мережа досягає необхідного рівня точності. Найбільш використовуваний метод для визначення внеску в

помилку кожного нейрона – зворотнє поширення помилки, за допомогою якого обчислюють градієнт. Це одна з модифікацій методу градієнтного спуску.

За допомогою додаткових прихованих шарів можливо зробити систему більш гнучкою і потужною. ШНМ з багатьма прихованими шарами називаються глибокими нейронними мережами; вони створюють складні нелінійні зв’язки.

Розглянемо популярні архітектури нейронних мереж, які добре показали себе в задачах нейролінгвістичного програмування і рекомендуються до використання.

1.1.1 Багатошаровий перцептрон

Багатошаровий перцептрон (рисунок 1.2) складається з трьох або більше шарів. Він використовує нелінійну функцію активації, часто тангенціальну або логістичну, яка дозволяє класифікувати лінійно нероздільні дані. Кожен вузол в шарі з’єднаний з кожним вузлом в наступному шарі, що робить мережу повністю пов’язаною. Така архітектура знаходить застосування в задачах розпізнавання мови та машинному перекладі.

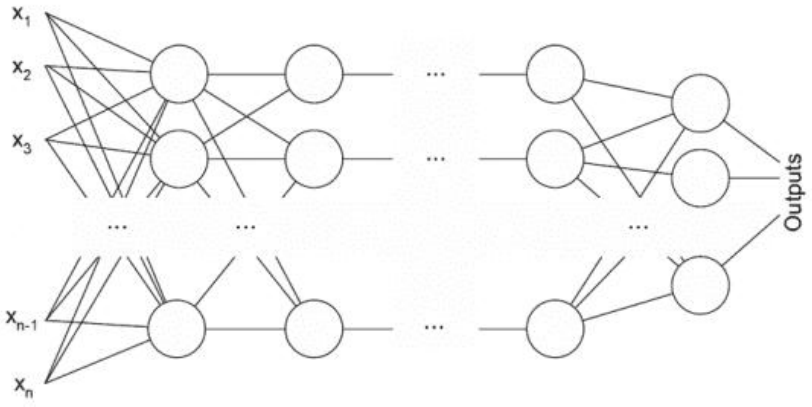


Рисунок 1.2 – Перцептрон

1.1.2 Згорткова нейронна мережа

Згорткова нейронна мережа (Convolutional neural network, CNN) містить один або більше об’єднаних або поєднаних згортальних шарів. CNN використовує варіацію багатошарового перцептрона, розглянутого вище. Згорткові шари використовують операцію згортки для вхідних даних і передають результат в наступний шар (рисунок 1.3). Ця операція дозволяє мережі бути глибше з меншою кількістю параметрів.

Згорткові мережі показують видатні результати в додатках до картинок і мови. У статті Convolutional Neural Networks for Sentence Classification автор описує процес і результати завдань класифікації тексту за допомогою згорткових нейронних мереж. У роботі представлена модель на основі word2vec, яка проводить експерименти, тестується на декількох бенчмарках і демонструє блискучі результати.

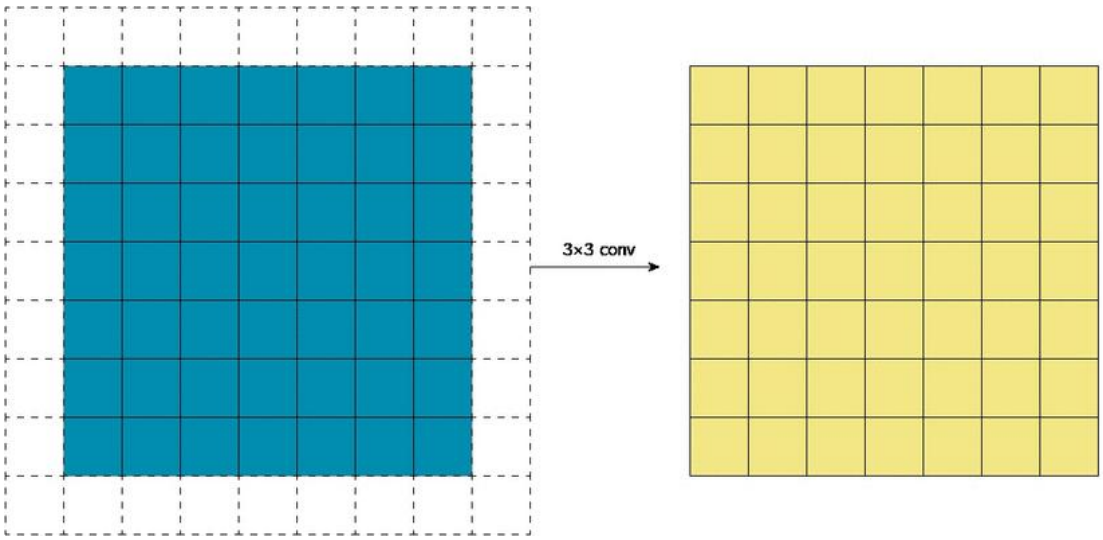


Рисунок 1.3 – Згорткова нейронна мережа

В роботі Text Understanding from Scratch автори показують, що згорткова мережа досягає видатних результатів навіть без знання слів, фраз пропозицій і будь-яких інших синтаксичних або семантичних структур

притаманних людській мові. Семантичний розбір, пошук парафраз, розпізнавання мови - теж додатки CNN.

1.1.3 Рекурсивна нейронна мережа

Рекурсивна нейронна мережа – тип глибокої нейронної мережі, сформована при застосуванні одних і тих же наборів ваг рекурсивно над структурою, щоб зробити скалярне або структуроване пророкування над вхідною структурою змінного розміру через активацію структури в топологічному порядку (рисунок 1.4). У простій архітектурі нелінійність, така як тангенціальна функція активації і матриця ваг використовуються для об’єднання вузлів в батьківські об’єкти.

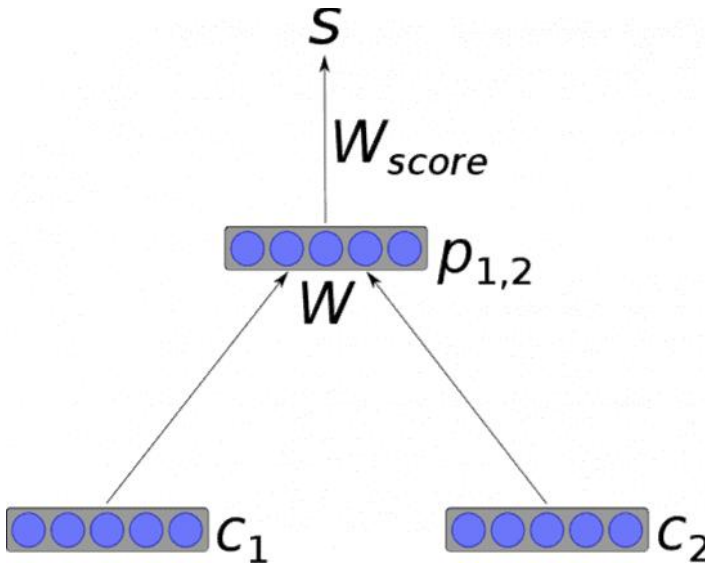


Рисунок 1.4 – Рекурсивна нейронна мережа

1.1.4 Рекурентна нейронна мережа

Рекурентна нейронна мережа, на відміну від прямої нейронної мережі, є варіантом рекурсивної ШНМ, в якій зв’язки між нейронами – спрямовані цикли. Останнє означає, що вихідна інформація залежить не тільки від

поточного входу, але також від станів нейрона на попередньому кроці. Така пам’ять дозволяє користувачам вирішувати завдання нейролінгвістичного програмування: розпізнання рукописного тексту або мови. У статті Natural Language Generation, Paraphrasing and Summarization of User Reviews with Recurrent Neural Networks автори показують модель рекуррентной мережі, яка генерує нові пропозиції та короткий зміст текстового документа.

Siwei Lai, Liheng Xu, Kang Liu, і Jun Zhao в своїй роботі Recurrent Convolutional Neural Networks for Text Classification створили рекуррентну згорткову нейромережу для класифікації тексту без рукотворних ознак. Модель порівнюється з існуючими методами класифікації тексту - Bag of Words, Bigrams + LR, SVM, LDA, Tree Kernels, рекурсивними і згортковими мережами. Описана модель перевершує за якістю традиційні методи для всіх використовуваних датасета.

1.1.5 Мережа довгої короткостроковій пам’яті

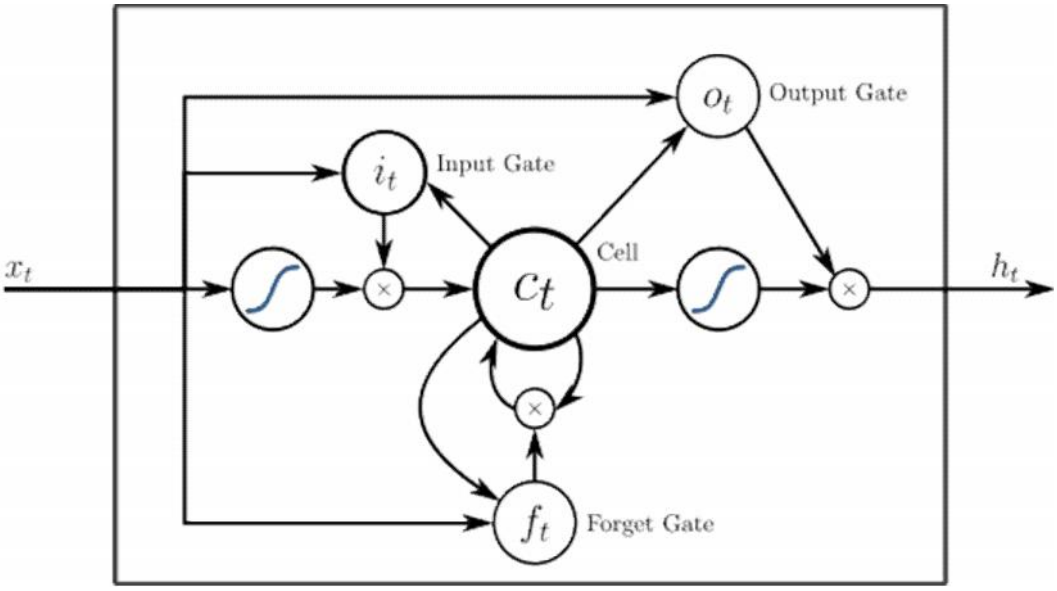


Рисунок 1.5 – Мережа довгої короткостроковій пам’яті з вхідним, вихідним і гейтом забування

Мережа довгої короткострокової пам’яті (LSTM) – різновид архітектури рекуррентної нейромережі, створена для більш точного моделювання часових послідовностей і їх довгострокових залежностей, ніж традиційна рекуррентна мережа. LSTM-мережа не використовує функцію активації в рекурентних компонентах, збережені значення не модифікуються, а градієнт не прагне зникнути під час тренування. Часто LSTM застосовується в блоках по кілька елементів. Ці блоки складаються з 3 або 4 затворів (наприклад, вхідного, вихідного і гейта забування), які контролюють побудову інформаційного потоку з логістичної функції (рисунок 1.5).

В Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling автори показують архітектуру глибокої LSTM рекуррентной мережі, яка досягає хороших результатів для великомасштабного акустичного моделювання.

В роботі Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network представлена модель для автоматичної морфологічної розмітки. Модель показує точність 97.4% в завданні розмітки. Apple, Amazon, Google, Microsoft та інші компанії впровадили в продукти LSTM-мережі як фундаментальний елемент.

1.1.6 Sequence-to-sequence модель

Часто Sequence-to-sequence моделі складаються з двох рекурентних мереж: кодувальника, який обробляє вхідні дані, і декодера, який здійснює висновок.

Sequence-to-Sequence моделі часто використовуються в питально-відповідних системах, чат-ботах і машинному перекладі.

У Paraphrase Detection Using Recursive Autoencoder представлена нова рекурсивна архітектура автошифрувальника, в якій уявлення – вектора в n-вимірному семантичному просторі, де фрази зі схожими значенням близькі один до одного.

1.1.7 Неглибокі (shallow) нейронні мережі

Неглибокі моделі, як і глибокі нейронні мережі, теж популярні і корисні інструменти. Наприклад, word2vec - група неглибоких двошарових моделей, яка використовується для створення векторних уявлень слів (word embeddings). Представлена в Efficient Estimation of Word Representations in Vector Space, word2vec приймає на вході великий корпус тексту і створює векторний простір. Кожному слову в цьому корпусі приписується відповідний вектор в цьому просторі. Відмітна властивість – слова із загальних текстів в корпусі розташовані близько один до одного в векторному просторі.

2 МЕТОДИ ОБРОБКИ ЗОБРАЖЕНЬ

Попередня обробка зображення – процес поліпшення якості зображення, що ставить за мету отримання на основі оригіналу максимально точного і адаптованого для автоматичного аналізу зображення.

Серед дефектів цифрового зображення можна виділити наступні види:

- цифровий шум;
- кольорові дефекти (недостатні або надлишкові яскравість і контраст, неправильний колірний тон);
- розмитість (расфокусировка).

Методи попередньої обробки зображень залежать від завдань досліджень і можуть включати наступні види робіт:

- фільтрація зашумленних зображень;
- корекція яскравості і контрасту.

2.1 Фільтрація зашумлених зображень

Цифровий шум зображення – дефект зображення, внесений фотосенсором і електронікою пристроїв, які їх використовують. Для його придушення використовують такі методи:

- лінійне усереднення точок по сусідах;
- розмиття по Гаусу;
- медіанна фільтрація;
- морфологічні перетворення.

2.1.1 Лінійне усереднення точок по сусідах

Лінійне усереднення точок по сусідах - найпростіший вид алгоритмів видалення шуму. Основна ідея їх в тому щоб брати середнє арифметичне

значення точок в деякому околі в якості нового значення точки.

Фізично така фільтрація реалізується за допомогою обходу пікселів зображення матрицею згортки, що має такий вигляд (рисунок 2.1)

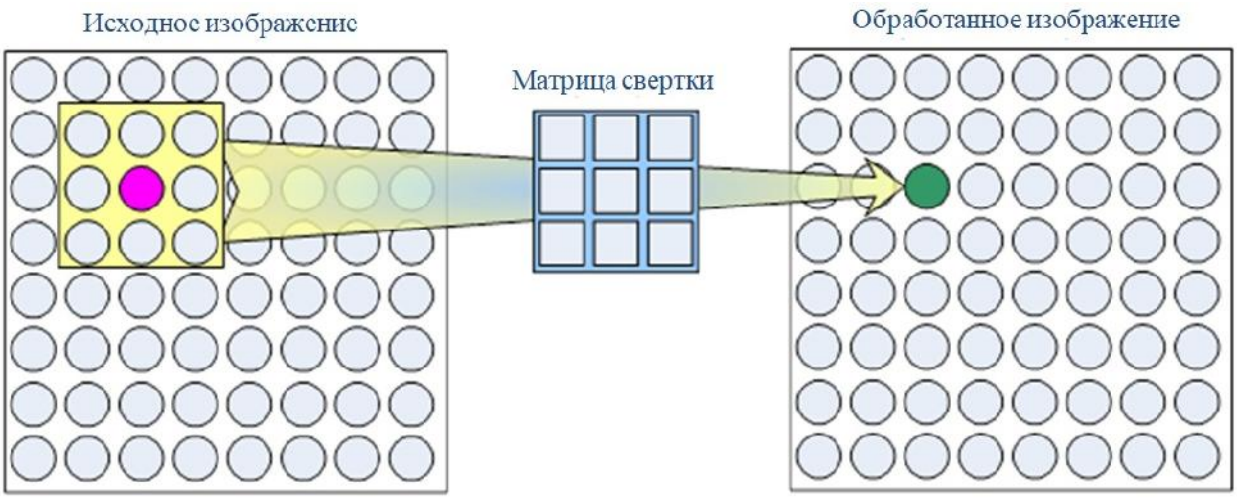


Рисунок 2.1 – Приклад роботи методу фільтрації

2.1.2 Розмиття за Гаусом

Розмиття за Гаусом (різновид лінійного згортання) реалізується за допомогою обходу пікселів зображення матрицею згортки, що має такий вигляд:

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Рисунок 2.2 – Матриця згортки

Матриця 5 × 5 заповнюється за нормальним (Гауссового закону). Нижче приведена та ж матриця, де коефіцієнти вже є нормованими, так що div для цієї матриці дорівнює одному.

0,000789	0,006581	0,013347	0,006581	0,000789
0,006581	0,054901	0,111345	0,054901	0,006581
0,013347	0,111345	0,225821	0,111345	0,013347
0,006581	0,054901	0,111345	0,054901	0,006581
0,000789	0,006581	0,013347	0,006581	0,000789

Від розміру матриці залежить сила розмиття (рисунок 2.3).

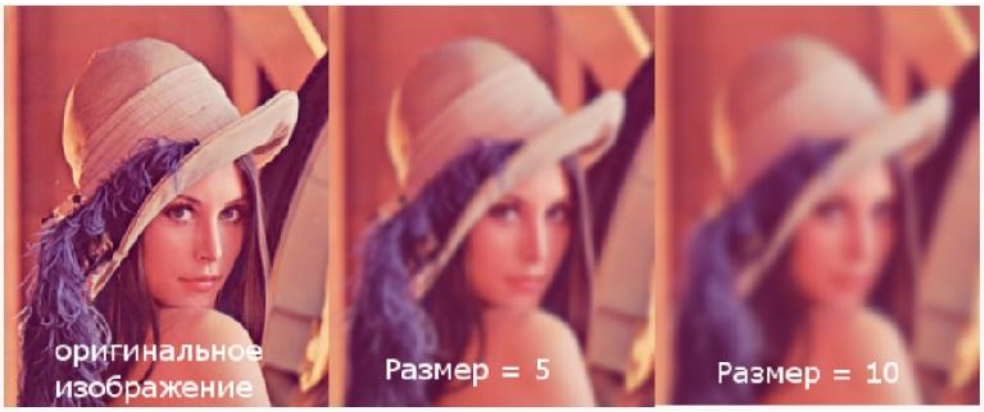


Рисунок 2.3 – Приклад розмиття

У верхнього лівого пікселя не існує «сусідів» зліва і зверху, отже, нам не потрібно множити коефіцієнти матриці (рисунок 2.4).

Для вирішення цієї проблеми потрібно створення проміжного зображення. Ідея в тому, щоб створювати тимчасове зображення з розмірами:

$$width + 2 \bullet gap / 2, height + 2 \bullet gap / 2, \quad (2.1)$$

де width і height - ширина і висота зображення що фільтрується;

gap - розмірність матриці згортки.

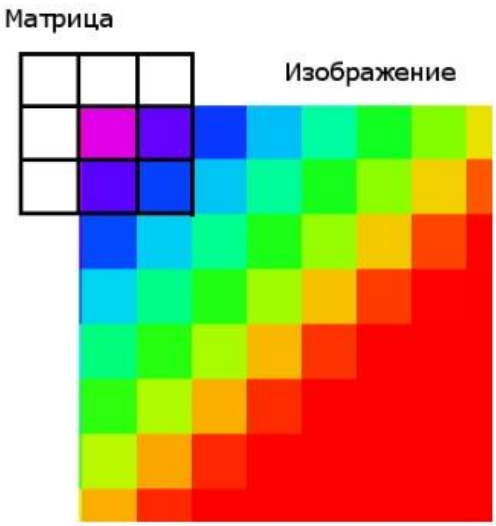


Рисунок 2.4 – Візуалізація матриці на зображенні

У центр зображення копіюється вхідна картинка, а краю заповнюються крайніми пікселями зображення (рисунок 2.5).

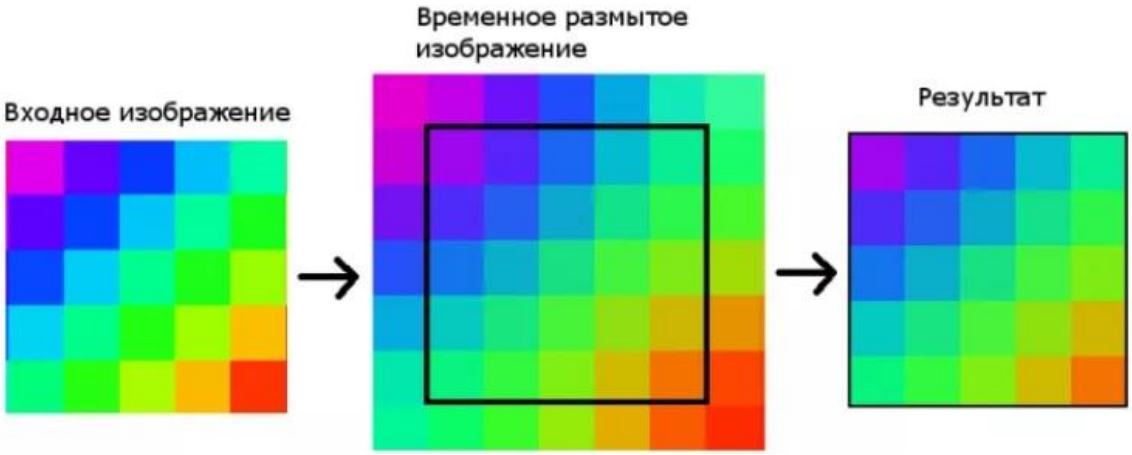


Рисунок 2.5 – Схематичне зображення роботи

2.1.3 Медіанний фільтр

Медіанний фільтр являє собою віконний фільтр, послідовно скануючий зображення, і повертає на кожному кроці один з елементів, що потрапили у вікно фільтра (рисунок 2.6).

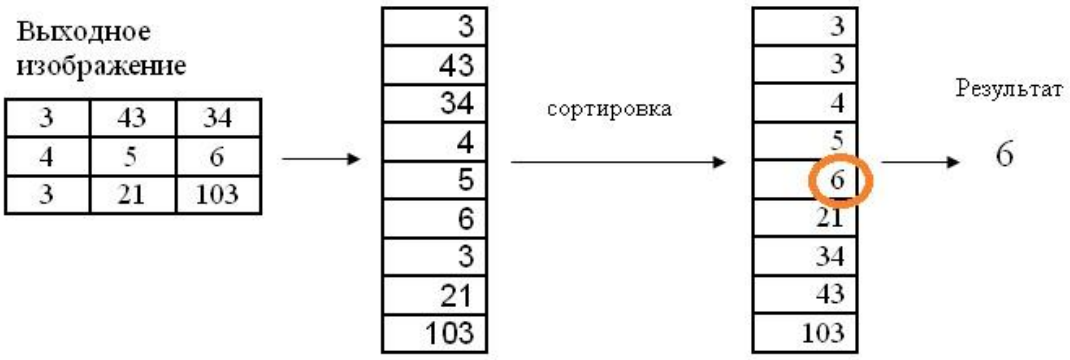


Рисунок 2.6 – Схематичне зображення медіанного фільтру

Пікселі, які «потрапляють» в вікно, сортуються в порядку зростання і вибирається те значення, яке знаходиться посередині відсортованого списку.

Медіанний фільтр зазвичай використовується для зменшення шуму або «згладжування» зображення (рисунок 2.7).

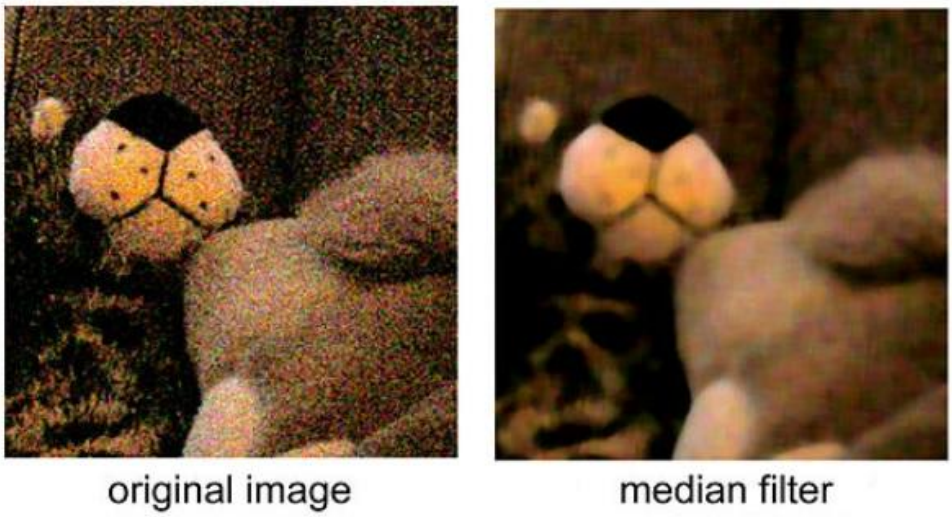


Рисунок 2.7 – Приклад роботи медіанного фільтру

Для поліпшення чіткості зображення використовується наступний фільтр (рисунок 2.8) (div = 1):

-1	-1	-1
-1	9	-1
-1	-1	-1

Рисунок 2.8 – Поліпшення чіткості

2.1.3 Морфологічні перетворення

Морфологічна фільтрація використовується для розширення (ділактації) або звуження (ерозії) елементів бінарного зображення (рисунок 2.9).

Дилатація (морфологічне розширення) – згортка зображення або виділеної області зображення деяким шаблоном. Шаблон може мати довільну форму і розмір. При цьому в ньому виділяється єдина провідна позиція (anchor), яка поєднується з поточним пикселем при обчисленні згортки.

Бінарне зображення – упорядкований набір (впорядкованої множини) чорно-білих точок (пікселів). Максимум інтенсивності пікселів зображення дорівнює одиниці, а мінімум - нулю.

Застосування дилатації зводиться до проходу шаблоном по всьому зображенню і застосування оператора пошуку локального максимуму інтенсивності пікселів зображення, які накриваються шаблоном. Якщо максимум дорівнює 1, то точка, в якій знаходиться анкор шаблону буде біла.

Така операція викликає зростання світлих областей на зображенні На малюнку сірим кольором відзначені пікселі, які в результаті застосування дилатації будуть білими.

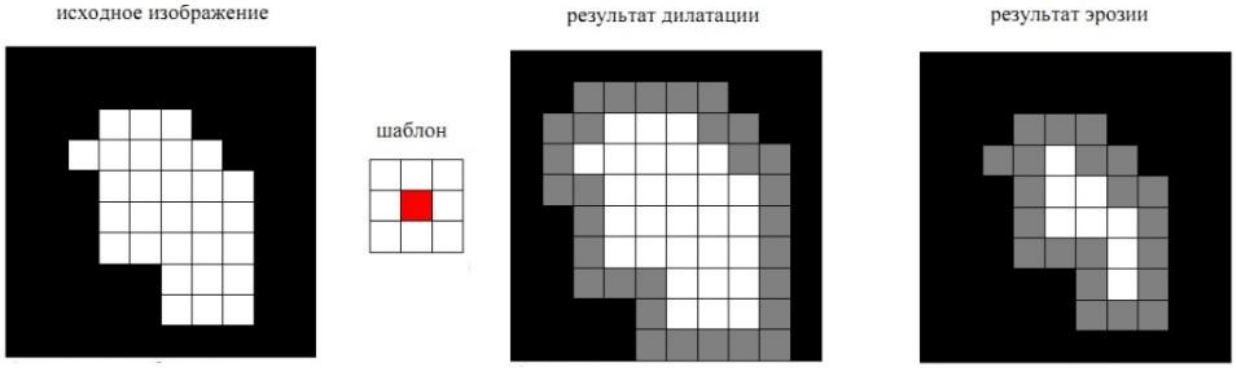


Рисунок 2.9 – Порівняння морфологічних перетворень

Ерозія (морфологічне звуження) – операція, зворотна дилатації. Дія ерозії, подібна дилатації, різниця лише в тому, що використовується оператор пошуку локального мінімуму. Якщо мінімум дорівнює 0, то точка, в якій знаходиться анкор шаблону, буде чорна. На малюнку праворуч сірим кольором відзначені пікселі, які стануть чорними в результаті ерозії.

Операція «Дилатація» – аналог логічного «або», операція «Ерозія» – аналог логічного «і».

Результат морфологічних операцій багато в чому визначається застосовуваним шаблоном (структурним елементом). Вибираючи різний структурний елемент можна вирішувати різні завдання обробки зображень:

- шумозаглушення;
- виділення меж об’єкта;
- нарощування точок області;
- виділення скелета об’єкта.

2.2 Корекція яскравості і контрасту зображень

Яскравість є характеристикою, що визначає те, на скільки сильно кольори пікселів відрізняються від чорного кольору. Наприклад, якщо оцифрована фотографія зроблена в сонячну погоду, то її яскравість буде значною. З іншого боку, якщо фотографія зроблена ввечері або вночі, то її яскравість буде невелика (рисунок 2.10).

Контраст є характеристикою того, наскільки великий розкид мають кольору пікселів зображення. Чим більший розкид мають значення кольорів пікселів, тим більший контраст має зображення.

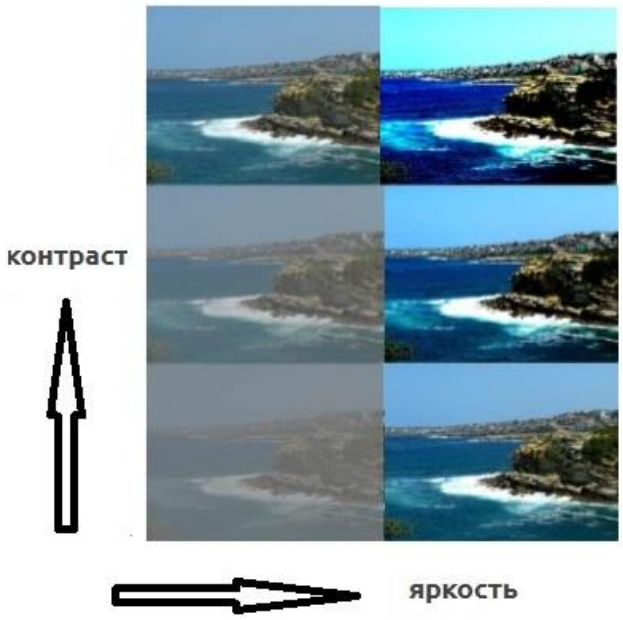


Рисунок 2.10 – Порівняння контрасту на яскравості

За аналогією з термінами математичної статистики можна відзначити, що:

- яскравість – середнє значення;
- контраст – середньоквадратичне відхилення від середнього значення.

Якщо яскравість і контраст зображення не змінюються в процесі перетворення, то функція має графік, представлений на (рисунку 2.11, а).

Рисунок 2.11 – Графіки контрасту та яскравості

Якщо пряма зсувається вгору (рисунок 2.11, б), яскравість зображення збільшується, а якщо пряма зсувається вниз (рисунок 2.11, в) – зменшується.

При збільшенні контрасту зображення (рисунок 2.12, а) нахил прямої збільшується, при зменшенні контрасту - зменшується (рисунок 2.12, б).

Рисунок 2.12 – Перетворення контрасту

Можна ставити різні перетворення яскравості / контрасту для кожного з компонентів моделі RGB.

2.2.1 Корекція контрастності зображення лінійної розтяжкою гістограми

Гістограма яскравості показує, скільки пікселів N з близьким значенням яскравості f потрапляє в інтервал від fi до f + Δfi.

Нехай, наприклад, рівні деякого зображення з відтінками сірого займають інтервал від 6 до 158 із середнім значенням яскравості 67 при можливому найбільшому інтервалі значень від 0 до 255. (рисунок 2.13)

Це зображення є малоконтрастним, превалює темний відтінок. Можливим шляхом поліпшення контрасту може стати так звана лінійна розтяжка гістограми (stretch), коли рівнями вихідного зображення, що лежить в інтервалі [fмін, Fмакс], присвоюються нові значення з тим, щоб охопити весь можливий інтервал зміни яскравості, в даному випадку [0, 255].

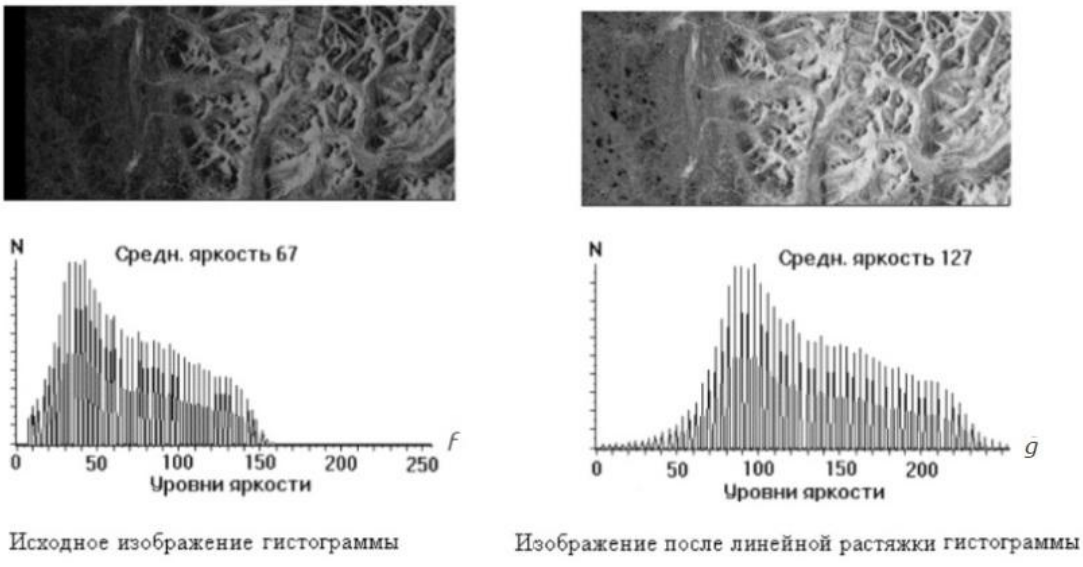


Рисунок 2.13 – Приклад зображення з яскравістю 67

При цьому контраст істотно збільшується (рисунок 2.13). Перетворення рівнів яскравості здійснюється за формулою:

$$g_i = a + bf_i, \qquad (2.2)$$

де f_i – старе значення яскравості і-го пікселя;

g_i – нове значення;

a, b - коефіцієнти.

Для рис. $f_{мін}$ = 6, Fмакс = 158. Виберемо a і b таким чином, щоб $g_{мін}$ = 0, $g_{макс}$ = 255. З (1) отримуємо: a = - 10,01; b = 1,67.

Ще можна більш поліпшити контраст, використовуючи нормалізацію гістограми. При цьому на весь максимальний інтервал рівнів яскравості [0, 255] розтягується не вся гістограма, що лежить в межах від fмін до Fмакс, а її найбільш інтенсивний ділянку. На наступному малюнку виключено 5% пікселів.

Метою вирівнювання гістограми (рисунок 2.14) (цю процедуру називають також лінеаризацією і еквалізацією – equalization) є таке перетворення, щоб, в ідеалі, всі рівні яскравості придбали б однакову частоту, а гістограма яскравості відповідала б рівномірному закону розподілу.

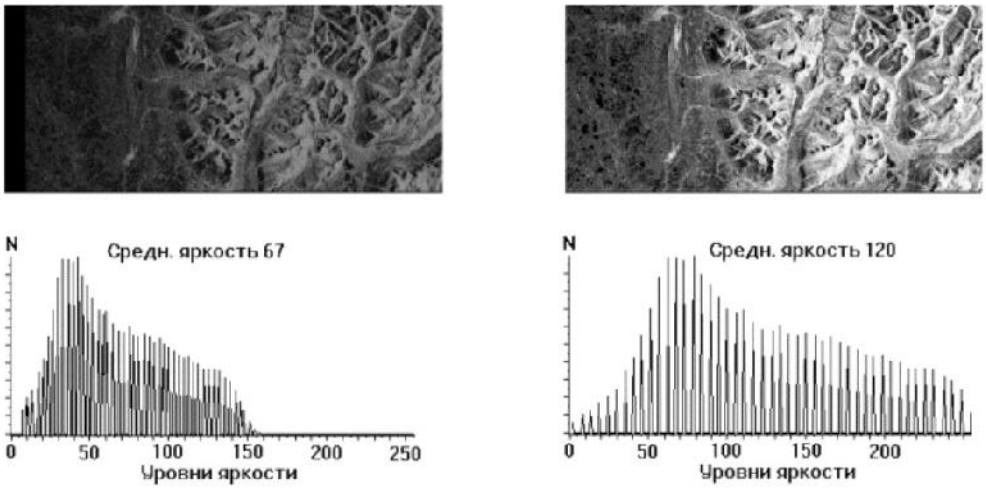


Рисунок 2.14 – Нормалізація гістограмми

Нехай зображення має формат: N пікселів по горизонталі і M по вертикалі, число рівнів квантування яскравості одно J. Загальне число пікселів одно N · M, на один рівень яскравості потрапляє, в середньому, no =

N · M / J пікселів.

Наприклад, N = M = 512, J = 256. У цьому випадку no = 1024. Відстань Δf між дискретними рівнями яскравості від fi до fi + 1 в гістограмі вихідного зображення однакове, але на кожен рівень випадає різне число пікселів. При еквалізації гістограми відстань Δgi між рівнями gi і gi + 1 різному, але число пікселів на кожному рівні, в середньому, однакове і дорівнює no. Алгоритм еквалізациі нескладний. Нехай рівнями з малою яскравістю володіє невелика кількість пікселів, як на рисунку 2.14.

Наприклад, рівень яскравості 0 на оригінальному документі мають 188 пікселів, рівень 1 - 347 пікселів, рівень 2 - 544 пікселя. В сумі це тисяча сімдесят дев’ять пікселів, тобто приблизно no. Дамо всім цим пикселам рівень 0. Нехай на оригінальному документі число пікселів з рівнями яскравості 3 і 4 в сумі приблизно також одно no. Цим пикселям присвоюється рівень 1. З іншого боку, нехай число пікселів з рівнем 45 на оригінальному документі становить 3012, тобто приблизно 3no. Всім цим пикселам присвоюється певний однаковий рівень gi, не обов’язково рівний 45, а сусідні два рівня залишаються незаповненими. Розглянуті процедури виконуються для всіх рівнів яскравості.

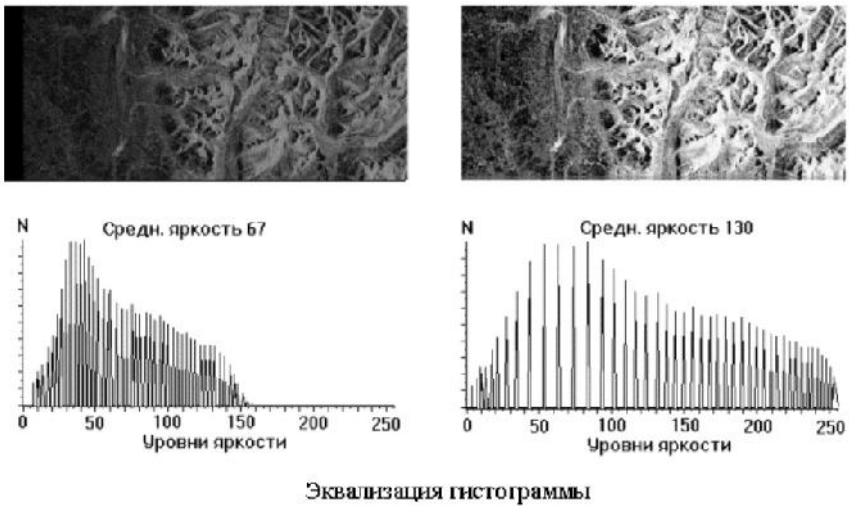


Рисунок 2.15 – Еквілізація гістограмми

2.3 Огляд моделей шумів. Зашумлення зображення

Часто під фільтрацією в більш вузькому сенсі розуміють так звану помехову фільтрацію, або фільтрацію зображень від шуму. При цьому неявно передбачається, що спочатку десь існувало якесь вихідне ідеально чисте (незашумлене) зображення, з якого потім шляхом зашумлення (певного виду спотворення), було отримано то реальне зображення, яке ми спостерігаємо. У підсумку завдання помехової фільтрації зводиться до того, щоб шляхом деякої обробки спостережуваного реального зображення якнайкраще очистити його від шуму, тобто отримати зображення, найбільш близьке за своїми характеристиками до вихідного незашумленого зображення.

Необхідно розуміти, що зашумлення – це всього лише дуже спрощена ідеалізована модель виникнення спотворень в цифрових зображеннях реальних об’єктів. Спотворення зображення, одержуваного шляхом відеозйомки реального тривимірного об’єкту в природній обстановці, можуть носити досить складний характер, що залежить від умов зйомки (освітленість, туман, відблиски, тіні, дощ, сніг і т.п.), характеристик оптичної системи (дисторсії, расфокусировки, замутненість лінз і дзеркал і т.п.), характеристик електронної реєструючої апаратури, характеристик каналу передачі, характеристик пристроїв оцифровки і ще багатьох і багатьох чинників. Наближені до реальності математичні моделі формування цифрових зображень містять сотні складних нелінійних рівнянь і безліч табличних поправок. При цьому закон формування значення яскравості кожного пікселя зображення, як правило, не є незалежним від формування сусідніх пікселів.

При спробі математично скоригувати подібну складну модель реєстрації зображення говорять вже не про фільтрацію від шуму, а про реставрацію або реконструкції зображень. Найбільш загальною моделлю незалежного зашумлення пікселів є шум заміщення. Нехай дано вихідне (незашумлене) півтонування I_m [x, y], кожен піксель якого може приймати

значення в діапазоні [0, ..., I_max-1].

Загальна модель шуму заміщення передбачає, що після зашумлення кожен піксель зображення, що мав раніше значення яскравості i, або з деякою відомою ймовірністю p (i) це значення збереже, або дане значення яскравості буде випадковим чином заміщено з ймовірністю q (i, j) деяким іншим значенням яскравості j з того ж кінцевого дискретного діапазону [0, ..., I_max-1].

Як видно, для опису такої загальної моделі випадкового заміщення необхідно задати таблицю перехідних ймовірностей розміру I_max2, що становить досить значну кількість в разі звичайного 8-бітового півтонування (розмір таблиці - 256 × 256 елементів). Такий опис явно є некомпактним і тому рідко використовується на практиці для напівтонових зображень. Для напівтонових зображень, як правило, розглядають іншу, більш приватну модель зашумлення – адитивний шум, яка передбачає, що зашумлене зображення породжується за законом:

$$Im'[x, y] = Im[x, y] + R(x, y), \quad (2.3)$$

де Im’[x, y] – піксель зашумленного зображення;

Im [x, y] – піксель вихідного зображення;

а R (x, y) – випадкова адитивна компонента.

Крім того, в більшості додатків залежність шуму від координат пікселя вважається несуттєвою. Нарешті, виходячи з відомого в статистиці закону великих чисел, закон розподілу адитивного шумового компоненти воліють описувати зручним параметричних сімейством нормальних (гауссовских) розподілів з нульовим середнім. Таким чином, гауссовский адитивний шум описується виразом:

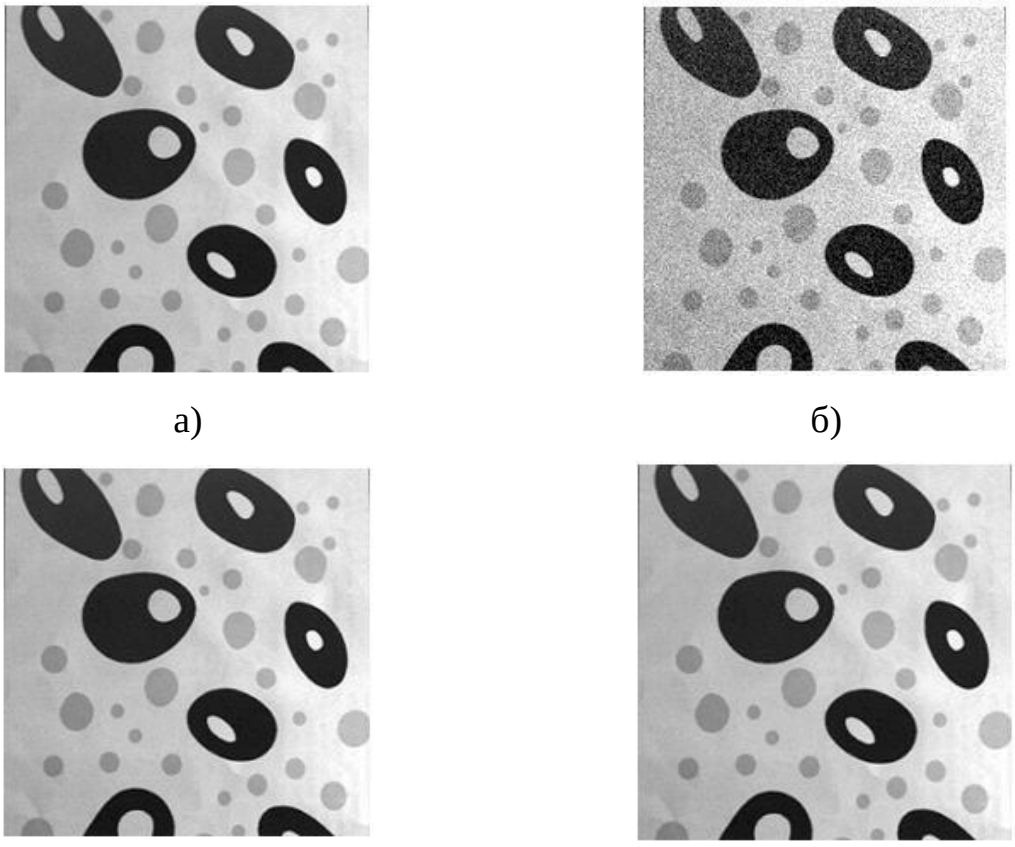
$$Im'[x, y] = Im[x, y] + N(0, \sigma),$$

де N (a, σ) - нормальний розподіл;

a – математичне сподівання нормально розподіленого сигналу;

σ – середній квадрат відхилення (СКВ) нормально розподіленої величини.

Саме така модель зашумлення найчастіше розглядається в задачах фільтрації напівтонових зображень (рисунок 2.16).



а) б)

в) г)

Рисунок 2.16 – Фільтрація напівтонових зображень: а) Початкове півтонування, σ=0; б) Зашумлення півтонування аддитивним гауссовским шумом, σ=20; в) Зашумлення півтонування аддитивним гауссовским шумом, σ=40; г) Зашумлення півтонування аддитивним гауссовским шумом, σ=60

3 ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ

Робота згорткової нейронної мережі забезпечується двома основними елементами:

- фільтри (filters) (визначники ознак);
- карти ознак (feature maps).

Фільтр – це невелика матриця, що дає ознаку, яку необхідно знайти на вихідному зображенні. За допомогою верхнього фільтру визначаються частини вихідного зображення з вертикальними лініями, нижній фільтр служить для визначення частин зображення з горизонтальними лініями.

Безпосередньо процес визначення заснований на операції згортки фільтром (визначники ознак) оригінального зображення. Результати згортки, які визначають місце розташування ознак вихідного зображення, називаються картами ознак.

Мета процесу згортки – зменшити розмірність карти ознак (feature maps) до такої міри, щоб з повним набором ознак могла працювати мережа прямого поширення (в більшості випадків багатошаровий персептрон).

Згортковий шар реалізує ідею локальних рецептивних полів, тобто кожен вихідний нейрон з’єднаний тільки з певною (невеликою) областю вхідної матриці і таким чином моделює деякі особливості людського зору.

Недоліками згорткових нейронних мереж (ЗНМ) є:

- висока складність архітектури;
- повнозв’язаність;
- фіксована площа вікна шару згортки.

З метою підвищення ефективності роботи ЗНМ необхідно знайти оптимальні значення наступних параметрів:

- кількість карт ознак;
- щільність зв’язків між картами ознак;
- розмір вікна;

- площа перекриття;
- початкова ініціалізація ваг.

3.1 Архітектура згорткової нейронної мережі

У звичайному перцептрону, який представляє собою повнозв’язну нейронну мережу, кожен нейрон пов’язаний з усіма нейронами попереднього шару, причому кожна зв’язок має свій персональний ваговий коефіцієнт. У вій нейронній мережі в операції згортки використовується лише обмежена матриця ваг невеликого розміру, яку «рухають» по всьому оброблюваного шару (на самому початку – безпосередньо по вхідному зображенню), формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічною позицією. Тобто для різних нейронів вихідного шару використовуються загальні ваги – матриця ваг, яку також називають набором ваг або ядром згортки.

Вона побудована таким чином, що графічно кодує будь-якої один ознака, наприклад, наявність похилої лінії під певним кутом. Тоді наступний шар, що вийшов в результаті операції згортки такою матрицею ваг, показує наявність даної похилій лінії в оброблюваному шарі і її координати, формуючи так звану карту ознак. Природно, в свёрточной нейронної мережі набір ваг не один, а ціла гама, що кодує всілякі лінії і дуги під різними кутами. При цьому такі ядра згортки не закладаються дослідником заздалегідь, а формуються самостійно шляхом навчання мережі класичним методом поширення помилки. Прохід кожним набором ваг формує свій власний примірник карти ознак, роблячи нейронну мережу багатовимірної (багато незалежних карт ознак на одному шарі). Також слід зазначити, що при переборі шару матрицею ваг її пересувають зазвичай не на повний крок (розмір цієї матриці), а на невелику відстань. Так, наприклад, при розмірності матриці ваг 5 × 5 її зрушують на один або два нейрона (пікселя) замість п’яти, щоб не «переступити» шуканий ознака.

Операція Субдіскретізація (англ. Subsampling, англ. Pooling, також перекладається як «операція підвибірки» або операція об’єднання), виконує зменшення розмірності сформованих карт ознак. У даній архітектурі мережі вважається, що інформація про факт наявності шуканого ознаки важливіше точного знання його координат, тому з кількох сусідніх нейронів карти ознак вибирається максимальний і приймається за один нейрон карти ознак зменшеною розмірності. Також іноді застосовують операцію знаходження середнього між сусідніми нейронами. За рахунок цієї операції, крім прискорення подальших обчислень, мережа стає більш інваріантної до масштабу вхідного зображення.

Таким чином, повторюючи один за одним кілька шарів згортки і Субдіскретізація будується свёрточная нейронна мережа. Чергування шарів дозволяє складати карти ознак з карт ознак, що на практиці означає здатність розпізнавання складних ієрархій ознак. Зазвичай після проходження декількох шарів карта ознак вироджується в вектор або навіть скаляр, але таких карт ознак стає сотні. На виході мережі часто додатково встановлюють кілька шарів повно нейронної мережі (перцептрон), на вхід якого подаються кінцеві карти ознак.

Якщо на першому шарі ядро згортки проходить тільки по одному вихідному зображенню, то на внутрішніх шарах одне і те ж ядро проходить паралельно по всіх картах ознак цього шару, а результат згортки підсумовується, формуючи (після проходження функції активації) одну карту ознак наступного шару, відповідну цього ядра згортки.

Модель вої мережі (рисунок 3.1), яку розглянемо, складається з трьох типів шарів: свёрточние шари, субдіскретізірующіе (subsampling, підвибірка) шари і шари "звичайної" нейронної мережі – перцептрона.

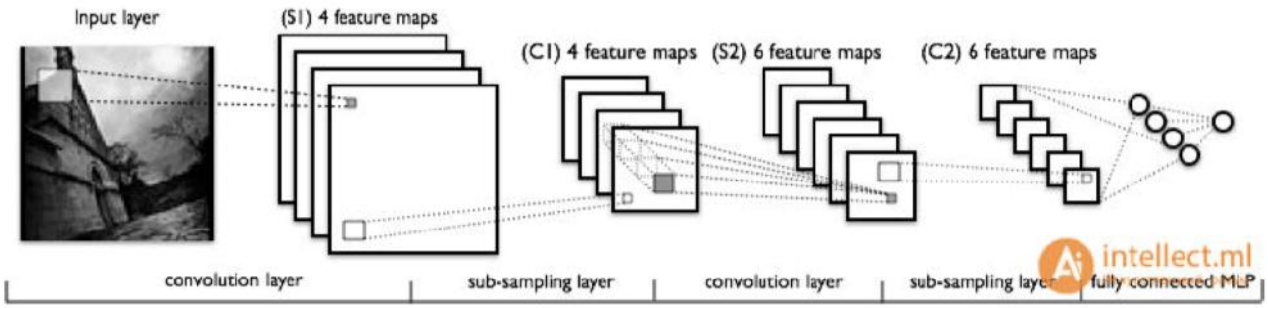


Рисунок 3.1 – Структура згорткової мережі

Перші два типи шарів (convolutional, subsampling), чергуючись між собою, формують вхідний вектор ознак для багатошарового перцептрону. Мережу можна навчати за допомогою градієнтних методів. Свою назву згорткова мережа отримала за назвою операції – згортка, вона часто використовується для обробки зображень і може бути описана наступною формулою:

$$(f \times g)\,[m,n] = \sum_{k,l} f\,[m-k,\qquad n\text{-}l] \times g[k,l], \tag{3.1}$$

де f – вихідна матриця зображення;

g – ядро (матриця) згортки.

Неформально цю операцію можна описати таким чином – вікном розміру ядра g проходимо з заданим кроком (зазвичай 1) все зображення f, на кожному кроці поелементно множимо вміст вікна на ядро g, результат підсумовується і записується в матрицю результату.

При цьому в залежності від методу обробки країв вихідної матриці результат може бути менше вихідного зображення (рисунок – 3.2), такого ж розміру (рисунок – 3.3) або більшого розміру (рисунок – 3.4).

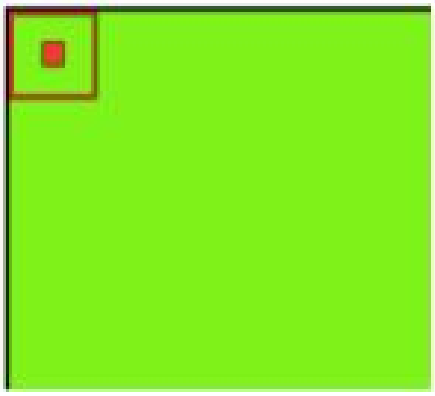


Рисунок 3.2 – Обробка країв valid

Рисунок 3.3 – Обробка країв same

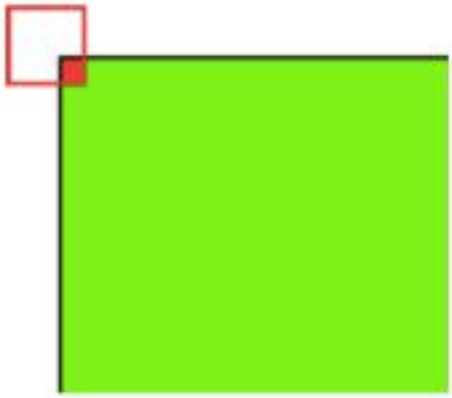


Рисунок 3.4 – Обробка країв full

3.2 Згорткові нейронні мережі

Ідея згорткових нейронних мереж полягає в чергуванні згорткових шарів (C-layers), субдискретизуючіх шарів (S-layers) і наявності повнозв’язних (F-layers) шарів на виході.

Така архітектура містить в собі 3 основних парадигми:

- локальне сприйняття;
- спільні ваги;
- субдіскретізація.

Локальне сприйняття має на увазі, що на вхід одного нейрона подається не все зображення (або виходи попереднього шару), а лише деяка його область. Такий підхід дозволив зберігати топологію зображення від шару до шару.

Концепція поділюваних ваг припускає, що для великої кількості зв’язків використовується дуже невеликий набір ваг. Тобто якщо у нас є на вході зображення розмірами 32x32 пікселя, то кожен з нейронів наступного шару візьме на вхід тільки невелику ділянку цього зображення розміром, наприклад, 5x5, причому кожен з фрагментів буде оброблений одним і тим же набором. Важливо розуміти, що самих наборів ваг може бути багато, але кожен з них буде застосований до всього зображення. Такі набори часто називають ядрами (kernels).

Фільтр являє собою матрицю коефіцієнтів, звичайно задану вручну. Ця матриця застосовується до зображення за допомогою математичної операції, званої зготкою. Суть цієї операції в тому, що кожен фрагмент зображення множиться на матрицю (ядро) згортки поелементно і результат підсумовується і записується в аналогічну позицію вихідного зображення. Основна властивість таких фільтрів полягає в тому, що значення їх виходу тим більше чим більше фрагмент зображення схожий на сам фільтр. Таким чином зображення згорнуте з якимось ядром дасть нам інше зображення, кожен піксель якого буде означати ступінь схожості фрагмента зображення

на фільтр. Іншими словами це буде карта ознак.

Кожен фрагмент зображення поелементно множиться на невелику матрицю ваг (ядро), результат підсумовується. Ця сума є пикселем вихідного зображення, яке називається картою ознак. Слід сказати, що в ідеалі не різні фрагменти проходять послідовно через ядро, а паралельно все зображення проходить через ідентичні ядра. Крім того, кількість ядер (наборів ваг) визначається розробником і залежить від того скільки ознак необхідно виділити. Ще одна особливість згорткового шару в тому, що він трохи зменшує зображення за рахунок крайових ефектів.

Суть Субдіскретизації і S-шарів полягає в зменшенні просторової розмірності зображення. Тобто вхідне зображення грубо (усередненням) зменшується в задану кількість разів. Найчастіше в 2 рази, хоча може бути і не рівномірний зміна, наприклад, 2 по вертикалі і 3 по горизонталі. Субдіскретизація потрібна для забезпечення інваріантності до масштабу.

Чергування шарів дозволяє складати карти ознак з карт ознак, що на практиці означає здатність розпізнавання складних ієрархій ознак.

Зазвичай після проходження декількох шарів карта ознак вироджується в вектор або навіть скаляр, але таких карт ознак стає сотні. У такому вигляді вони подаються на один-два шари повно-мережі. Вихідний шар такої мережі може мати різні функції активації. У найпростішому випадку це може бути тангенціальна функція, також успішно використовуються радіальні базисні функції.

3.2.1 Структура згорткової нейронної мережі

У згортковій нейронній мережі виходи проміжних шарів утворюють матрицю (зображення) або набір матриць (кілька шарів зображення). Так, наприклад, на вхід згорткової нейронної мережі можна подавати три шари зображення (R-, G-, B-канали зображення). Основними видами шарів в ЗНМ є згорткові шари (англ. Convolutional layer), пулінгові шари (англ. Pooling

layer) і повнозв’язні шари (англ. Fully-connected layer).

3.2.2 Згортковий шар

Згортковий шар (англ. Convolutional layer) є основним будівельним блоком ОНР. Параметри шару складаються з набору фільтрів для навчання (або ядер), які мають невеликі рецептивні поля, але простягаються на всю глибину вхідного об’єму. Протягом прямого проходу кожен фільтр здійснює згортку по ширині і висоті вхідного об’єму, обчислюючи скалярний твір даних фільтра і входу, і формуючи двовимірну мірну карту активації цього фільтру. У результаті мережа вчиться, які фільтри активуються, коли вони бачать певний конкретний тип ознаки в певному просторовому положенні при вході.

Складання активаційних карт всіх фільтрів уздовж вимірювання глибини формує повний вихідний об’єм згорткового шару. Таким чином, кожен запис у вихідному об’ємі може також трактуватися як вихід нейрона, який дивиться на невелику область на вході, і ділить параметри з нейронами тієї ж активіаційної карти.

Згортковий шар реалізує ідею – локальних рецептивних полів (рисунок 3.5), тобто кожен вихідний нейрон з’єднаний тільки з певною (невеликою) областю вхідної матриці і таким чином моделює деякі особливості людського зору.

У спрощеному вигляді цей шар можна описати формулою:

$$x^l = f\ (x^{l-1} * k^l + \ b^l), \tag{3.2}$$

де x^l – вихід шару l;

f () – функція активації;

b – коефіцієнт зсуву, символом * позначена операція згортки входу x з

ядром k.

При цьому за рахунок крайових ефектів розмір вихідних матриць зменшується за формулою:

$$x_j^l = f(\sum_i x_i^{l-1} * k_j^l + b_j^l) \quad (3.3)$$

Тут x^l_j – карта ознак j (вихід шару l);

f () – функція активації;

b_j – коефіцієнт зсуву для карти ознак j;

k_j – ядро згортки номер j;

x^{l-1} – карти ознак попереднього шару.

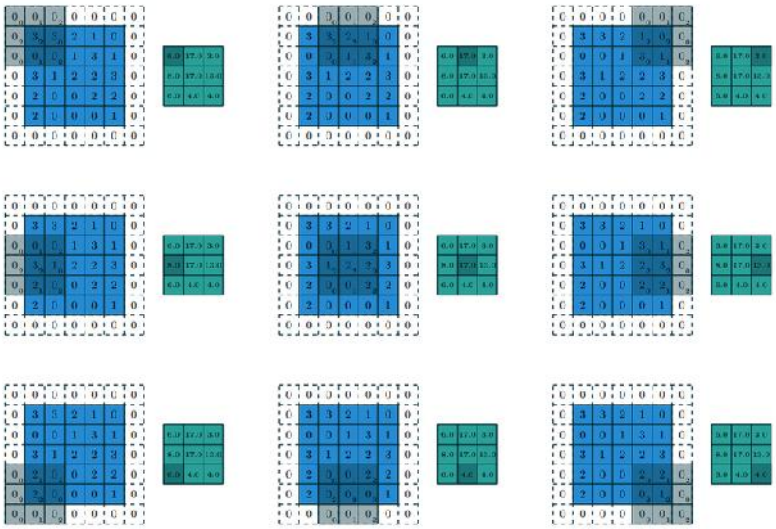


Рисунок 3.5 – Приклад згортки двох матриць з доповненням нулями і зрушенням 2

3.2.2 Пулінговий шар

Пулінговий шар покликаний знижувати розмірність зображення. Початкове зображення поділяється на блоки розміром w * h і для кожного

блоку обчислюється деяка функція. Найчастіше використовується функція максимуму (англ. Max pooling) або (зваженого) середнього (англ. (Weighted) average pooling). Учнів параметрів у цього шару немає. Основні цілі пулінгового шару:

- зменшення зображення, щоб наступні пакунки оперували над більшою областю початкового зображення;
- збільшення інваріантності виходу мережі по відношенню до малого переносу входу;
- прискорення обчислень.

3.2.3 Повнозв’язний шар

Шари цього типу (рисунок 3.6) виконують зменшення розмірності (зазвичай в кілька разів). Це можна робити різними способами, але найчастіше використовується метод вибору максимальному елементу (max-pooling) – вся карта ознак поділяється на осередки, з яких вибираються максимальні за значенням.

Dropout шар (dropout регуляризація) [14] – спосіб боротьби з перенавчанням в нейронних мережах, навчання яких зазвичай виробляють стохастичним градієнтним спуском, випадково вибираючи деякі об’єкти з вибірки. Dropout регуляризація полягає в зміна структури мережі: кожен нейрон викидається з певною ймовірністю p.

За такою прорідженості мережі проводиться навчання, для решти ваг робиться градієнтний крок, після чого всі викинуті нейрони повертаються в нейромережу. Таким чином, на кожному кроці стохастичного градієнтного спуску ми налаштовуємо одну з можливих 2N архітектур мережі, де під архітектурою ми розуміємо структуру зв’язків між нейронами, а через N позначаємо сумарне число нейронів.

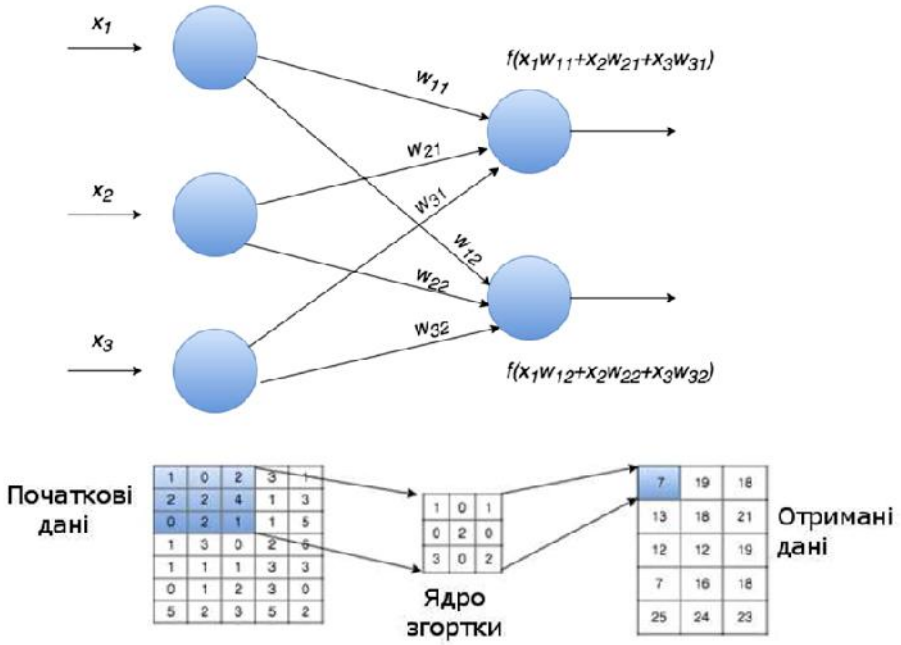


Рисунок 3.6 – Повнозв’язний шар

При тестуванні нейромережі нейрони вже не викидаються, але вихід кожного нейрона множиться на (1 – p) – завдяки цьому на виході нейрона ми будемо отримувати маточікування його відповіді по всіх 2N архітектурах. Таким чином, навчену за допомогою dropout-регуляриза- ції нейромережу можна розглядати як результат усереднення 2N мереж.

3.2.4 Inception module

Inception module - це спеціальний шар нейронної мережі, який був запропонований в роботі, в якій була представлена мережа GoogLeNet.

Основна мета цього модуля полягає в наступному. Автори припустили, що кожен елемент попереднього шару відповідає певній галузі вихідного зображення. Кожна згортка по таким елементам буде збільшувати область вихідного зображення, поки елементи на останніх шарах не будуть відповідати всьому зображенню цілком. Однак, якщо з якогось моменту все згортки стануть розміром 1 × 1, то чи не знайдеться елементів, які покривали

б все вихідне зображення, тому було б неможливо знаходити великі ознаки на зображенні. Щоб вирішити цю проблему, автори запропонували так званий inception module - конкатенацію виходів для згорток розміру 1 × 1, 3 × 3, 5 × 5, а також операції max pooling’а з ядром 3 × 3. На жаль, подібний наївний підхід (англ. Naive inception module) призводить до різкого збільшення шарів зображення, що не дозволяє побудувати з його використанням глибоку нейронну мережу. Для цього автори запропонували використовувати модифікований inception module з додатковим зменшенням розмірності - додатково до кожного фільтру вони додали шар згортки 1 × 1, який схлопивается всі верстви зображення в один. Це дозволяє зберегти мале число шарів, зі збереженням корисної інформації про зображення.

4 ПРАКТИЧНА ЧАСТИНА

4.1 Програмні засоби розпізнавання зображень на основі нейронних мереж

Нейромережеве програмне забезпечення використовується для моделювання, дослідження, розробки і застосування штучних нейронних мереж, штучного інтелекту і машинного навчання.

Існує багато програмних засобів для розпізнавання зображень на основі нейронних мереж. Одним з найпоширеніших серед них є Matlab.

Matlab – одна з сучасних систем автоматизації математичних розрахунків, яка побудована на розширеному представленні й застосуванні матричних операцій. Це знайшло відображення в назві системи – MATrix LABoratory – матрична лабораторія [28]. Однак синтаксис мови програмування системи продуманий настільки ретельно, що ця орієнтація майже не відчувається тими користувачами, яких не цікавлять безпосередньо матричні обчислення.

Матриці широко застосовуються в складних математичних розрахунках, наприклад, при рішенні задач лінійної алгебри і математичного моделювання статичних і динамічних систем і об’єктів. Вони є основою автоматичного складання й рішення рівнянь стану динамічних об’єктів і систем. Прикладом може служити розширення Matlab – Simulink [17]. Це істотно підвищує інтерес до системи Matlab, що увібрала в себе кращі досягнення в області швидкого рішення матричних задач.

Однак сьогодні Matlab далеко вийшла за межі спеціалізованої матричної системи і стала однією з найбільш могутніх універсальних інтегрованих обчислювальних систем.

В цілому, Matlab – це унікальна колекція реалізацій сучасних чисельних методів комп’ютерної математики, створених за останні три

десятки років. Вона увібрала в себе і досвід, правила і методи математичних обчислень, які були накопичені за тисячі років розвитку математики. Це сполучається з могутніми засобами графічної візуалізації і навіть анімаційної графіки. Систему з прикладеною до неї об’ємною документацією цілком можна розглядати як фундаментальний багатотомний електронний довідник по математичному забезпеченню комп’ютерів – від масових персональних комп’ютерів до суперкомп’ютерів [29].

Для роботи з нейронними мережами у Matlab створено Neural Network Toolbox. Neural Network Toolbox забезпечує всебічну підтримку проектування, навчання та моделювання багатьох відомих мережевих парадигм, від базових моделей персептрона до найсучасніших асоціативних і самоорганізованих мереж [30].

На рисунку 4.1 наведено приклад вікна у Neural Network Toolbox.

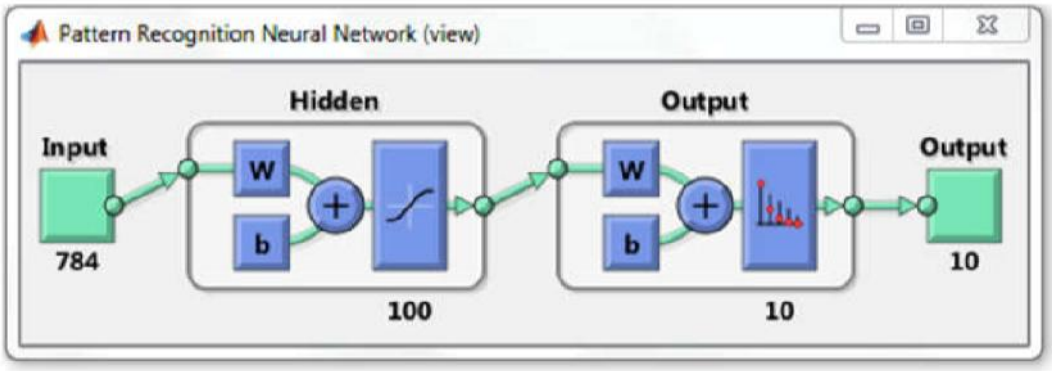


Рисунок 4.1 – Приклад вікна

Пакет фірми "The MathWorks" MATLAB також надає користувачам можливість роботи з нейронними мережами. Включений в стандартну комплектацію MATLAB набір інструментів надає широкі можливості для роботи з нейронними мережами всіх типів. Використання "Neural Network Toolbox" спільно з іншими засобами MATLAB відкриває широкий простір для ефективного комплексного використання сучасних математичних методів для вирішення найрізноманітніших завдань прикладного та

наукового характеру [31]. В даний час доступна версія 4.0 "Neural Network Toolbox", яку поставляють в складі MATLAB 6.0.

Peltarion Synapse – це компонентно-орієнтоване середовище розробки для нейронних мереж і адаптивних систем. Даний програмний комплекс дозволяє проводити дослідження у таких напрямках як data mining, статистичний аналіз, візуалізація, попереднє оброблення зображень, створення та навчання нейронних мереж та адаптивних систем [32].

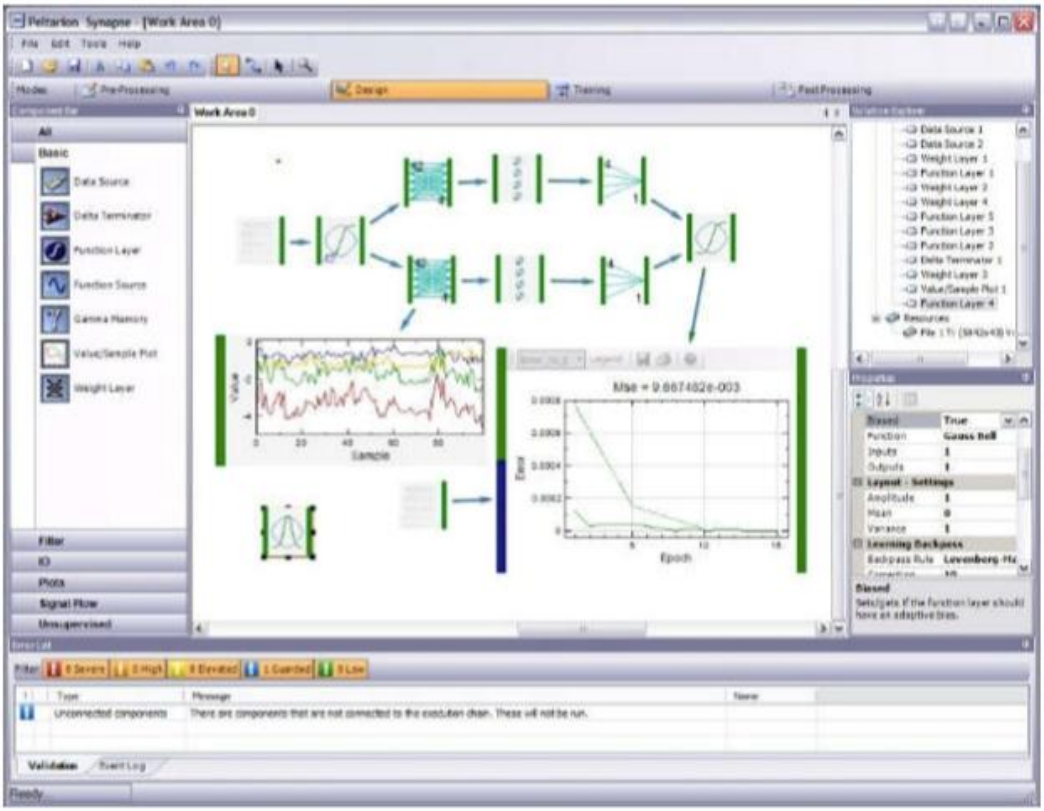


Рисунок 4.2 – Приклад вікна в ПЗ Peltarion Synapse

Цикл розробки в Synapse базується на основі канонічного циклу інтелектуального аналізу даних.

NeuroSolutions – це середовище розробки нейронних мереж, розроблене компанією NeuroDimension. Воно поєднує в собі модульний, компонентно- орієнтований інтерфейс проектування нейронних мереж [33].

На рисунку 4.3 наведено приклад процесу проектування нейронної

мережі у середовищі NeuroSolutions.

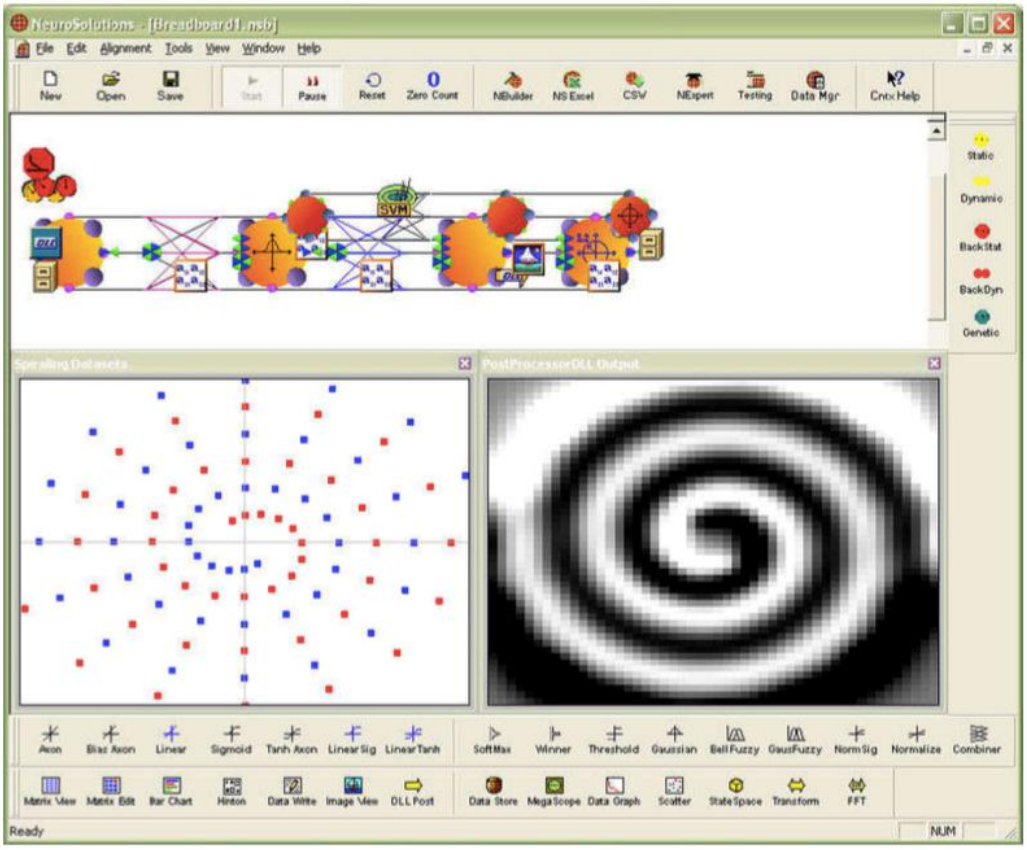


Рисунок 4.3 – Процес проектування нейронної мережі у середовищі NeuroSolutions

Вище наведене програмне забезпечення використовується для проектування, навчання і розгортання моделей нейронних мереж (навчання з вчителем і без вчителя) для виконання широкого спектру завдань, таких як класифікація, інтелектуальний аналіз даних, апроксимація функцій, множинна регресія та прогнозування часових рядів.

Deeplearning4j (DL4J) – це перша у своєму роді бібліотека з відкритим кодом (Open-source), яка використовується для машинного навчання. Дана бібліотека розроблена на мові програмування Java та Scala [35]. Крім цього, Deeplearning4j підтримує обчислення як на центральному процесорі (CPU), так і на графічному адаптері (GPU). Використання графічного адаптера

дозволяє реалізувати розпаралелення виконання обчислень, що значно збільшує швидкодію та зменшує тривалість цих обчислень.

DL4J застосовується для виявлення шахрайства, рекомендаційних систем (CRM, Adtech), регресії і прогностичного аналізу, розпізнавання зображеннь та образів, пошук голосу, мови в текст (транскрипції), і планово-попереджувального апаратного моніторингу (виявлення аномалій).

Deeplearning4j включає в себе як розподілене, багато-потокове машинне навчання, так і нормальне однопоточне машинне навчання. Навчання проходить в кластері, що означає, що він може обробляти великі обсяги даних максимально швидко. Мережі проходять навчання паралельно з допомогою ітеративного зменшення, і вони в рівній мірі сумісні з Java, Scala і Clojure. Роль Deeplearning4j як модульного компонента у відкритому стеку робить перший каркас глибинного навчання адаптована для архітектури мікро-послуг [34].

У Deeplearning4j включені засоби захисту від зовнішнього втручання і безпеки від злому, що особливо важливо в фінансових задачах, в промислових системах, в електронній комерції та підприємництві, де застосовується розпізнавання аномалій і розпізнавання образів. Deeplearning4j інтегрується з іншими платформами машинного навчання - такими як RapidMiner, Prediction.io і Weka.

4.2 Реалізація згорткової нейронної мережі для обробки та розпізнавання тексту

Розпізнавання зображень знаходить широке застосування в різних областях: це може бути класифікація об’єктів, розпізнавання осіб, символів, маркування об’єктів, автомобільних номерних знаків, відбитків пальців і т.д. На сьогоднішній день дані технології застосовуються в мобільних телефонах, планшетних комп’ютерах, системах контролю доступу до секретних об’єктах, оперативному пошуку в картотеці і т.д. Однак до сих пір при вирішенні задач

в цій області виникає ряд складних наукових і технічних проблем.

У зв’язку з цим можна припустити, що в найближчому майбутньому будуть актуальні завдання ідентифікації і розпізнавання рукописних підписів, тому що підпис кожної людини може бути важливим ідентифікатором особистості.

Рукописні підписи мають досить складну структуру і дрібну деталізацію, все це становить велику складність для вирішення даного завдання математичними методами і вимагає великих обчислювальних витрат.

Для початку необхідно розглянути задачу розпізнавання рукописних цифр. Завдання це хороша і цікава з наступних причин:

- для розпізнавання рукописного символу досить важко скласти формалізований (не інтелектуальне) алгоритм і це стає зрозуміло, варто лише поглянути на одну й ту ж саму цифру написану різними людьми;
- завдання досить актуальна і має відношення до OCR (optical character recognition);
- існує вільно поширювана база рукописних символів, доступна для скачування і експериментів.

Окрім цього обробка та розпізнавання зображень з рукописними цифрами є обгрунтованним для подальшого розширення системи з метою удосконалення для розпізнавання української та англійської мови.

В якості вхідних даних було прийнято рішення використовувати базу даних відкритого типу MNIST. Ця база містить 60 000 навчальних пар (зображення - мітка) і 10 000 тестових (зображення без міток). Зображення нормалізовані за єдиним розміром і отцентровани. Розмір кожної цифри не більше 20x20, але вписані вони в квадрат розміром 28x28. Однак варто зазначити, що представлені як великі варіанти шрифтів так і більш мініатюрні. Приклад перших 12 цифр з навчального набору бази MNIST наведено на рисунку 4.4.

Рисунок 4.4 – Приклад даних з бази загального доступу MNIST

Таким чином завдання було сформульовано так: створити і навчити нейромережа для розпізнавання рукописних символів, приймаючи їх зображення на вході і активуючи один з 10 виходів. Під активацією будемо розуміти значення 1 на виході. Значення інших виходів при цьому повинні (в ідеалі) бути рівні -1.

4.4 Навчання

Для того щоб можна було почати навчання нашої мережі потрібно визначитися з тим, як вимірювати якість розпізнавання. У нашому випадку для цього будемо використовувати найпоширенішу в теорії нейронних мереж функцію середньоквадратичної помилки (СКО, MSE) [3]:

$$E^p = \frac{1}{2}(D^p - O(I^p, W))^2 \tag{4.1}$$

У цій формулі Ep - це помилка розпізнавання для p-ой навчальної пари;

Dp - бажаний вихід мережі;

O (Ip, W) – вихід мережі, що залежить від p-го входу і вагових коефіцієнтів W, куди входять ядра згортки, зміщення, вагові коефіцієнти S- і F- шарів.

Завдання навчання так налаштувати ваги W, щоб вони для будь-якої навчальної пари (Ip, Dp) давали мінімальну помилку Ep. Щоб порахувати помилку для всієї навчальної вибірки просто береться середнє арифметичне

по помилках для всіх навчальних пар. Таку усереднену помилку позначимо як E.

Для мінімізації функції помилки Ep найефективнішими є градієнтні методи. Розглянемо суть градієнтних методів на прикладі найпростішого одновимірного випадку (тобто коли у нас всього один вага). Якщо ми розкладемо в ряд Тейлора функцію помилки Ep, то отримаємо такий вираз:

$$E(W) = E(W_c) + (W - W_c)\frac{dE(W_c)}{dW} + \frac{1}{2}(W - W_c)^2\frac{d^2E(W_c)}{dW^2} + \dots \quad (4.2)$$

Тут E - все та ж функція помилки;

Wc - деяке початкове значення ваги.

Візьмемо похідну функції помилки по вагам, відкинувши члени вище 2го порядку:

$$\frac{dE(W)}{dW} = \frac{dE(W_c)}{dW} + (W - W_c)\frac{d^2E(W_c)}{dW^2} \quad (4.3)$$

з цього виразу випливає, що вага, при якому значення функції помилки буде мінімальним можна обчислити з наступного виразу:

$$W_{min} = W_c - \left(\frac{d^2E(W_c)}{dW^2}\right)^{-1}\frac{dE(W_c)}{dW} \quad (4.4)$$

Тобто оптимальна вага обчислюється як поточний мінус похідна функції помилки по вазі, поділена на другу похідну функції помилки. Для багатовимірного випадку (тобто для матриці ваг) все одно, тільки перша похідна перетворюється в градієнт (вектор приватних похідних), а друга похідна перетворюється в гессіан (матрицю друге окремих похідних). І тут можливі два варіанти. Якщо ми опустимо другу похідну, то отримаємо

алгоритм найшвидшого градієнтного спуску. Зазвичай гессіан замінюють чимось простішим. Наприклад, один з найвідоміших і успішних методів - метод Левенберга-Марквардта (ЛМ) замінює гессіан, його апроксимацією за допомогою квадрадного якобіана.

Алгоритм ЛМ вимагає обробки всієї навчальної вибірки, тоді як алгоритм градієнтного спуску може працювати з кожною окремо взятою навчальною вибіркою. В останньому випадку алгоритм називають стохастичним градієнтом. З огляду на те, що наша база містить 60 000 навчальних зразків нам більше підходить стохастичний градієнт. Ще однією перевагою стохастичного градієнта є його менша схильність потрапляння в локальний мінімум в порівнянні з ЛМ.

Представлені формули дозволяють легко обчислити похідну помилки по вагам, що знаходяться в вихідному шарі.

4.4.1 Реалізація навчання

Ідея навчання полягає в тому, щоб змусити нейронну мережу узагальнювати. Величезна кількість часу витрачається на аналіз навчального набору, але в реальному світі продуктивність нейронної мережі на навчальному наборі не представляє ніякого інтересу. Практично будь-яка нейронна мережа може бути навчена настільки добре, що на навчальній множині може взагалі не виникнути ніяких помилок.

Здатність нейронної мережі процвітати в патернах, яких вона ніколи не бачила раніше, називається «узагальненням». Зусилля, що витрачається на навчання, змушує нейронну мережу бачити внутрішні характеристики навчального набору, які, як ми сподіваємося, будуть присутні і в реальному світі. Коли ми обчислюємо MSE навчального набору, ми робимо це з розумінням, що це тільки оцінка числа, яке нас дійсно цікавить, а саме MSE даних реального світу.

Кілька методів використовуються під час навчання, щоб поліпшити узагальнення нейронної мережі. В якості одного прикладу, навчальний набір передається через нейронну мережу в рандомізованому порядку в кожній епосі. Концептуально, ці методи примушують ваги трохи «відскакувати», тим самим перешкоджаючи нейронної мережі обґрунтовуватися на вагах, які підкреслюють помилкові атрибути шаблонів. За фактом є атрибути, які властиві шаблонами в навчальному наборі, але не є внутрішніми характеристиками моделей, що зустрічаються в реальному світі.

Одним з найбільш важливих методів поліпшення узагальнення є спотворення: схеми навчання злегка спотворюються перед використанням в зворотному поширенні. Теорія полягає в тому, що ці спотворення змушують нейронну мережу більш уважно вивчати важливі аспекти навчальних шаблонів, оскільки ці шаблони будуть відрізнятися кожен раз, коли нейронна мережа їх бачить. Інший спосіб поглянути на це полягає в тому, що спотворення збільшують розмір тренувального набору, штучно створюючи нові тренувальні шаблони з існуючих.

У цій програмі `CMNistDoc::GenerateDistortionMap()` функція генерує карту спотворень, яка застосовується до шаблонів навчання (використовуючи `CMNistDoc::ApplyDistortionMap()` функцію) під час навчання. Спотворення розраховується випадковим чином для кожного шаблону безпосередньо перед зворотним поширенням. Застосовуються три різних типи спотворень:

- Коефіцієнт масштабування: масштаб малюнка змінюється, щоб збільшити або зменшити його. Коефіцієнт масштабування для горизонтального напрямку відрізняється від коефіцієнта масштабування для вертикального напрямку, так що можна бачити усадку в вертикальному масштабі і збільшення в горизонтальному масштабі.

- Обертання: весь малюнок обертається за годинниковою стрілкою або проти годинникової стрілки.

- Пружні: це слово запозичене у доктора Сімарда «Кращі практики».

Візуально, пружні спотворення виглядають як легке натиснення і розтягнення пікселів в візерунку, майже як при перегляді візерунка в хвилястою воді.

Результат спотворень показаний на рисунку 4.5, який показує оригінал шаблону для цифри «3» разом з 25 прикладами спотворень, застосованих до шаблону.

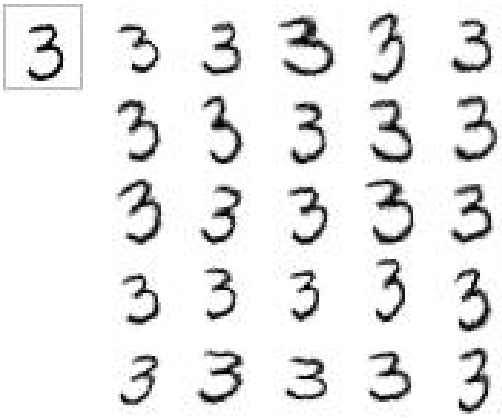


Рисунок 4.5 – Результат спотворення цифри 3

Ви можете побачити ефект від трьох типів спотворень, але для людини ясно, що кожне спотворення все ще «схоже» на цифру «3». Для нейронної мережі, однак, зразки пікселів помітно відрізняються. Через цих відмінностей нейронна мережа змушена дивитися за межі фактичних шаблонів пікселів, розуміти і застосовувати внутрішні якості, які роблять «3» схожим на «3».

Можливо вибрати, чи слід застосовувати спотворення, коли ви натискаєте кнопку «Почати зворотнє поширення» в уявленні навчання, і в цей момент відображається наступне діалогове вікно, в якому можна також вибрати інші параметри навчання (рисунок 4.6).

Створили мережеві опції навчання. Навчали протягом 30 епох (рисунок 4.7). Була встановлена початкова буква, вивчався рівень 0,001 і знижений темп навчання після 20 епох. Важливо було контролювати мережеву точність під час навчання шляхом визначення даних про валідації і частоти валідації.

Програмне забезпечення навчає мережу на навчальних даних і обчислює точність на дані про валідації рівномірно під час навчання. Дані про валідації не використовуються, щоб оновити мережеві ваги (приклад 4.1).

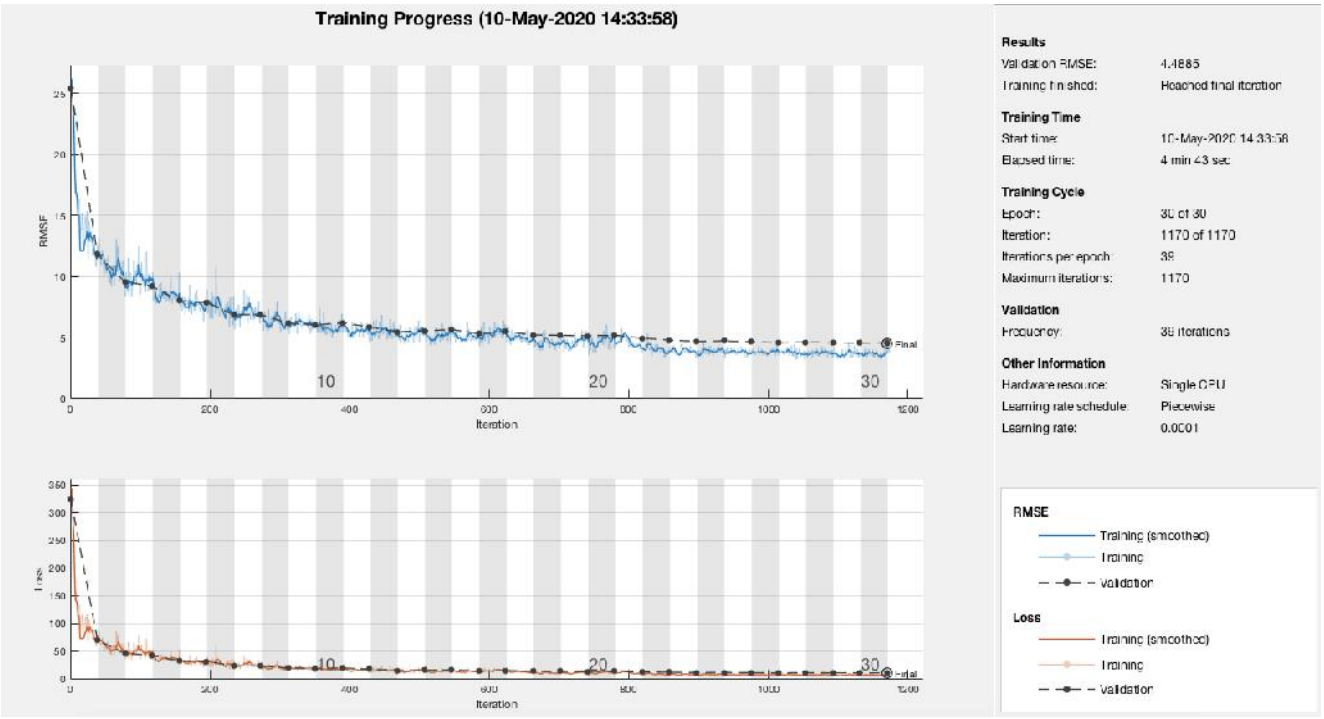


Рисунок 4.6 – Тренувальний процес

```
miniBatchSize  = 128;
validationFrequency = floor(numel(YTrain)/miniBatchSize);
options = trainingOptions('sgdm', ...
    'MiniBatchSize',miniBatchSize, ...
    'MaxEpochs',30, ...
    'InitialLearnRate',1e-3, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropFactor',0.1, ...
    'LearnRateDropPeriod',20, ...
    'Shuffle','every-epoch', ...
    'ValidationData',{XValidation,YValidation}, ...
    'ValidationFrequency',validationFrequency, ...
    'Plots','training-progress', ...
    'Verbose',false);
```

Приклад 4.1 – Мережеві опції навчання

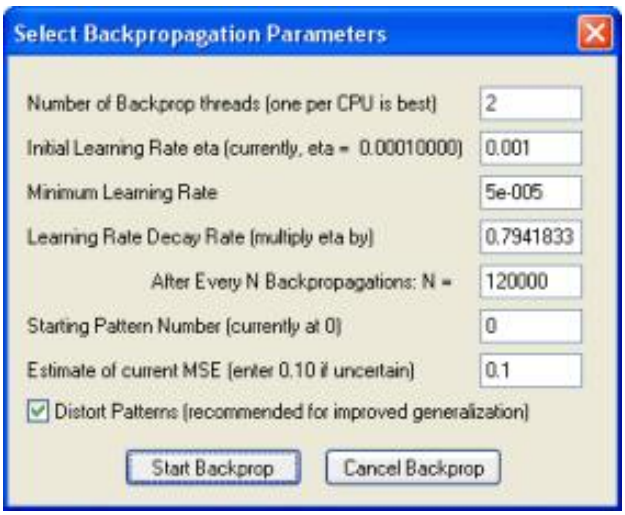


Рисунок 4.7 – Вікно вибору параметрів зворотнього розповсюдження

Крім спотворень, цей діалог дозволяє встановити інші параметри для зворотного поширення. Зокрема, необхідно вказати початкову швидкість навчання і кінцеву швидкість навчання. Взагалі кажучи, швидкість навчання повинна бути вище для непідготовлених нейронних мереж, так як велика швидкість навчання дозволить великі зміни в вагах під час зворотного поширення. Швидкість навчання поступово знижується під час навчання, так як нейронна мережа вчиться, щоб ваги могли сходитися до деякого кінцевого значення. Але ніколи не має сенсу занадто сильно знижувати швидкість навчання, інакше мережа перестане вчитися. Це мета записи для остаточного значення.

4.1.2 Створення мережевих шарів

Щоб вирішити задачу регресії, створіть шари мережі і включайте шар регресії в кінці мережі.

Перший шар задає розмір і тип вхідних даних. Вхідні зображення 28 28 1. Створено вхідний шар зображень одного розміру з навчальними зображеннями.

Середні верстви мережі define базова архітектура мережі, де велика

частина розрахунку і вивчення відбуваються.

Останні шари задають розмір і тип вихідних даних. Для проблем регресії повнозв’язну шар повинен передувати шару регресії в кінці мережі. Був створений повністю пов’язаний вихідний шар розміру 1 і шар регресії.

Об’єднані всі шари разом в Layer масив (приклад 4.2).

```
layers = [
    imageInputLayer([28 28 1])

    convolution2dLayer(3,8,’Padding’,’same’)
    batchNormalizationLayer
    reluLayer

    averagePooling2dLayer(2,’Stride’,2)

    convolution2dLayer(3,16,’Padding’,’same’)
    batchNormalizationLayer
    reluLayer

    averagePooling2dLayer(2,’Stride’,2)

    convolution2dLayer(3,32,’Padding’,’same’)
    batchNormalizationLayer
    reluLayer

    convolution2dLayer(3,32,’Padding’,’same’)
    batchNormalizationLayer
    reluLayer

    dropoutLayer(0.2)
    fullyConnectedLayer(1)
    regressionLayer];
```

Приклад 4.2 – Масив об’єднання шарів

4.5 Реалізація згорткової нейронної мережі

Для реалізації різних алгоритмів, від яких не потрібно функціонування в реальному часі використовуємо Matlab. Це дуже зручний пакет, М-мову якого дозволяє зосередитися на самому алгоритмі не піклуючись про виділення пам’яті, операціях в / в і т.п. Крім того, безліч різних тулбоксів дозволяє створювати дійсно міждисциплінарні програми в найкоротші терміни.

Наприклад, за допомогою Image Acquisition Toolbox підключити

вебкамеру, за допомогою Image Processing Toolbox обробляти зображення з неї, за допомогою Neural Network Toolbox формувати траєкторію руху робота, а за допомогою Virtual Reality Toolbox і Control Systems Toolbox моделювати рух робота. До речі про Neural Network Toolbox - це досить гнучкий набір інструментів, що дозволяє створювати безліч нейронних мереж різних типів і архітектур, але, на жаль, не згорткові НС. Вся справа в поділюваних вагах. У NN toolbox можливість задавати колективні ваги відсутня.

Для початку там потрібно інтегрувати бібліотеку з набором необхідних зображень. Так ми зможемо швидко з’ясувати джерело даних для навчання. У прикладі 4.3 був представлений базовий варіант вибору джерела даних.

```
      mnist_train_reader.num_samples = 60000;
      mnist_train_reader.current = 1;
      mnist_train_reader.data_file    =    '../../data/MNIST/train-images.idx3-
ubyte';
      mnist_train_reader.label_file   =   '../../data/MNIST/train-labels.idx1-
ubyte';
      mnist_train_reader.buffer_size = 1000;
      mnist_train_reader.read = @mnist_datareader;
```

Приклад 4.3 – Вибір джерела даних

```
      %загальна кількість слоїв
      cnet_struct.nlayers = 7;
      %кількість вхідних зображень.
      cnet_struct.nInputs = 1;
      %ширина
      cnet_struct.inputWidth = 32;
      %висота
      cnet_struct.inputHeight = 32;
```

Приклад 4.4 – Параметри ЗНМ

Наступним кроком було визначення загальних параметрів роботи згорткової нейройної мережі. До них ми відносимо:

- загальна кількість слоїв;
- кількість вхідних зображень (яка одночасно обробляється);

- ширина зображення;
- висота зображення.

Наступними кроками роботи було визначення по мережі визначених параметрів. На першому слої ми відокремили 6 згортки ядер з мартицею розміром 5x5.

Під час роботи над другим слоєм ми задали чітку кількість весів та зміщення – по 6.

На третьому слої ми зазначили кількість ядер для обробки – 16. Матриця була обрана розмірністю 5x5. Вид підключення карти був запозичений у Оксфордських роботах з обробки документів.

```
cnet_struct.layers{3}.NumFMaps = 16;
cnet_struct.layers{3}.KernelWidth = 5;
cnet_struct.layers{3}.KernelHeight = 5;
cnet_struct.layers{3}.TransferFunc = 'tansig_mod';
cnet_struct.layers{3}.LayerType = 'clayer';
cnet_struct.layers{3}.conn_map = ...
[1 0 0 0 1 1 1 0 0 1 1 1 1 0 1 1;
 1 1 0 0 0 1 1 1 0 0 1 1 1 1 0 1;
 1 1 1 0 0 0 1 1 1 0 0 1 0 1 1 1;
 0 1 1 1 0 0 1 1 1 1 0 0 1 0 1 1;
 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 1;
 0 0 0 1 1 1 0 0 1 1 1 1 0 1 1 1;
];
```

Приклад 4.5 – Третій слой

На всіх наступних обробках слоїв ми зазначали експериментально виведені кількості ядер та размірність. Завдяки цьому ми повністю аналізували вхідне зображення та система розпізнавала представлену у прикладі цифру.

Наша система має деякі недоліки у роботі. Основним недоліком було виликий загальний час навчання (рисунок 4.8).

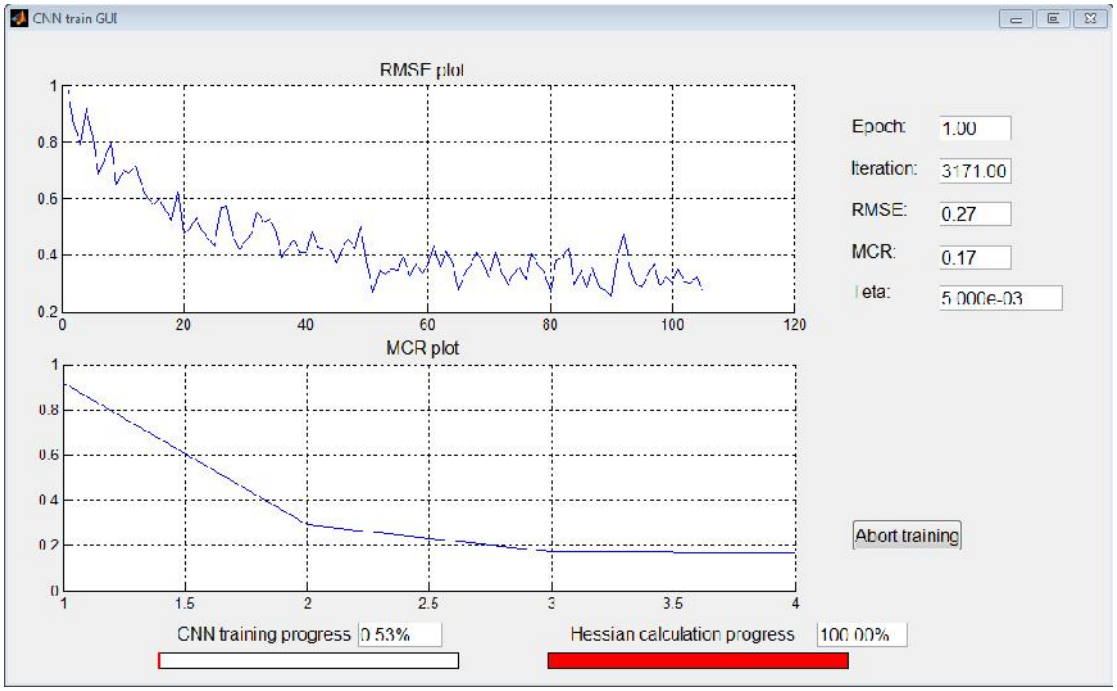


Рисунок 4.8 – Навчання нейронної мережі

Однак ця проблема не є критичною та реалізація вирішує поставлену проблему. Наступними кроками для вдосконалення такого методу є:

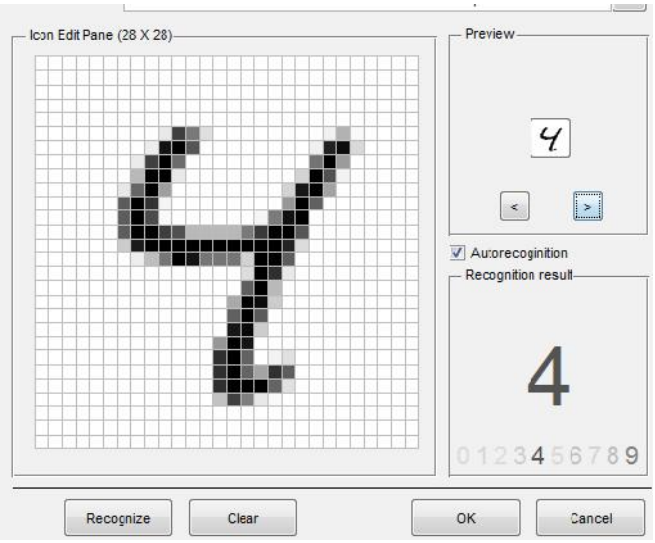
- розширення системи для обробки рукописного тексту різними мовами;

- навчання нейронної мережі задля влосконалення обробки почерку людини та идентифікування особистих підписів.

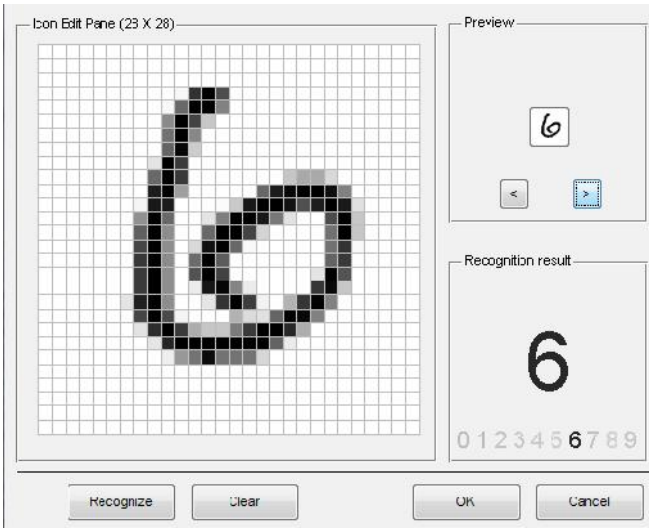
Всі результати представлені у наступному підрозділі.

4.6 Результати

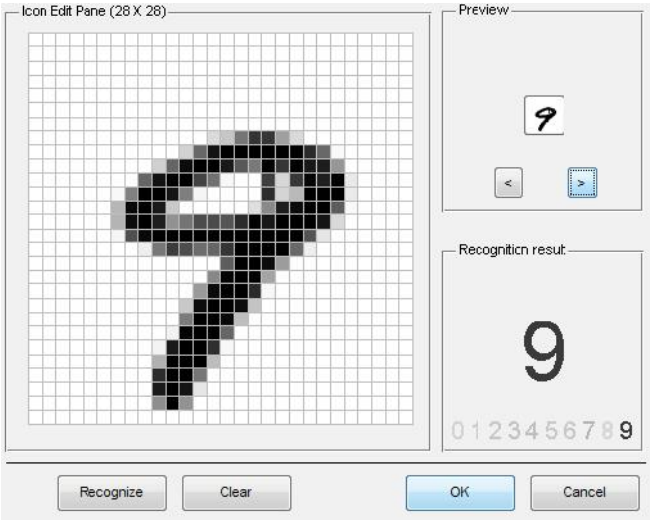
Основною метою роботи було обробка зображень на основі згорткової нейронної мережі. Так, за допомогою MatLab буди отримані результати? Приведені на рисунку 4.9.



а)

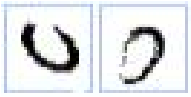


б)



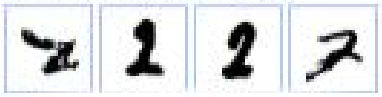
в)

Рисунок 4.9 – Результат роботи: а) розпізнавання 4; б) розпізнавання 6; в) розпізнавання 9



0 => 6 0 => 7

1 => 7 1 => 3 1 => 8



2 => 7 2 => 1 2 => 8 2 => 7

3 => 7 3 => 5 3 => 5

4 => 6 4 => 9 4 => 9 4 => 9 4 => 9

5 => 3 5 => 3 5 => 3 5 => 3 5 => 3 5 => 6 5 => 3 5 => 0 5 => 3 5 => 3 5 => 6 5 => 0

6 => 1 6 => 1 6 => 1 6 => 0 6 => 8 6 => 1 6 => 5 6 => 3 6 => 8 6 => 5 6 => 5 6 => 2

7 => 3 7 => 2 7 => 9 7 => 2 7 => 2 7 => 2 7 => 2

8 => 3 8 => 2 8 => 9 8 => 1 8 => 4 8 => 0 8 => 9 8 => 3 8 => 7 8 => 6 8 => 4 8 => 9 8 => 5

9 => 5 9 => 5 9 => 4 9 => 7 9 => 5 9 => 1 9 => 4 9 => 4 9 => 4 9 => 8 9 => 7 9 => 8 9 => 8

Рисунок 4.10 – Помилки

Однак нейромережа має неточності у своїй роботі. Так, система не розпізнає дуже засвітлені зони зображення та некорректно видає результат.

Приклади помилок, які зробила згорткова нейромережа з поясленням кожної представлені на рисунку 4.10.

Для деяких з цих шаблонів легко зрозуміти, чому нейронна мережа була неправильною: шаблони дійсно збивають з пантелику. Наприклад, патерн з шостого ряду, третыз в кынця, який нібито називається 5, виглядає для мене як 3, як і думала нейронна мережа. Інші шаблони, здається, мають артефакти від попередньої обробки авторами MNIST, і саме наявність цих артефактів заплутало мережу. Наприклад, останный номер другого ряду, який явно дорівнює 1, також має артефакти на правій стороні, можливо, викликані, коли шаблон був відділений від сусіднього шаблону.

Але для переважної більшості нейронна мережа просто помилилася. Шаблони не дуже заплутані. Кожна з 7 виглядає як 7, і немає вагомої причини, по якій нейронна мережа не змогла розпізнати їх як 7.

Цю помилку можливо вирішити за допомогою навчання згорткової нейронної мережі спочатку, але верогідність вдачі не більше 30%.

ВИСНОВКИ

У данній роботі було проведено дослідження методів обробки зображень на основі згорткових нейронних мереж до інтелектуального аналізу зображень на прикладі вирішення задачі розпізнавання рукописних цифр різної якості.

Було досліджено теоретичні відомості вирішення задачі класифікації зображень засобами машинного навчання, які включають згорткові та рекурентні нейронні мережі. Також були проаналізовані різні методи передобробки зображення для більшої якості розпізнавання.

Під час аналізу існуючих рішень були досліджені різноманітні архітектури нейронних мереж. За результатами теоретичних даних та існуючих рішень було виокремлено деякі проблеми існуючих підходів. Одна із них пов’язана з виконанням класифікації зображень або фото з подальшою агрегацією результатів.

У ході реалізації практичної частини було здійснено створення та навчання згорткової нейронної мережі для розпізнавання рукописних цифр. Так, на основі загальнодоступної бібліотеки MNIST було здійснено тестування створеного рішення. Завдяки більше ніж 6000 образців цифр у цій бібліотеці було детально протестоване створене рішення.

Так, було побачено, що під час розпізнавання зображення згорткова нейронна мережа робить помилки, які людині здаються очевидними, а для штучного інтелекту – важка задача. Система допускає помилки в тих випадках, де є напівпрозора частина.

Надалі можна розширювати систему для розпізнавання різного почерку людини та конфігурації отриманого тексту у печатний шрифт та подальшої роботи з ним у текстових документах.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Hu X., Ahuja N. Motion and Structure Estimation Using Long Sequence motion models.// Image and Vision Computing. – 1993 – 569.

2. Марр Д. Зор: інформаційний підхід до вивчення подання та обробки зорових образів. – М.: Радио и связь, 1987.

3. Задача фильтрації зображень. [Электронний ресурс]. – Режим доступу: http://wiki.technicalvision.ru/index.php/

4. Колесник А. В. - Розподілена программна система для розпізнавання зображень. [Электронний ресурс]. – Режим доступу: http://masters.donntu.org/2010/fknt/kolesnik/diss/index.htm

5. Фильтрація зображень різними способами. Швидке перетворення Фур'е [Электронний ресурс] – Режим доступу: https://sites.google.com/site/alexeysidnev/docs/filter_fft

6. Sabour S. Dynamic routing between capsules / S. Sabour, N. Frosst, G. E. Hinton. //Advances in Neural Information Processing Systems (NIPS). – 2017. – С. 3859–3869.

7. Згорткова нейронна мережа [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Згорткова_нейронна_мережа.

8. Cooley J. W. An algorithm for the machine calculation of complex Fourier series / J. W. Cooley, J. W. Tukey. // Mathematics of Computation. – 1965. – No19(90). – С. 297–301.

9. Dettmers T. How to Parallelize Deep Learning on GPUs Part 1/2: Data Parallelism [Електронний ресурс] / Tim Dettmers – Режим доступу до ресурсу: http://timdettmers.com/2014/10/09/deep-learning-data-parallelism/.

10. Krizhevsky A. ImageNet classification with deep convolutional neural networks / A. Krizhevsky, I. Sutskever, G. E. Hinton. // Advances in Neural Information Processing Systems (NIPS). – 2012. – No25. – С. 1097–1105.

11. Руденко О., Бессонов А. Адаптивное управление многомерными

нелинейными объектами на основе радиально-базисных сетей [Текст] / Руденко О. Бессонов А.: Кибернетика и системный анализ, 2005. – 168-176 с.

12. Rudenko O., Bezsonov O.,Romanyk O., Lebediev O. Analysis of convergence of adaptive algorithms for the identification of non-stationary objects. Eastern-European Journal of Enterprise Technologies, 4 (97), 2019. pp. 6-14.

13. Bezsonov O., Rudenko O., Udovenko S., Dudinova O. Processing of noisy digital images with use of evolving autoencoders. Eastern-European Journal of Enterprise Technologies, 6/9 (90), 2017 . pp. 63-69.

14. Petruk S., Zhyvotovskyi R., Shyshatskyi A. Mathematical Model of MIMO // 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T). 2018. doi: https://doi.org/10.1109/ infocommst.2018.8632163

15. Korchenko A. Neural network models, methods and tools for assessing the security parameters of Internet-oriented information systems: monograph / A. Korchenko, I. Tereykovsky, N. Karpinsky, S. Tynimbaev. K.: TOV "Our Format". 2016. - 275 p.

16. Rudenko O.G. Shtuchni neironni schizhi. Nav. посіб. / O.G. Rudenko, Є.V. Bodiansky. - Ch: "CMIT", 2006. - 404 p.

17. Fedotov D. V. Optimisation of convolutional neural network structure with self-configuring evolutionary algorithm in one identification problem / D. V. Fedotov, E. A. Popov, V. A. Okhorzin // Vestnik SibGAU. - Vol. 16, No. 4. - P. 857–862.