Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Метод підвищення точності аналізу та обробки складноструктурних зображень

(тема)

Виконав:
здобувач 2 року навчання,
групи СПм-23-5
Важа ЧХЕІДЗЕ
(власне ім'я, прізвище)

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: зав.каф. Андрій КОВАЛЕНКО
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

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

Освітня програма Системне програмування
(повна назва)

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

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Чхеідзе Важі Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Метод підвищення точності аналізу та обробки складноструктурних зображень

затверджена наказом по університету від "21" квітня 2025 р. № 296 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 16 червня 2025 р.

3. Вхідні дані до роботи операційна система –Windows або Linux, технологія OpenGL

4. Перелік питань, що потрібно опрацювати у роботі

1) Аналіз сучасного стану аналізу та обробки складноструктурних зображень.
2) Метод обробки складноструктурних зображень.
3) Експериментальна перевірка методу підвищення точності аналізу та обробки складноструктурних зображень.
4) Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

Слайд-презентація – 15 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	Строк / терміни виконання етапів роботи	Примітка
1	Огляд літературних джерел та аналіз	22.04.25-29.04.25	
	предметної області досліджень		
2	Вибір та обгрунтування методики дослідження	30.04.25-05.05.25	
3	Вибір інструментальних засобів	06.05.25-09.05.25	
4	Розробка моделей протоколів	10.05.25-21.05.25	
5	Проведення експериментів	22.05.25-02.06.25	
6	Оформлення матеріалів кваліфікаційної роботи	03.06.25-05.06.25	
7	Подання кваліфікаційної роботи керівникові та	06.06.25-09.06.25	
	її попередній захист		
8	Подання кваліфікаційної роботи на	10.06.25-12.06.25	
	рецензування		

Дата видачі завдання " 21 " квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ зав.каф. Андрій КОВАЛЕНКО
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 95 с., 17 рис., 8 табл., 1 дод., 17 джерел.

СКЛАДНОСТРУКТУРНІ ЗОБРАЖЕННЯ, ТОЧКОВИЙ ОБ’ЄКТ, ЛІНІЙНИЙ ОБ’ЄКТ, ПЛОЩИННИЙ ОБ'ЄКТ.

Об'єктом дослідження є процес аналізу та обробки растрових складноструктурних зображень. Предметом дослідження є моделі та підвищення точності та продуктивності алгоритмів аналізу та обробки складноструктурних зображень. Мета кваліфікаційної роботи полягає у підвищенні точності та продуктивності алгоритмів аналізу та обробки складноструктурних зображень.

У ході виконання кваліфікаційної роботи було проведено аналіз сучасного стану аналізу та обробки растрових складноструктурних зображень. Проведені дослідження підходів до локалізації та визначення типів сегментованих об'єктів. Проведені дослідження підходів до розпізнавання образів та їх угруповання. Удосконалений та досліджений метод підвищення точності аналізу та обробки складноструктурних зображень.

ABSTRACT

Master’s thesis: 95 pages, 17 figures, 8 tables, 1 appendices, 17 sources.

COMPLEX STRUCTURE IMAGES, POINT OBJECT, LINEAR OBJECT, PLANE OBJECT.

The object of the study is the process of analysis and processing of raster complex-structure images. The subject of the study is models and increasing the accuracy and productivity of algorithms for analyzing and processing complex-structure images. The purpose of the qualification work is to increase the accuracy and productivity of algorithms for analyzing and processing complex-structure images.

During the qualification work, an analysis of the current state of analysis and processing of raster complex-structure images was conducted. Research was conducted on approaches to localization and determination of types of segmented objects. Research was conducted on approaches to pattern recognition and their grouping. A method for increasing the accuracy of analysis and processing of complex-structure images was improved and investigated

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ГІС – геоінформаційної системи

ІКТ – інфокомунікаційні технології

ПЗ – програмне забезпечення

ЛО – лінійний об’єкт

ПО – площинний об'єкт

ССЗ – складноструктурне зображення

ТО – точковий об’єкт

ЦОД – центр обробки даних

ШНМ – штучні нейронні мережі

ШІ – штучний інтелект

AI – штучний інтелект

IoU – Intersection over Union

QoS – Quality of Service

RNN – рекурентна нейронна мережа

RTT – Round trip Time

ВСТУП

В даний час відбувається прискорений розвиток засобів обробки та аналізу інформації внаслідок накопичення великої кількості даних про різні об'єкти і процеси. Одними з важливих складових таких засобів є програмні комплекси аналізу та обробки растрових складноструктурних зображень, що поєднують у собі числові та графічні дані. Метою таких програмних комплексів є отримання інформації про просторово-розподілені об'єкти для подальшої обробки. Основними підходами при цьому є векторизація та розуміння семантичних зв'язків між об'єктами.

Прикладами ССЗ є діаграми, номограми, креслення, схеми, архітектурні проекти, карти місцевостей. У цілому нині дані зображення можна охарактеризувати наявністю багатьох семантичних верств, накладених друг на друга у візуальному плані. Наприклад, топографічні карти, які можна назвати типовим представником ССЗ, містять найбільш повну інформацію про місцевість, тому з них можна отримати інформацію для геоінформаційної системи (ГІС), які можна використовувати як підтримку прийняття рішення на етапі проектування для вирішення різних завдань, наприклад, оптимального розміщення станцій мобільного зв'язку, логістичних центрів, транспортних розв'язок і транспортних розв'язок. Також ГІС є корисним інструментом аналізу для здійснення, наприклад, кадастрових робіт, передбачення потенційних зон пожеж та затоплень. Згідно зі звітом «Geographic Information System (GIS) Market», опублікованому Allied Market Research, загальносвітовий ринок ГІС зросте до 25,5 млрд. дол.

При цьому ручна обробка великої кількості ССЗ стає економічно неефективною. Проблема розпізнавання та аналізу образів набула визначного значення в умовах інформаційних перевантажень, коли людина не справляється з розумінням вступників до нього все зростаючих обсягів даних. В результаті його мозок переключається на режим одночасного

сприйняття і мислення, якому таке розпізнавання властиво, але негативно впливає на якість роботи в умовах небажаності помилок. Тому сучасний період розвитку інформаційних технологій характеризується широким впровадженням інтелектуальних методів у процес обробки графічної інформації, а автоматична обробка та аналіз зображень, що володіють складною структурою, є одним з найбільш перспективних напрямів розвитку.

Отже, науково-технічне завдання підвищення точності аналізу та обробки складноструктурних зображень є актуальним.

Об'єктом дослідження є процес аналізу та обробки растрових складноструктурних зображень.

Предметом дослідження є моделі та методи підвищення точності та продуктивності алгоритмів аналізу та обробки складноструктурних зображень.

Мета кваліфікаційної роботи полягає у підвищенні точності та продуктивності алгоритмів аналізу та обробки складноструктурних зображень.

Завдання, вирішення яких необхідне для досягнення мети дослідження.

- аналіз сучасного стану аналізу та обробки растрових складноструктурних зображень;
- дослідження підходів до локалізації та визначення типів сегментованих об'єктів;
- дослідження підходів до розпізнавання образів та їх угруповання;
- удосконалення та дослідження методу підвищення точності аналізу та обробки складноструктурних зображень.

Для вирішення поставлених завдань були застосовані геометричні методи, методи математичної морфології, теорії графів, математичного та імітаційного моделювання.

Наукова новизна дослідження полягає в такому:

удосконалений метод підвищення точності аналізу та обробки складноструктурних зображень за рахунок сегментації, локалізації,

розпізнавання та групування точкових, лінійних і площинних об'єктів, що засновані на комплексному використанні відомих методів аналізу та обробки зображень.

Практична цінність дослідження полягає в підвищенні точності аналізу та обробки складноструктурних зображень.

За результатами проведених досліджень була надрукована наукова стаття в фаховому виданні за 123 спеціальністю «Системи управління, навігації та зв’язку» [1].

1 АНАЛІЗ СУЧАСНОГО СТАНУ АНАЛІЗУ ТА ОБРОБКИ СКЛАДНОСТРУКТУРНИХ ЗОБРАЖЕНЬ

1.1 Класифікація складноструктурних зображень

Структурне зображення (як часткове від терміну «структурні дані») є зображенням, що має внутрішню організацію або метаінформацію про об'єкти, що відображаються. Складність цієї організації, виражену у вигляді великої кількості залежностей та різноманітних проявів об'єктів, призводить до складності сприйняття, аналізу та обробки. Так як векторні зображення і елементи на них мають відомий математичний опис і властивості, то їх програмна обробка може бути виконана автоматично з використанням формальних методів. На відміну від них, растрові зображення не зберігають безпосередньо всіх характеристик об'єктів, їх аналіз та обробка є актуальним науковим завданням, і тому саме вони розглянуті на цій роботі.

Поняття «складноструктурне зображення» зараз є зонтичним терміном, що включає кілька різнорідних визначень. У одних джерелах [1, 2] під зображенням зі складною структурою розуміється «зображення, що складається з ряду просторово організованих статистично однорідних фрагментів різних класів і детермінованих зображень, розташованих на їх тлі». У інших [3, 4] виділені такі ознаки: поєднання графічних (в т.ч. різної розмірності) і текстових елементів, складноформатність, залежність атрибутів об'єктів від розташування.

Цифрове зображення сприймається у двох планах: візуальному та семантичному. Візуальний план містить у собі растрове опис об'єктів у одному з колірних просторів. Семантичний план містить у собі інформацію про ці об'єкти. Якщо на зображенні існує кілька різних за природою або функціями об'єктів, то на ньому буде кілька семантичних планів; поділ візуальних планів можливий тільки при існуванні можливості поділу в

колірному просторі, в загальному випадку можна говорити про гетерархічність візуальних планів. Під гетерархічності тут розуміється можливість перетину візуальних об'єктів та/або неможливість виділення для них просторових відносин «зовні», «всередині», «поруч».

Відповідність між семантичними та візуальними планами визначається за допомогою правил відображення, які є метаінформацією. Порушення цих правил або їх нерегулярність також є ознакою складноструктурності [5]. Порушення правил, однак, не повинно бути занадто частим або значним: для ССЗ все-таки передбачається можливість їхнього аналізу та обробки.

Також однією з ознак складноструктурності можна виділити високу інформаційну ємність. Цю ємність можна порахувати, якщо векторизувати вихідне растрове зображення і розділити розмір файлу, що вийшов, на одиницю площі вихідного зображення. Порівнювати інформаційні ємності можна лише за однакового дозволу (dpi) вихідних растрових файлів.

Можна виділити такі класи зображень: відбивають реальність (клас I), створені людиною рукотворно (клас II), створені людиною автоматизовано (клас III). Класи далі представлені разом із сучасними науковими дослідженнями. Ця класифікація не розглядає найпростіші зображення (однорідні фони, градієнти, текстури) і фрактальну графіку, оскільки дані зображення мають векторну, ніж растрову природу. Виведені підкласи розташовані від мають більш загальне застосування до більш вузькоспеціалізованим: такий характер мають і методи, що використовуються для їх обробки.

У результаті ССЗ можна визначити таким чином.

Складноструктурне зображення – растрове зображення, яке містить об'єкти, що відносяться до різних семантичних планів, і яке володіє набором явних і неявних правил відображення цих об'єктів на візуальний план за допомогою різних технік у вигляді розподілених образів, що передбачають можливість однозначного відновлення змісту вихідних об'єктів та їх атрибутів.

Неявні правила включають не тільки недетерміновані, а й описують [5] «неповну, різнорідну, непряму, нечітку, неоднозначну, імовірнісну» інформацію.

1.2 Концепція та типові завдання обробки цифрових зображень

Проблема обробки зображень безпосередньо з кількома областями досліджень. З одного боку, це вивчення людського зору, як фізіологічного процесу, і сприйняття кольору, форм, текстур та образів, як психологічного. З іншого боку, це розробка математичного та алгоритмічного апаратів для імітації даних форм людської діяльності, створення моделей зображень, як фізичних об'єктів матеріального світу. Загальні питання комп'ютерного зору розглядаються в [7].

Такі наукові галузі, як комп'ютерний зір, машинний зір та обробка зображень, що тісно пов'язані між собою. Обробка зображень має на увазі переробку та аналіз цифрових зображень без знань про зміст таких зображень. Для комп'ютерного зору деякі знання вже є необхідними, оскільки відбувається процес як обробки, а й розуміння те, що відбувається на зображенні. Машинний зір, що застосовується на виробництві, повністю пов'язаний із зображеннями певного типу, наприклад, робот-маніпулятор відстежує переміщення деталей, що конкретно виглядають по конвеєру. Тому в задачі розпізнавання ССЗ необхідно користуватися методами обробки зображень на початковій стадії, коли ми ще не користуємося спеціальними знаннями, а потім переходити в область комп'ютерного зору.

Послідовність етапів обробки є наступною: передобробка, редукція даних, сегментація, розпізнавання образів та аналіз сцен; всі етапи мають супроводжуватися оптимізацією. Відповідно, проблемами є отримання репрезентативної вибірки зображень для машинного навчання, вибір образів для розпізнавання, адаптація алгоритмів, узагальнення та оцінка якості виконаних алгоритмів.

Будь-яке ССЗ є безперервним на площині свого відображення, причому колірні відчуття можна описати за допомогою набору колірних координат. Зазвичай розглядають колірний простір RGB, що складається з трьох компонентів: червоного, зеленого та синього кольорів. Таким чином, кожен колір розуміється, як вектор у колірному просторі.

Для обробки зображення на цифровому пристрої необхідно перевести зображення в цифровий формат, тобто дискретизувати його, що здійснюється за допомогою сканування. При цьому відбувається, по-перше, квантування простору, що відповідає дозволу скануючого пристрою (dpi), а по-друге, квантування колірних координат, що відповідає глибині кольору (зазвичай використовується TrueColor, що кодує колір по 8 біт на кожну колірну компоненту). У результаті сканування виходить двовимірна матриця (растр), елементами якої є колірні вектори окремих точок зображення - пікселів.

Отримане цифрове зображення ще непридатне для такого складного аналізу, як розпізнавання образів, так як володіє численними спотвореннями [9], викликаними як фізичними змінами в документі (поява потертостей, плям), так і скануванням (наприклад, шумів від фотодетектора, «стрибків» колірних векторів між сусідніми). Щоб позбавитися цих спотворень, необхідно провести покращення та реставрацію цифрового зображення.

Передобробка здійснюється без участі людини, тому що в цьому випадку ще не виконано редукцію даних і відбувається обробка досить великих масивів. В основі операцій, що покращують зображення - попіксельні перетворення, наприклад, посилення контрастності, підвищення різкості або видозміна гістограми за рахунок її вирівнювання. Також сюди відносяться перетворення колірних просторів, у тому числі перетворення зображення в бінарне за допомогою порогової обробки.

Для придушення шумів можна скористатися тим, що вони не мають просторової кореляції між собою, тому можуть бути «видалені» за рахунок використання статистичних фільтрів, що враховують значення векторів кольорів пікселів-сусідів, наприклад, середній або медіанний фільтр. Також

застосовуються високо- та низькочастотні фільтри, перетворення Фур'є, Уолша, Хаара. Такі фільтри мають свій недолік, так як «розмивають» зашумлені області, що неприпустимо при обробці зображень з низьким дозволом; за шум можуть бути сприйняті невеликі образи, наприклад, точки відображення штрихпунктирної лінії.

Після покращення зображення його необхідно кластеризувати, тобто розділити всі пікселі по кластерах приблизно одного кольору, так як зазвичай правила відображення будуються так, що об'єкти, що належать до одного семантичного поля, що характеризуються одним кольором у візуальному полі. Найпростіша кластеризація полягає в перенесенні векторів всіх пікселів карти в один колірний простір і виділення найбільш щільних областей в окремий кластер. Таким чином, все різноманіття кольорів замінюється функцією приналежності пікселя до певного кластера, що означає редукцію по глибині кольору.

Множини пікселів, розбиті за кластерами, ще не є образами, це лише об'єднання пікселів приблизно одного кольору. Образи одного кольору можуть відрізнятися за формою або текстурою, наприклад, річки та озера зображуються за допомогою блакитного кольору, але є образами різних розмірностей. Тому необхідно провести сегментацію: розбити пікселі, що належать одному кластеру, на локально безперервні підмножини, що мають деяку властивість, наприклад, рівність яскравості, відображення контуру, внутрішньої текстури, форми. Такі сегменти можна називати «образами», так як у них вже можна виділити такі атрибути, як положення на карті, кут нахилу, довжина або площа, але ніяких інформативних атрибутів (функція, значення) у них ще немає.

Множини пікселів одного кластера і сегменти можуть бути перетворені за допомогою апарату математичної морфології [8], щоб нівелювати прояви невиправлених спотворень для усунення розривів і зв'язування або для поділу сегментів. Апарат математичної морфології може бути використаний для поділу сегментів за своїми класами.

Концепцією розпізнавання образів є отримання відповідності між чином та об'єктом за рахунок аналізу дескрипторів образу. Кожен об'єкт також має свої дескриптори, але не всі вони можуть безпосередньо порівнюватися з дескрипторами образів. Не можна, наприклад, говорити про довжину одиниці, але можна говорити про довжину одиниці на конкретному зображенні. Таким чином, об'єкт повинен бути представлений у вигляді еталонного образу або сукупності образів. Крім характерних ознак (форма), деякі його властивості безпосередньо залежатимуть від властивостей всього зображення в загалом (кольоровість, дозвіл). Тому образи перед розпізнаванням слід нормалізувати, наприклад, привести символи однакового розміру.

Існує два основні підходи до розпізнавання – оптичний та інтелектуальний. Оптичний підхід більш простий у реалізації, має високу швидкість виконання на ЕОМ. Але на відміну від нього інтелектуальні методи точніші, мають вищі показники якості.

Оптичні методи розпізнавання розглянуті в [9, 10], серед них виділяється три підходи: шаблонний, структурний та ознаковий. Метод збігу шаблону заснований на безпосередньому порівнянні зображення з відомими шаблонами; такий метод працює добре і швидко на чистих зображеннях, але показує погані результати при відмінності форми зображення, наприклад, при використанні інших шрифтів при розпізнаванні символів тексту.

Структурні методи використовують інформацію про топологію зображення, таких як перетин ліній і наявність отворів. Такий підхід більш гнучкий по відношенню до зміни шрифту, розміру букв або повороті на деякий кут, але коли елементи зображення можуть перетинатися або розриватися, то якість може значно погіршитися.

Кращим оптичним підходом для зашумлених зображень є ознаковий, до нього відносяться такі методи, як зонування, використання инвариантов геометричних моментів, проекційних гістограм тощо. У ньому зображення ставиться у відповідність набір ознак, обчислених згідно знайденому образу.

Далі будуть розглянуті методи опрацювання цих ознак.

Найпростішим методом є класифікація за допомогою відстані: якщо сприймати образ, як вектор дескрипторів в ознаковому просторі, то його приналежність до деякого класу може бути визначена за допомогою відстані між вектором образу і вектором еталона. Такий евристичний підхід працює, якщо класи виявляють тенденцію до кластеризації. Якщо функція ймовірності відхилення образів від стандартів не залежить тільки від відстані до стандарту в ознаковому просторі, то застосовується класифікація за допомогою функції правдоподібності або байєсівський класифікатор.

Так як виділити конкретний еталон або групу еталонів часто виявляється неможливим через високу варіативність атрибутів або високу зашумленність, що викликає сильні спотворення атрибутів, доводиться приймати за де-скриптори значення колірних координат пікселів в їх зв'язках. Для таких випадків дуже хороші результати показують нейронні мережі різних топологій, для яких за допомогою машинного навчання можна в автоматичному режимі відрегулювати внутрішні параметри. В основі аналізу лежить принцип ковзного вікна: аналіз проводиться не над конкретним чином, певним своїм становищем у растрі, а над його сусідами теж, «переміщаючись» від одного до іншого, доки не буде знайдено образ, що видає максимальне значення для будь-кого класу. Цей принцип може значно прискорити розпізнавання під час використання багатоядерної чи багатопроцесорної архітектури ЕОМ.

Особливе місце також займає гібридний підхід, що полягає в поєднанні оптичного та інтелектуального підходів [11] або декількох штучних нейронних мереж [12], що призводить до підвищення якості розпізнавання та швидкості виконання алгоритмів. Класифікатори об'єднуються в комітети за допомогою бустингу [13], основна ідея якого полягає в тому, що кілька слабких класифікаторів можуть проявляти себе сильніше в комплексі.

Оптимізувати класифікатори можна за допомогою генетичних алгоритмів або алгоритму імітації відпалу [14]. У них перебираються правила

взаємодії дескрипторів, поки не буде знайдено оптимальний набір правил, що дає максимальний функціонал якості. Також поліпшення (як, так і швидкості) можна досягти за допомогою зменшення числа дескрипторів, так як деякі з них самі можуть бути шумами.

1.3 Концепція та методи штучного інтелекту у вирішенні задачі розпізнавання образів

Застосування штучного інтелекту (ШІ) ставить перед собою проблему автоматизації виконання інтелектуальної роботи людини за допомогою імітації складних психічних та нейрофізіологічних процесів. Інтелектуальна система повинна мати здібності:

- до сприйняття інформації;
- до навчання на основі досвіду;
- до адаптації у нових ситуаціях;
- до розуміння та застосування абстрактних концепцій та знань.

Для вирішення завдання розпізнавання образів ці здібності відповідають процесам зчитування інформації з ССЗ, пошуку та розпізнавання образів, стабільності розпізнавання при виникненні образів схожих відтінків та форм, використання бази знань образів. Без використання інтелектуалізації алгоритми стають сприйнятливими до різних шумів і спотворень даних, що особливо впливає при обробці зображень.

Теорія ШІ використовує різні методи досліджень, такі як моделювання міркувань, обробка природної мови, інженерія знань, машинне навчання. Усі вони використовуються для розпізнавання зображень.

До методів моделювання міркувань входить теорія прийняття рішень, що використовується, коли необхідно на основі деяких даних та моделі правил визначити прийняти рішення щодо поставленої проблеми. Такий проблемою при розпізнаванні може бути узгодженість даних, наприклад, на деякій відстані один від одного стоять позначки висот, які розпізнав

алгоритм. Якщо обчислений кут нахилу буде більшим критичного значення, означає, що якесь число або обидва були розпізнані невірно. Або алгоритм розпізнавання образів може видавати як результат, а й упевненість у ньому, і якщо ступінь впевненості низька, то ця інформація також може бути передана оператору для ручного аналізу.

Обробка природної мови потрібна при розпізнаванні назв географічних об'єктів, що призведе до можливості об'єднання двох образів загальної природи через перевірку рівності їх топонімів як атрибутивних даних. Інженерія знань дозволяє перенести знання про об'єкт дослідження з інтуїтивно розуміються людиною і оброблюються автоматично, поза свідомістю, формальною мовою ЕОМ.

Для налаштування параметрів інтелектуальних моделей використовується машинне навчання. Існують різні способи машинного навчання в залежності від того, що взято за його основу, і який класу завдань необхідно вирішити. У загальному випадку вхідні дані для машинного навчання є впорядкованою безліччю n значень, що об'єднуються в вектор $\boldsymbol{x}=(x_1,\ldots,x_n)$.

Для «навчання з учителем» алгоритм вводять набір пар значень «ситуація – необхідне рішення». Типовим завданням для такого способу є розпізнання символів, так як для деякої множини пікселів дослідник задає значення символу, відображеного на цій множині. У загальному випадку це завдання називається класифікацією. Розрізняють бінарну та багатокласову класифікацію; класи можуть бути такими, що не перетинаються, перетинаються і нечіткими.

Для навчання без вчителя в алгоритм вводять тільки набір ситуацій. У задачі аналізу зображень такий спосіб застосовується для його кластеризації, тобто поділу множини елементів, що володіють деякими ознаками, на підмножини в ознаковому просторі деякі однорідні групи. Для ССЗ кластеризація, відповідно, виконується для поділу кольорів пікселів по групах. Даним способом можна знизити розмірність і варіативність даних,

оскільки мільйони можливих відтінків кольорів замінюються кластерами невеликої кількості кольорів. Проблемою кластеризації є необхідність знання кількості кластерів

Для виміру узгодженості обчислених рішень реальності запроваджується поняття функціоналу якості алгоритму. Для «навчання з учителем» функціоналом є середня помилка невідповідності. Якщо намагатися мінімізувати середню помилку з навчальної вибірці даних, можливо отримання ефекту «перенавчання», коли система стає нездатною до передбачення реакцій на нові ситуації. Для «навчання без вчителя» функціоналом може бути, наприклад, середня внутрішньокластерна або міжкластерна відстані (для кольорів це може бути декартова відстань між точками в колірному просторі центрів мас кластерів).

Щоб уникнути перенавчання, використовується крос-валідація. При крос-валідації навчальна вибірка поділяється на *k* частин, у своїй навчання проводиться на *k-1* частини, а тестування – на що залишилася. При цьому виникає *k* незалежних моделей, результат дії яких знаходять по самому зустрічається класу.

Одним із актуальних завдань машинного навчання є так зване «прокляття розмірності», коли при великій кількості вхідних даних стає неможливим отримання вирішальної моделі з достатньою точністю. Щоб вирішити цю проблему необхідно, по-перше, знизити розмірність вихідних даних (через кластеризацію або відкидання малоінформативних ознак), а по-друге, по можливості збільшити навчальну вибірку.

За наявності групи еталонів (особливо близько розташованих) часто виникає ситуація, коли образ належить відразу кільком класам. Для таких випадків застосовуються такі універсальні методи: пошук найближчих k сусідів, метод опорних векторів, метод вирішальних дерев.

1.4 Сучасні методи попередньої обробки та післяобробки цифрових зображень

Розглянемо методи попередньої обробки цифрових зображень. Початок роботи із складноструктурними зображеннями передбачає аналіз того, які недосконалості яскравіше позначаються на їх якості. Це важливо, оскільки низькорівневі перетворення попередньої обробки дуже чутливі до найменших варіацій графічних властивостей. Цей аналіз також є підставою для подальшої розробки програмного забезпечення з обробки цифрових зображень.

Крім формальних математичних способів колірної корекції зображень (виправлення яскравості, контрасту, кольоровості тощо) також розроблені методи, що застосовують більш орієнтовані на об'єкти, а не пікселі підходи, наприклад, аналіз спектрально-контурних елементів.

Перед початком попередньої обробки ми маємо цифрове зображення, яке потрібно перетворити на набір об'єктів. Розроблено адаптивний низькочастотний фільтр, що пригнічує цифровий шум, і завадостійкий аналізуючий алгоритм, що працює на зображеннях, виконаних в умовах недостатньої освітленості. Схожими проблемами локально мають і топографічні карти, але в роботі не порушені питання подальшого розпізнавання образів на зображеннях.

Інформація про об'єкти перш за все визначається їх становищем і межами: навіть не знаючи, якого кольору образ, яку функцію він може нести, ми можемо виділити межі декількох образів, звичайно, якщо зображення вже позбавлено шумів, так як вони можуть бути інтерпретовані, як краї. Для виділення кордонів приймаються так звані оператори, найпростіші: Собеля, Прю-ітт, Робертса та лапласіана гауссіана. На основі оператора Собеля побудовано метод виділення контурів об'єктів з розмитими межами сплайн-функціями та метод формування лінійних контурів на аерофотознімках.

Більш просунутий детектор, розроблений Дж. Ф. Кенні, дозволяє отримувати кордони з кращою якістю, але розмиває дрібні деталі і утворює «хибні» контури, тому непридатний для топографічних карт без додаткової модернізації. Алгоритм, який використовує вейвлет-перетворення, навпаки, практично не утворює «хибних» контурів, але створює розриви в кривих кордонів, які також необхідно видаляти спеціальними алгоритмами.

Часто для виявлення меж використовується перетворення Хафа. За допомогою нього на зображенні можна знаходити фігури певного класу, задані деякою моделлю, роблячи перетворення двовимірного цифрового зображення в дискретний простір параметрів, що визначаються моделлю. Наприклад, якщо ця модель лінія, то простір параметрів також двовимірно, параметрами є кут нахилу лінії та відстань від неї до центру координат. Якщо на зображенні присутня лінія з даними параметрами, то в параметричному просторі це відповідає більшому значенню.

Крім статистичних детекторів розроблено детектор, що використовує ШНС. Даний детектор адекватно працює на зображеннях непостійного розмаїття і розмитих кордонів, як на топографічних картах. Недоліком даного методу є використання великої нейромережі, в якій на кожен піксель зображення припадає по 3 нейрони, що нездійсненно для топографічних карт розмірів в кілька мільйонів пікселів. Рішенням є використання нейромережі на меншу область і застосування методу ковзного вікна.

Згорткові ШНМ застосовуються в основному для розпізнавання великих за довжиною і складних за кольором об'єктів, наприклад осіб або класів об'єктів при здійсненні дорожнього руху. Використання їх при обробці, наприклад, кольорових зображень низької роздільної здатності недоцільно, оскільки неточно-сті при зображенні маленьких об'єктів дуже впливають.

Також кластеризація може виконуватися не за допомогою одного обчислювача, а за допомогою декількох окремих агентів, наприклад, як

алгоритм бджолиної колонії. Даний алгоритм дозволяє легше розпаралелити процес, але є дуже вимогливим до апаратного забезпечення.

Після виділення кордонів необхідно виділити образи різної функціональності на топографічній за допомогою кластеризації кольорів. Алгоритм k-means дозволяє розбити багато кольорів карти на групи, зменшуючи сумарне квадратичне відхилення точок кластерів від точок центрів цих кластерів. Побудовані алгоритми, що прискорюють роботу стандартного алгоритму Ллойда, за допомогою введення деревоподібних структур даних або використанням кластерів, що розділяються, і медіанного фільтра.

Розроблено фільтр селективної уваги, який виконує розбивку карт землекористування на шари, виділяючи особливості, що узгоджуються з результатами поділу безлічі кольорів карти, виконаного за допомогою методу нечіткої кластеризації C-means. Регулюючи різні параметри алгоритму, і, враховуючи контекст (вплив окремих верств один на одного), можна здійснити вибірку необхідних концептуальних шарів. Досліджувані в роботі карти землекористування орієнтовані на аналіз забудованого простору, мають більшу різноманітність контрастних кольорів, що дозволяє використовувати спрощений метод кластеризації, непридатний для топографічних карт.

Наступною операцією після кластеризації є сегментація згідно з деякими графічними ознаками. Основні техніки сегментації розрізняють пороговий підхід, методи, засновані на стисканні, використанні гістограм або знайдених контурів, методи вирощування областей, графові методи, методи вододілу, багатомасштабні методи. Крім цього останнім часом розвиток отримав об'єктно-орієнтований похід, що полягає в сегментації за рахунок пошуку конкретних елементів.

Так як порогові методи досить негнучкі (на різних картах порогами кольорів можуть бути різні значення), то був розроблений метод, заснований на концепції адаптивного порогу, що володіє таким значенням, щоб

внутрішньокласова дисперсія кольору була мінімальною, який був модернізований на випадок декількох класів. Також запропоновано метод, заснований на обчисленні порога згідно з значенням нечіткої ентропії.

Метод водоподілу укладено у поданні операції сегментації як процесу поділу води всередині пікселів-вододілів, що мають максимальне значення градієнта яскравості. Вода, розміщена на будь-який піксель усередині загальної лінії вододілу, тече вниз до локального мінімуму яскравості. Пікселі, від яких вода стікає до загального мінімуму, і являють собою обчислений сегмент. Даний метод не дозволяє контролювати гладкість отриманих сегментів, тому був удосконалений у декількох дослідженнях.

Метод вирощування областей та багатомасштабного аналізу полягає в розростанні областей-сегментів на основі центрів кристалізації і поступовому переході від локальних областей до глобальних. Вони дозволяють адекватно працювати на зображеннях, де існує безліч зв'язків, таких як топографічні карти, використовувати не тільки яскраву або колірну, а й текстурну інформацію. До недоліків методів можна віднести високі вимоги до ресурсів ЕОМ.

Одним з найважливіших апаратів для роботи із зображеннями, особливо в цілях сегментації, є математична морфологія. Морфологічний аналіз - це аналіз геометричної структури зображення, заснований на теорії множин і топології. В основі апарату - виконання певних операцій над кожним пікселем зображення. Основними операціями є: перенесення, дилатація та ерозія, замикання та розмикання; вони виконуються за допомогою структурних елементів (невелике зображення), за допомогою якого визначається вплив значення пікселя на своїх сусідів.

Класичний спосіб використання математичної морфології наведено в дослідженні, що автоматизує отримання логічних ланцюгів із сканованих документів. Тут морфологія з одного боку використовується у тому, щоб позбутися шумових пікселів, з іншого – щоб виділити об'єкти, мають певну орієнтацію і форму.

Даний підхід також корисний, коли відбувається робота з моделями, для яких виділити дескриптори дуже складно або неможливо, наприклад, як при обробці карт для знаходження та аналізу лісової території. Розроблений метод досить простий, але дозволяє розрізняти риштування різного типу між собою.

Одним з термінів математичної морфології є скелет – множина точок, рівновіддалених від меж фігури. Скелети використовуються, коли потрібно привести криву до форми, коли її товщина складатиметься лише з одного пікселя. Отримані скелети не можна безпосередньо використовувати в розпізнаванні, але вони є адекватними спрощеними моделями кривих, що збігаються з ними за розташуванням і формою і відрізняються лише товщиною ліній. Хвильовий алгоритм виконується ітераційно всередині вже певного сегмента: скелет являє собою криву, утворену центром сферичної хвилі, що розповсюджується по сегменту.

Аналіз сцен – фінальний етап розпізнавання образів, розуміння як характеристик самих об'єктів на зображенні, а й їхні стосунків загалом. Тільки тут можливе найбільш повне розуміння сукупності отриманої інформації та пошук помилок, скоєних на попередніх етапах. Ніякий метод не дає абсолютної якості розпізнавання, але воно можливе при комплексній взаємодії аналізованих образів.

В основі аналізу сцен лежить принцип узгодженості або використання метаінформації про образи, наприклад, ми знаємо, що букви в словах або цифри в числах розташовані поруч, на одній прямій, відображені за допомогою одного шрифту, розміру та кольору. Так само ми можемо сказати і про інші пов'язані образи, наприклад, двовимірні об'єкти гідрографії має рівну альтитуду по всій поверхні, горизонталі не можуть перетинатися і т.д. буд. Знаходження таких неявних правил і використання їх при розпізнаванні, тобто використанні угруповання розпізнаних об'єктів, що належать до різних семантичних планів, сприяє поліпшенню якості та розуміння зображення в цілому.

Алгоритм угруповання літер у слова на топографічних картах використовує принцип узгоджених обмежень: рівність кольору та рівність розміру. Всі ці обмеження дозволяють відокремити різні рядкові позначення один від одного, але не дозволяють працювати з назвами, розташованими за кривими, наприклад, назви річок.

Модернізація розпізнавання топонімів на топографічних картах може бути за допомогою накладання додаткових зв'язків на літери: геометричних чи лексичних. Додатковими обмеженнями є частота літер вздовж напрямку слова, параметри шрифту. Алгоритм для розпізнавання рукописних слів в зв'язковий текст заснований на прихованої марківської моделі, що навчається (HMM), і порівнюється з алгоритмом динамічної трансформації часової шкали (DTW). У розробленому алгоритмі не використовується угруповання букв, а рядки різної довжини, і алгоритму неважливо розуміння алфавіту і словника.

Важливим питанням при розпізнаванні образів на ССІ є врахування неоднорідності фону: символи та лінії можуть лежати на кількох двомірних об'єктах різного кольору, розпізнавання при цьому різко падає. Єдиною моделлю, що адекватно працює на змішаному тлі, є нейромережі.

1.5 Оцінка алгоритмів

Тільки на кінцевому етапі обробки цифрового зображення можна визначити якість безпосередньо через ставлення правильно розпізнаних об'єктів до кількості об'єктів. Оцінювання проміжних результатів розпізнавання, таких як сегментація символів і шматочків ліній або кластеризація кольорів, що використовують різні методи, не є коректним, оскільки велика помилка при використанні одного методу може нівелюватися в післяобробці, а менша, при використанні іншого, – ні.

Тим не менш, залишати оцінку методів до фінального етапу не є відповідним, особливо у випадках, коли провести її автоматично повністю

неможливо, як у випадку з ССЗ зі слабкою формалізацією. Перевірити працездатність методів розпізнавання текстових документів набагато простіше, адже крім документа, як множини пікселів, є також і документ, як упорядкована безліч символів, які можна поставити у відповідність розпізнаним образам.

Способи оцінювання для різних моделей машинного навчання розглянуті в багатьох роботах, зазначені варіанти можливих оптимізацій та зниження показників помилок. Показані сучасні способи оцінювання якості роботи класифікаторів, детекторів країв, процесів сегментації. Головними питаннями тут є адекватність цих критеріїв та методи оцінювання якості процесів постобробки для розпізнавання, оскільки у цій галузі вони мало вивчені.

Одним із найпоширеніших загальних принципів оцінки якості для бінарної класифікації (в рамках перевірки нульової H0 і альтернативної H1 гіпотези) є частка правильних відповідей (accuracy), повнота (recall), точність (precision), оцінка помилки першого (false positive), другого (false negative) роду та їх комбінація.

На жаль, дані метрики не відображають стійкості моделей розпізнавання, тому що їх навчання (особливо якщо вони є інтелектуальними) відбувається, по-перше, при випадковому виборі даних для навчання, а, по-друге, порядок даних для навчання теж вибирається випадково. Наприклад, при запуску навчання ШНМ точність розпізнавання може змінюватись від запуску до запуску в районі 10-15%, тому важливою метрикою є ще й міра розкиду показників якості.

Щоб підвищити стабільність інтелектуальних моделей, використовується метаалгоритм бегінга (bootstrap aggregating). Його ідея полягає у генерації великої кількості інтелектуальних моделей. Початковий тренувальний набір для кожної моделі модифікується взяттям із загальної вибірки із поверненням.

Важливим критерієм якості є не так кількість помилок, але відносна кількість помилок на об'єм зображення. Розмір ССЗ, особливо карт, найчастіше вимірюється у мегапікселях. Цей параметр важливий, оскільки можливо доведеться порівнювати алгоритми, які проходили експериментальну перевірку на зображеннях з різною роздільною здатністю, тобто з різним dpi. При збільшенні dpi збільшується її розмір у мегапікселях, при збереженні фізичного розміру карти. Алгоритми при цьому мають давати меншу помилку.

Для визначення якості роботи алгоритмів, що видають двовимірні області, використовується міра Жаккара або Intersection over Union (IoU), що розуміється як відношення потужності безлічі перетину вірної та пропонованої області до потужності безлічі їх об'єднання.

Швидкість роботи алгоритмів також необхідно вважати щодо розміру зображень, оскільки збільшення розміру зображення в 2 рази має призводити до збільшення часу обробки в 2 рази за збереження апаратного забезпечення. На жаль, у багатьох сучасних дослідженнях питання часу виконання обробки не поставлено; у рідкісних дослідженнях зазначено апаратне забезпечення, на якому відбувалася перевірка алгоритмів.

Отже, можна зробити такі висновки із проведеного аналізу. У дослідженні проведено аналіз існуючих моделей та алгоритмів обробки цифрових зображень з точки зору концепції складноструктурного зображення. Проведено класифікацію зображень за категоріями відображуваних об'єктів і з'ясовано ознаки складноструктурності. Були досліджені основні методи і моделі, що використовуються для попередньої обробки цифрових зображень, розпізнавання образів, розуміння зображення та принципи оцінювання цих методів.

У розглянутих дослідженнях представлено велику кількість різноманітних методів та моделей для обробки зображень. Використання конкретних методів може давати хороший результат при обробці зображень, що мають вузьку спрямованість. При цьому ССЗ тяжіють до консолідації

графіки різного типу та гетерархічності семантичних планів, що призводить до того, що необхідно використовувати гібридні та ітеративні алгоритми, які були знайдені лише в невеликій кількості публікацій, що говорить про слабку розробленість теми.

Більшість досліджень розглядає лише окремі кроки обробки ССЗ, без комплексного вирішення завдання, залишаючи цю роботу оператору ЕОМ. Без вирішення комплексних завдань неможлива подальша автоматизація та автоматичний пошук помилок. Не виявлено робіт, що передбачають спільну обробку образів, що належать до різних семантичних полів. Аналіз неявних правил відображення об'єктів або аналіз впливу порушення цих правил виражений лише в малій частині робіт.

У більшості проаналізованих досліджень не відображені часова складність алгоритмів, конкретний час виконання та апаратне забезпечення. При популярності часу воно зазвичай є досить великим через використання складних інтелектуальних моделей і неоптимальних алгоритмів поліноміального часу високого порядку. Точність і стабільність алгоритмів також є важливими критеріями якості, але сумарна кількість часу роботи ЕОМ і оператора є ключовим у плані ефективності.

2 АНАЛІЗ ТА ОБРОБКА СКЛАДНОСТРУКТУРНИХ ЗОБРАЖЕНЬ

2.1 Методика аналізу і обробки складноструктурних зображень

Інформацію у вигляді цифрового зображення можна подати у вигляді такого відображення

$$f : E \to P, \tag{2.1}$$

де E – деякий опис зовнішнього світу; P – цифрове зображення, задане у вигляді матриці.

Таким чином, основною метою аналізу та обробки ССЗ є розробка зворотного відображення

$$f^{-1} : P \to E_1, \quad E_1 \subseteq E, \tag{2.2}$$

де E_1 – необхідна частина опису зовнішнього світу, наприклад, властивості та розташування певних об'єктів на зображенні.

Вхідне відображення f не є оборотним у загальному випадку, оскільки, по-перше, є функцією згортки, що перетворює вхідні дані довільного розміру на матрицю встановленого. Тим не менш, для подальшої роботи (наприклад, прийняття рішень) отримання повної та точної інформації про зовнішній світ часто не потрібно: достатньо знати її з певною похибкою.

По-друге, правила відображення f можуть мати як явний, так і неявний характер. Під явними правилами мається на увазі однозначне відображення об'єктів зовнішнього світу та їх властивостей візуального плану, наприклад, зображення будинку на карті у вигляді умовної піктограми або спрощеного кадастрового плану. Під неявними правилами можна мати на увазі як багатозначні відносини, коли елементи зображення не мають конкретної прив'язки в просторі растру, або виражені у вигляді неявних функцій, що зв'язують частини

$$P : R\,(p_1, \ldots, p_k) = 0, \tag{2.3}$$

де p_i – елемент зображення (ЕЗ) , або його частини;

$$E : R\,(e_1, \ldots, e_k) = 0, \tag{2.4}$$

де e_i – об'єкт або система об'єктів зовнішнього світу, або більш складного відношення $R\,(p_1, \ldots, p_k, e_1, \ldots, e_k) = 0$.

Багатозначні відносини, як можна побачити на прикладі з'ясування відповідності підписів та піктограм міст на середньовічних картах, призводять до вирішення комбінаторної NP-повної задачі. Неявні відносини можуть бути виражені складними характеристиками, що обчислені на невизначеному числі елементів зображення, наприклад, обмеження на кількість міток на 1 дм^2 карти, або накладення один на одного кількох візуальних планів, або породжені фізичними законами зовнішнього світу, наприклад динамічні процеси, що передують фіксації статичного зображення, можуть нести додаткову інформацію. Прикладом неявного відношення, що поєднує в собі як елементи зображення, так і об'єкти зовнішнього світу, можна вважати псування фізичного носія зображення, яке зберігається під час сканування та переведення в цифровий формат.

Чим більше відношення явних правил у відображенні f над неявними, тим простіше виконати зворотне відображення. Якщо явні правила зазвичай виражені у вигляді певних стандартів, які поширюються на широкий клас зображень, то неявні правила можуть залежати від багатьох речей. Наприклад, ці правила можуть бути залежними від переваг оператора, що відповідає за створення зображення P або від складності зовнішнього світу, що відображається (як приклад, вирішення завдань ідентифікації по фото паспорту та ідентифікації по камерах відеоспостереження в метрополітені).

Для мінімізації неявних правил і спрощення їх вирішення необхідно нівелювати вплив ЕЗ одне на одного, що досягається зменшенням їх розмірів, які розглядаються, за допомогою декомпозиції. Для цього використовуватимемо висхідний підхід до створення загальної методики

обробки ССЗ: аналіз та обробка відбуватиметься від окремих пікселів до зображення в цілому. Узагальнену методику показано нарисунку 2.1.

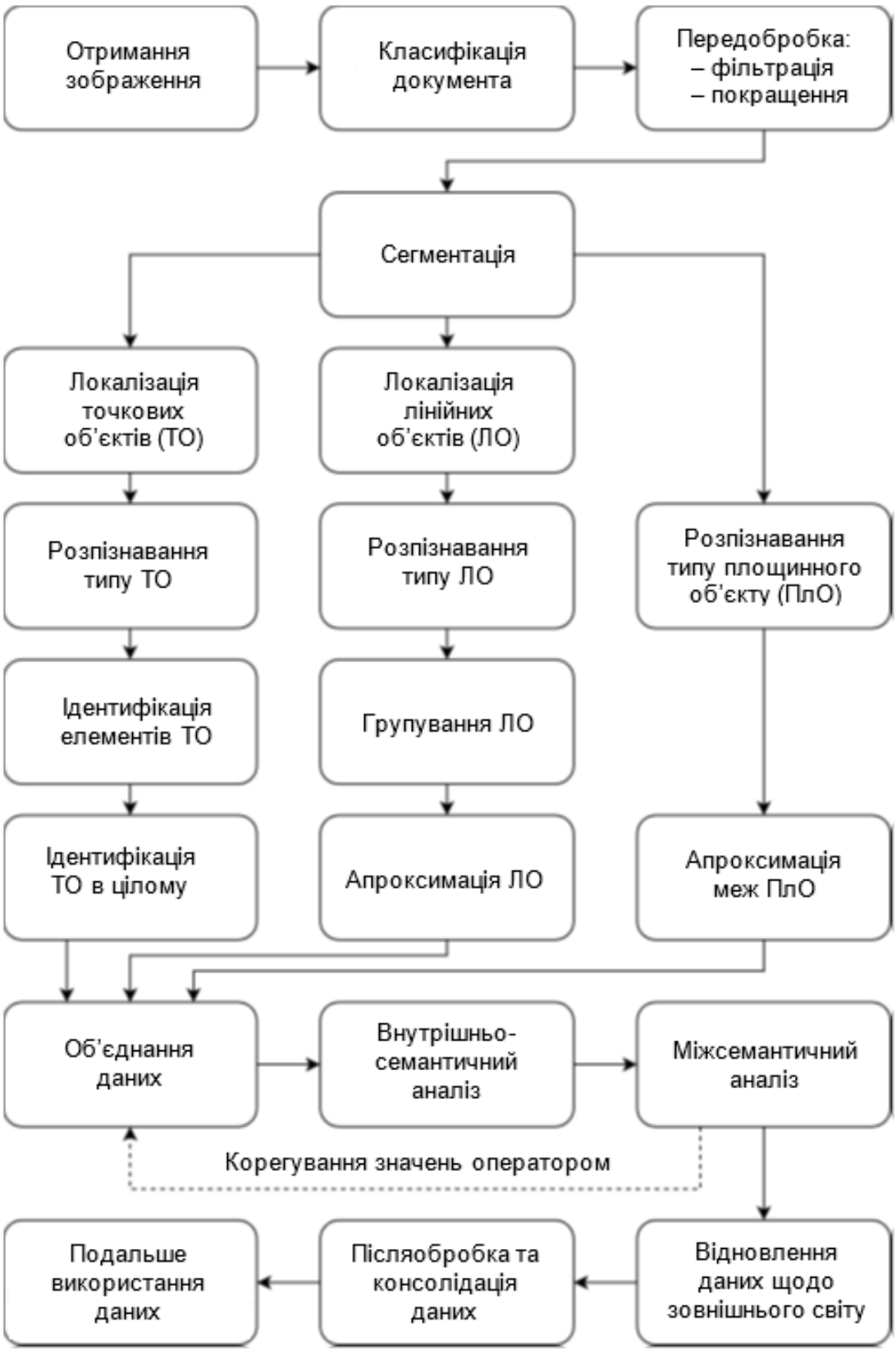


Рисунок 2.1 – Узагальнена методика програмного аналізу та обробки складноструктурних зображень

Після отримання цифрового зображення необхідно класифікувати його, так як спеціальні документи мають на увазі різну подальшу обробку. на цьому ж етапі відбувається поділ документа, якщо це необхідно на піддокументи, наприклад, відділення текстовий частини від ілюстрації.

Передобробка в здебільшого полягає в фільтрації від шумів, очищення від макроскопічних об'єктів і кольоровому покращення зображення. Поліпшення може полягати в нормалізації контрастності, яскравості, виконання необхідних зміщень в кольоровому просторі. Передобробка в здебільшого виконується попіксельно або з допомогою ковзаючих вікон невеликого розміру для виконання сверткових операцій.

Сегментація поділяє множину растру на об'єкти-сегменти, що мають точкову (точковий об’єкт, ТО), лінійну (лінійний об’єкт, ЛО) або площинну (площинний об’єкт, ПлО) розмірність. Всі об'єкти на двовимірному растрі можна вважати ПлО, тому під ТО передбачаємо об'єкти, розмір яких у пікселях не змінюється при переході до зображень інших масштабів (можна говорити про клас O(1) на витрати пам'яті на зберігання окремого ТО).

Під ЛО припускаємо об'єкти, розмір яких у пікселях змінюється пропорційно змінюваному масштабу (можна говорити про клас $O(\sqrt{S})$ на витрати пам'яті на зберігання окремого ЛО, де S – площа растру).

Розподіл об'єктів за розмірністю пояснюється тим, що для кожного з них існують специфічні способи обробки: ТО та ЛО тяжіють до більш інтелектуальних методів, ПлО – до більш статистичних.

Після сегментації відбувається етап розпізнавання об'єктів, кожен із яких специфічний за розмірністю. При необхідності прості об'єкти групуються у складніші і відбувається їх спрощення. Далі дані збираються за семантичними планами і відбувається їх комплексна обробка: спочатку внутрішньо-семантична, потім міжсемантична . На етапі комплексного аналізу в оброблених даних знаходяться помилки та виходить відсутня інформація.

Наприкінці відбувається відновлення даних про зовнішній світ для всієї

множини растру, наприклад, інтерполяція, на етапі післяобробки відбувається додаткова прив'язка і консолідація даних з інформацією з інших документів. Надалі прикладне програмне забезпечення може виконувати візуалізацію, статистичний і інший аналіз для підтримки прийняття рішень або перетворення векторизованих даних в інші типи.

Якщо автоматизувати таку методику в цілому неможливо через висоий ступень впливу об'єктів друг на друга або особливостей оброблюваних документів, то необхідно залишити можливість виправлення даних оператору (на схемі рисунку 2.1 відображено коригування тільки для частини, котра відповідає за комплексний аналіз).

2.2 Метод багатометкової сегментації складноструктурних зображень

У підсумку при обробці складноструктурних зображень потрібно перейти до отримання та аналізу об'єктів. Історично склалися два підходи до реалізації такої «об'єктною» сегментації:

- екземплярна, що заснована на поділі об'єктів;
- семантична, що заснована на поділі класів об'єктів.

Ні той, ні інший підхід не може надати достатньої якості при аналізі ССЗ. Перший підхід спирається на ідею обробки об'єктів - речей (things), які можуть в значному ступені накладатися друг на друга (немає строгого відокремлення). Другий підхід спирається на ідею обробки об'єктів - регіонів (stuff), які на ССЗ можуть виявлятися з велико. варіативністю.

Відтепер активно формується паноптичний підхід, який не поділяє два вищезазначених поняття і тому є більшегнучким. Основна його ідея полягає в використанні найпростіших обчислювачів для формування непересічних суперпікселів. При цьому все одно вважається, що окремий піксель може належати тільки одному класу, що не відповідає властивості гетерархічності ССЗ, тому пропонується метод багатометкової сегментації, при якому піксель відповідає класам об'єктів з деякою мірою.

Для опису окремого пікселя на зображенні у випадку, коли кожен об'єкт визначається одним кольором, використовується така формула:

$$p = \sum_{i=1}^{n} w_i \cdot C_i + E, \tag{2.5}$$

де p – вектор кольору пікселя у просторі RGB, координати якого приймають дискретні значення, наприклад, [0, 255]; n – кількість класів об'єктів зображення; C_i – вектор кольору i-го класу об'єкта; E – вектор зашумленості (може мати від'ємні координати) ; w_i – вагові коефіцієнти при векторах кольорів.

Хоча даний випадок і є найпростішим, але на ССЗ з великим числом перетинів його можна використовувати в якості основи, так як навіть складні об'єкти часто представляються у вигляді набору одноколірних об'єктів.

Завданням багатометкової сегментації є визначення вагових коефіцієнтів для кожного типу об'єкта. Працюючи з кольоровими зображеннями це завдання можна розв'язати загалом за допомогою формальних математичних методів не більше, ніж для трьох класів об'єктів.

Якщо об'єкт не можна представити у вигляді сукупності об'єктів одного кольору, то можна висунути більш загальну формулу, в якій враховується різниця координат між пікселем, що розглядається, і центром об'єкта (Δ, Δ) та інші властивості об'єкта, наприклад, кут нахилу і масштаб:

$$p = g\left(\sum_{i=1}^{n} w_i \cdot C_i\left(\Delta x_i, \Delta y_i, a_i, s_i\right) + E \right), \tag{2.6}$$

де g – загальна функція.

У даній формулі можна врахувати і зміни ССЗ, як зміну яскравості чи насичення, за допомогою загальної функції g.

З аналізу формули (2.6) можна зробити висновок щодо неможливості проведення сегментації без знання інформації про розташування об'єктів. Хоча ця інформація згідно розробленій методиці (рис. 2.1) виходить при

наступних етапах обробки, але вона може бути отримана і без проведення етапу сегментації. Це підтверджується численним використанням алгоритмів R-CNN і YOLO при обробці фото- і відеоматеріалів. Проблема даних алгоритмів полягає в їх локальності: вони можуть бути застосовані тільки для знаходження ТО, але не для ЛО або ПлО.

Враховуючи попередні зауваження щодо якості ССЗ, наявності на них об'єктів певних типів та дослідження методів сегментації, було зроблено висновок про необхідність використання інтелектуальних методів для сегментації, що є класифікацією окремих пікселів. Так як для визначення, чи є даний піксель частиною деякого об'єкта, необхідно знати оточення даного пікселя, то для аналізу можна використовувати метод ковзного вікна. Вікно розміром $a \times a$ ($a \in \mathbb{N}$, $a > 1$) пікселів призведе до створення $3a^2$ значень вхідних даних, які необхідно проаналізувати з урахуванням різних напрямків. Квадратне вікно вибрано, виходячи з можливості зображення об'єктів під будь-яким кутом.

Оскільки нам необхідно лише визначити, чи належить цей піксель деякому *i*-му типу об'єктів, то вихідний шар інтелектуальної моделі має бути функцією активації *softmax*:

$$\sigma_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}, \tag{2.6}$$

де K – кількість класів, z – dектор розмірності K.

Значення цієї функції (що може лежати в інтервалі [0, 1]) для кожного i визначає міру того, що піксель належатиме класу об'єктів i.

Сума всіх значень σ дорівнює одиниці. Таким чином, зображення після проведення даної багатометкової сегментації перетворюється на напівтонових зображень, що пред'являє підвищені вимоги до пам'яті системи за наявності великої кількості класів. Зберігання даних растрів можна

скоротити, якщо використовувати розряджені матриці, до яких можна перейти, якщо переініціалізувати низькі значення нуля , наприклад $\sigma_I \leq 0{,}1$.

Наприклад, сегментація цифрових топографічних карт (ЦТК) з метою визначення рельєфу має на увазі існування об'єктів трьох класів.

Перший – це об'єкти відображення рельєфу: горизонталі, бергштрихи, написи на горизонталях; вони зображуються коричневим кольором.

Другий – точкові геодезичні об'єкти: позначки висот (зокрема командних), геодезичні символи; вони зображуються чорним кольором.

Третій тип (фоновий) – пікселі, які відповідають іншим об'єктам.

Для початку необхідно навчити інтелектуальну модель за допомогою навчальних даних, для цього необхідно безпосередньо на зображенні розмітити пікселі. Через великий розмір ЦТК провести повністю цю операцію неможливо, тому необхідно забезпечити репрезентативність вибірки. Вона має відображати всі можливі прояви зображення умовних знаків: на різному тлі, під різними кутами, знаки повинні мати різне значення. Особливу увагу слід приділити областях, де перетинаються об'єкти різних класів.

Після складання навчальної вибірки необхідно розділити її на три групи: перша група буде навчальною, друга валідуючою, а третя – тестовою.

Валідуюча вибірка необхідна обчислення генералізації навчання й зупинення навчання, коли генералізація припиняє поліпшуватися.

Тестова група потрібна для перевірки системи на можливе перенавчання: для таких параметрів інтелектуальних моделей показники помилок будуть значно вищими у тестових вибірок.

Алгоритм навчання інтелектуальної моделі (на прикладі штучної нейронної мережі прямого поширення з одним прихованим шаром) для багатометкової сегментації показаний в модулі, програмний код якого наведений на рисунку 2.2, за допомогою псевдокоду на основі мови програмування Matlab, в якому опущені несуттєві для розуміння команди.

Вхідними параметрами для даного модуля є такі:

- вхідне цифрове зображення;
- розмічене зображення;
- вектор кольорів для розмічених K класів;
- розмір ковзного вікна.

Вихідним параметром є навчена інтелектуальна модель.

```
1:    function net = trainModel(picture, labeledPicture, c, a)
2:       counter = 0
3:       points = zeros(2, sum(picture ~= labeledPicture))
4:       output = zeros(size(c, 1), sum(picture ~= labeledPicture))
5:       for i = 1 : size(c, 1)
6:       |  pictureOfIthClass = (labeledPicture(:, :, 1) == c(i, 1))
                              & (labeledPicture(:, :, 2) == c(i, 2))
                              & (labeledPicture(:, :, 3) == c(i, 3))
7:       |  [x, y] = find(pictureOfIthClass)
8:       |  points(1 : 2, counter : counter + size(x)) = [x, y]
9:       |  output(i, counter : counter + size(x)) = 1
10:      |  counter = counter + size(x)
11:      end
12:      input = getNeighborhood(picture, points, a, 0) / 255
13:      net = patternnet(hiddenLayerSize)
14:      initParameters(net);
15:      net = train(net, input, output)
16:   end
```

Рисунок 2.2 – Модуль навчання інтелектуальної моделі

Допоміжним для даного алгоритму та багатьох інших алгоритмів далі є модуль (рисунок 2.3), який за відомою множиною точок знаходить опис усіх сусідніх пікселів у вікні розміру $a_x \times a_y$, поверненому на кут φ.

Опис кожної точки є вектором, у якому спочатку йдуть значення крайнього лівого верхнього сусіда.

У першій частині програми навчання інтелектуальної моделі (рядки 2.. 12) знаходяться розмічені пікселі та створюється розряджена матриця, яка ставить у відповідність координатам пікселів значення класу. Потім для цих пікселів формується опис їхніх сусідів. Байтові значення колірних координат з інтервалу [0, 255] нормалізуються до інтервалу [0, 1] підвищення стабільності моделі. Часова складність цієї частини дорівнює $O(K \cdot S + a^2 \cdot N_{label})$, де N_{label} – кількість розмічених пікселів.

У другій частині навчання інтелектуальної моделі (рядки 13..15) відбувається безпосереднє навчання за допомогою сполученого градієнтного

алгоритму зворотного поширення помилок, так як при ньому відбувається більш ефективне використання пам'яті, що використовується, ніж при алгоритмах типу Левенберга-Марквадта або байесівської регуляції. Особливість даного методу – вибір зміни параметрів: воно вибирається таким чином, щоб бути ортогональним до попередніх напрямків. Як критерій оптимізації обрано перехресна ентропія.

```
1:    function result = getNeighborhood(picture, points, a, phi)
2:       depth = numel(picture) / (size(picture, 1) * size(picture, 2))
3:       ax = a(1)
4:       ay = a(2)
5:       result = zeros(depth * ax * ay, size(points, 2))
6:       [dx, dy] = meshgrid(-(ax - 1) / 2 : (ax - 1) / 2,
                             -(ay - 1) / 2 : (ay - 1) / 2)
7:       picture = padarray(picture, [ax, ay, 0], 0)
8:       points(1, :) = points(1, :) + ax;
9:       points(2, :) = points(2, :) + ay;
10:      for i = 1 : size(points, 2)
11:      |  for j = 1 : ax * ay
12:      |  |  cx = round(points(1, i) + dx(j) * cos(phi) - dy(j) * sin(phi))
13:      |  |  cy = round(points(2, i) + dx(j) * sin(phi) + dy(j) * cos(phi))
14:      |  |  if depth == 3
15:      |  |  |  result(3 * j - 2 : 3 * j, i) = [picture(cx, cy, 1),
                                                  picture(cx, cy, 2),
                                                  picture(cx, cy, 3)]
16:      |  |  elseif depth == 1
17:      |  |  |  result(j, i) = picture(cx, cy)
18:      |  |  end
19:      |  end
20:      end
21:   end
```

Рисунок 2.3 – Модуль знаходження інформації про сусідні пікселі

Вихідне зображення при необхідності аугментується за рахунок додавання шуму або виконання інших кольорових маніпуляцій для підвищення стабільності роботи інтелектуальної моделі.

Часова складність другої частини залежить від обраної інтелектуальної моделі. Для вибраного прикладу вона залежить від кількості епох навчання $n_{епох}$ та кількості нейронів на прихованому шарі H: $O(N \cdot n_{епох} \cdot H \cdot (a^2 + K))$.

Тоді сукупна часова складність буде такою: (∗ епох ∗ ∗ (2 + K) + K ∗)

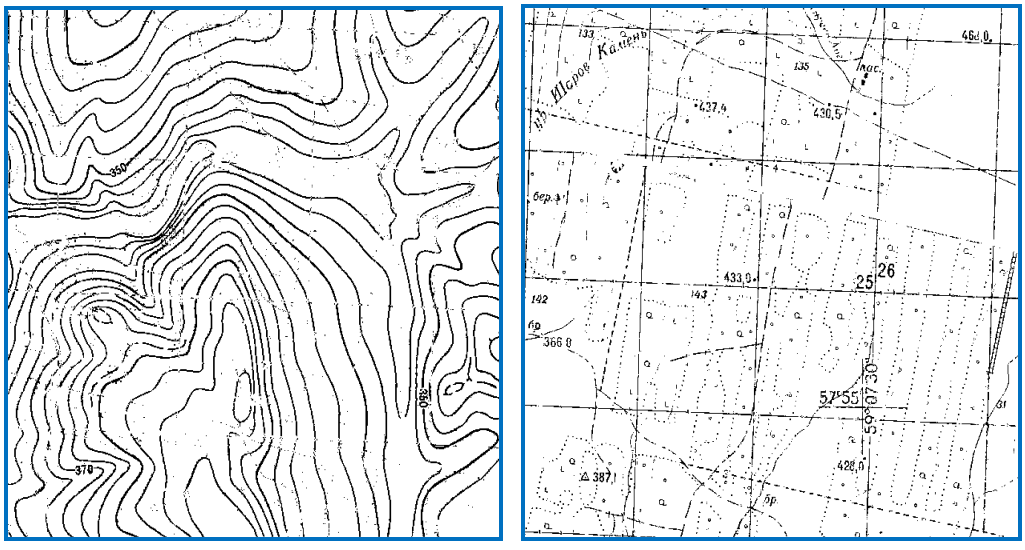
$$O_{\Sigma} = O(N \cdot n_{епох} \cdot H \cdot (a^2 + K) + K \cdot S). \quad (2.6)$$

На рисунку 2.4 показаний алгоритм проведення багатометкової сегментації, який використовує раніше навчену модель (рисунок 2.2). На рисунку 2.5 показаний приклад роботи алгоритму.

Вхідними параметрами є вихідне зображення, розмір ковзного вікна, інтелектуальна модель.

```
function result = segmentation(picture, net, a)
    [nx, ny, ~] = size(picture)
    input = zeros(3 * a * a, nx * ny)
    paddedPicture = padarray(picture, [(a - 1) / 2, (a - 1) / 2])
    for i = 1 : a
    |  for j = 1 : a
    |  |  shiftedPicture = paddedPicture(i : nx + i - 1, j : ny + j - 1, :)
    |  |  reshapedPicture = reshape(shiftedPicture, [3, nx * ny])
    |  |  shift = 3 * (a * i + j - a)
    |  |  input(shift - 2 : shift, :) = reshapedPicture
    |  end
    end
    batch = 1 : floor((nx * ny) / (a * a)) : (nx * ny + 1)
    batch(numel(BatchInd)) = nx * ny
    for i = 1 : numel(batch) - 1
    |  batchInput = input(1 : 3 * a * a, batch(i) : batch(i + 1)) / 255
    |  result(:,batch(i) : batch(i + 1)) = sim(net, batchInput)
    end
    result = reshape(result, [nx, ny, nClasses])
    result = result(result > 0.1)
end
```

Рисунок 2.4 – Модуль багатометкової сегментації



а б

Рисунок 2.5 – Приклад багатометкової сегментації під час обробки ЦТК:

а – напівтонове зображення для «коричневих» об'єктів;

б – напівтонове зображення для «чорних» об'єктів

Вихідним параметром є результат сегментації, виражений у вигляді напівтонових зображень *K*.

Перша частина алгоритму багатометкової сегментації (рядки 2..12) відповідає за отримання з вихідного зображення вхідних даних для роботи інтелектуальної моделі. Це здійснюється з виконанням зміщення, яке лежить в інтервалі [– (a – 1) / 2; (a – 1) / 2] по обидві осі розширеного зображення (*paddedPicture*). Щоб пікселі, що лежать на границі ССЗ, з меншою ймовірністю були віднесені до неправильного класу, то розширення варто проводити значеннями, що знаходяться симетрично границі. Потім дані перетворюються на форму (*reshapedPicture*), аналогічну тій, що використовується у алгоритмі навчання.

У другій частині алгоритму багатометкової сегментації (рядки 13..20) відбувається пакетна обробка вхідних даних, оскільки вони мають розмір $\Theta(S \cdot a^2)$, що може призвести до переповнення.

Часова складність алгоритму дорівнює (для нейромережі прямого поширення аналогічна навчанню, але без n_{enox}):

$$O_{\Sigma 1} = O(S \cdot a^2 \cdot S \cdot H \cdot (a^2 + K)) = O(S \cdot H \cdot (a^2 + K)). \quad (2.7)$$

2.3 Локалізація та визначення типів сегментованих об'єктів

Після виконання попередніх алгоритмів були отримані напівтонові зображення, значення в яких означають міру належності пікселя вихідного зображення до певного класу об'єктів. Необхідно розділити ці зображення на окремі образи, тобто зробити їх локалізацію в просторі растру, підкласифікацію (наприклад, відділення текстових міток одного розміру, несучих одну функцію, від інших) і виявити їх більше абстрактні властивості, наприклад, просторовий тип об'єкта (точковий, лінійний, площинний), кут нахилу (для точкових об’єктів), товщину ліній (для локальних об’єктів) тощо.

Через суттєві відмінності властивостей об'єктів просторовим типом неможливо виконати їх спільну обробку одним алгоритмом. При спробі покращити якість розпізнавання ЛО в ТО вносяться спотворення. Так, наприклад, складність локалізації міток на топографічних картах виникає через високу ступінь перетину об'єктів і низької якості цифрових карт в цілому: точність локалізації об'єктів для розробленого алгоритму складає 85% з необхідністю великої числа операцій, що виконуються оператором.

При цьому стоїть відзначити топологічну зв'язок між ТО і ЛО: часто ТО знаходяться прямо на місці проходження ЛО (як на діаграмах з зображенням кількох функцій з параметром або на картах міських доріг), або поруч (як на кресленнях), і мають однаковий колір (як на цифрових картах). ПлО ж зазвичай виконують роль фонових об'єктів для ТО і ЛО, тому не поєднують в собі воз- можливості просторового перетину і збіги квітів з ними. Тому можна виділити два алгоритма: для локалізації ТО і ЛО і для локалізації ПлО, які розглянемо у наступних підрозділах.

2.3.1 Алгоритм для локалізації точкових та лінійних об'єктів

Існує кілька підходів до відділення ТО і ЛО. Найпростіші спираються на контурний аналіз: якщо зробити більше строгу сегментацію (з бінарної карткою на виході, а не напівтонової), то ЛО можна відрізнити від ТО за допомогою обчислення довжини контуру. Такий підхід можливий при високих значеннях dpi і достатньому відокремленні об'єктів у кольоровому просторі.

Набагато більше гнучкі результати виходять при використання математичної морфології.

Мінусом зазначеного дослідження є те, що підбір морфологічних операцій в нім здійснюється безпосередньо оператором. А так як на ССЗ можуть існувати ТО великої кількості типів, то підбір операцій для підвищення автоматизації можна виконувати за допомогою генетичних

алгоритмів (ГА): можна реалізовувати локалізацію і визначення найпростіших типів об'єктів (квадрати, зірки, диски) на порожньому тлі для розпізнавання відсканованих нотних партій або використовувати для покращення рентгенівських зображень за допомогою дерева операцій.

Для того, щоб сформувати ГА, необхідно визначити кілька основних понять для розв'язуваною завдання.

1. Ген – виконувана операція над поточним зображенням з деякими параметрами. Як операції розглядаються:

- морфологічні: дилатація, ерозія, замикання, розмикання, скелетонізація – з різними структурними елементами та зв'язністю, напівтонові та бінарні;

- порогова: перетворює напівтонове зображення в бінарне через значення порога;

- фільтр по властивості бінарного об'єкта: площа, ексцентриситет, розмір більшої або меншої півосі еквівалентного еліпса – дані операції можуть виконуватися тільки з бінарними об'єктами, тому їм повинна передувати операція порогової обробки.

Оскільки ССЗ представляють собою двовимірне зображення, то при обробці зазвичай реалізується два виду зв'язності: 4-зв'язність (або окіл фон Неймана) та 8-зв'язність (околиця Мура).

4-зв'язність припускає, що піксель має 4 сусідів: по одному з кожної сторони; 8-зв'язність припускає наявність, крім цього, ще і сусідів по діагоналях.

Під еквівалентним еліпсом мається на увазі еліпс, який має ті ж моменти інерції площі щодо осей x та y, як і бінарний об'єкт.

2. Хромосома – упорядкована послідовність операцій – генів, виконуваних над вихідним зображенням послідовно. можна розглядати також більше складну систему, коли вхідними зображеннями будуть бути інші зображення, в том числі також отримані з допомогою морфологічної обробки, а операціями – бінарні (арифметичні або логічні додавання і

множення). Тоді хромосому можна буде уявити в вигляді дерева операцій. Так як ГА самі по собі є трудомісткими в плані обчислювальної складності, то таке рішення може ще сильніше збільшити варіативність при пошуку оптимальної хромосоми, тому таку можливість розглядати не будемо.

Типова виконання ГА містить генотип (у нашому випадку він складається з однієї хромосоми) певної довжини, але для нашого завдання така інтерпретація не підходить, тому будемо припускати варіативність довжини хромосоми.

Враховуючи, що морфологічні операції можуть проводитися як з напівтоновими, так і з бінарними зображеннями, а проведення кількох порогових операцій не має сенсу, то можна укласти, що в нашому випадку хромосома буде мати такий вигляд, що спочатку йдуть тільки морфологічні операції (назвемо її лівою частиною хромосоми), одна порогова (аналог центроміри), потім морфологічні операції та фільтри (права частина).

3. Функція пристосованості – критерій, за яким буде проводитися селекція особин. Для визначення функції пристосованості відзначимо, що є оптимальним результатом виконання операцій по локалізації і визначенню типів сегментів. за напівтоновому зображення повинно бути отримано бінарне, в до тором все об'єкти є розділеними в околиці Мура: ніякі пікселі будь-яких двох об'єктів не мають загальної вершини. Таким чином, в рамках одного класу буде виключено можливість нашарування або гетерархічності .

Візуальні об'єкти на зображенні можна розмітити за допомогою областей, в якості яких можна використовувати мінімальну опуклу оболонку або іншу, більше зручну для розмітки фігуру, наприклад прямокутник. Тоді функцію пристосованості можна висловити в вигляді суми *IoU* від всіх пар об'єкт фігури. Нестача такої функції полягає в тому, що якщо якісь об'єкти будуть втрачені, то максимізація такий суми все одно може бути зроблена, але це буде тільки локальний оптимум.

Тому пропонується спочатку виконувати селекцію за повнотою розпізнавання, яку будемо розуміти, як кількість знайдених об'єктів, що

містять критичну точку вихідних об'єктів, наприклад, центр розміченої фігури. Якщо після зміни хромосоми даний параметр не зменшується або знаходиться на рівні не нижче 95% від максимального, то тоді дана хромосома може розраховуватися по параметром IoU.

Функція селекції за повнотою буде такою:

$$RecallGA(P')=[|P' \wedge P_{crit}| \geq |P_{\pi'} \wedge P_{crit}| \; or \; |P' \wedge Pcrit| \geq 0.95 * |P_{crit}| \;]. \qquad (2.8)$$

де P' − бінарне зображення, отримане в результаті виконання операцій над оригінальним у поточній хромосомі, P_{crit} – бінарне зображення, де одиницями відзначені центри розмічених фігур, $P_{\pi'}$ – бінарне зображення, отримане для предка цієї хромосоми π (для початкової популяції вважатимуться рівної нульовому растру), $[statement]$ – дужки Айверсона, $|P|$ – потужність ненульових пікселів бінарного зображення.

Таким чином, сукупну функцію пристосованості можна сформулювати таким чином:

$$FitnessFunction(P')=RecallGA(P') \cdot \frac{|P' \wedge P_{crit}|}{|P_{crit}|} \cdot \frac{|P' \wedge P_{ideal}|}{|P' \vee P_{ideal}|}. \qquad (2.9)$$

де P_{ideal} – розмічене бінарне зображення.

4. Генетичні оператори – дії, які будуть проводитися з генами для їх модифікації.

У класичних генетичних алгоритмах такими операціями є схрещування та мутація.

Для нашого завдання операція схрещування з довільним перемішуванням генів немає сенсу, оскільки тут гени є незалежними за своєю природою, тобто кожна ген-операція виконується строго після попередньої, і функція пристосованості визначається відразу саме для всієї послідовності. Тому під схрещуванням розуміється таке визначення нащадка-хромосоми, при якому від одного предка береться ліва частина, а від іншого права:

$$Chromosome_{new}=[Chromosome_n(1{:}k_1), Chromosome_m(k_2{:}end)], \quad (2.10)$$

де $Chromosome_n(1{:}k_1)$ – перші k_1 генів, що взяті із n-ї хромосоми-батька, $Chromosome_m(k_2{:}end)$ – гени, що йдуть після k_2 із m-ї хромосоми-батька, $[x,y]$ – оператор *append* для двох хромосом.

Для обчислення функції пристосованості від хромосоми після схрещування беремо предка з максимальним значенням цієї функції.

Як мутацію розглянемо 3 варіанти:

- вставка гена-операції в середину (може бути також реалізована вставка на початок або в кінець) хромосоми, при цьому вибирається тільки один з батьків:

$$Chromosome_{new}=[Chromosome_n(1{:}k), Gene, Chromosome_n(k{+}1{:}end)]; \quad (2.11)$$

- модифікація випадкового гена, наприклад, зміна числових параметрів структурного елементу морфологічної операції (СЕМО), рівня порога чи параметра фільтра;

- видалення випадкового гена, якщо він не є граничною операцією.

Операція мутації зі зміною СЕМО гена не розглядається, оскільки, по-перше, така зміна швидше зменшить значення функції пристосованості (наприклад, заміна дилатації на ерозію призведе до зникнення областей, що зменшить перетин із зображенням критичних точок), а, по-друге, може бути замінена послідовним виконанням двох мутацій: видалення та вставлення нового гена.

ГА для отримання послідовності виконуваних операцій можна записати у вигляді, поданому в алгоритмі отримання послідовності операцій для локалізації, наведеного на рисунку 2.6.

Вхідними параметрами для алгоритму є такі: напівтонове зображення після сегментації, бінарні зображення ідеальної класифікації окремих сегментів та зображення з розміченими критичними точками.

Вихідним параметром є хромосома з операціями, яку можна використовувати для отримання результуючого зображення.

```
1:    function bestChromosome = geneticAlgorithm(picture, pictureIdeal, pictureCrit)
2:       parent = zeros(populationSize, 1)
3:       fitFunc = zeros(populationSize, 1)
4:       for i = 1 : populationSize
5:       |  chromosome(i, :) = struct('name', "threshold",
                 'properties', struct('name', "level", 'param', rand()))
6:       end
7:       fitFunc = fitnessFunction(chromosome, 1:populationSize, parent,
                                picture, pictureIdeal, pictureCrit)
8:       end
9:       [~, sortIdx] = sort(fitFunc, 'descend')
10:      chromosome = chromosome(sortIdx, :)
11:      fitFunc = fitFunc(sortIdx)
12:      iteration = 0
13:      while iteration < maxIteration
14:      |  newChromosome = chromosome(1:ceil(populationSize * elitePart), :)
15:      |  for i = ceil(populationSize * elitePart) + 1 : populationSize
16:      |  |  parent1 = ceil(rand() * populationSize * elitePart)
17:      |  |  if rand() < probCrossover
18:      |  |  |  parent2 = ceil(rand() * populationSize * elitePart)
19:      |  |  |  newChromosome(i, :) = crossover(chromosome(parent1, :),
                                     chromosome(parent2, :))
20:      |  |  |  parent(i) = (fitFunc(parent1) > fitFunc(parent2)) * ...
                           (parent1 - parent2) + parent2
21:      |  |  else
22:      |  |  |  newChromosome(i, :) = mutation(chromosome(parent1, :))
23:      |  |  |  parent(i) = parent1
24:      |  |  end
25:      |  end
26:      |  chromosome = newChromosome
27:      |  fitFunc = fitnessFunction(chromosome, 1:populationSize, parent,
                                   picture, pictureIdeal, pictureCrit)
28:      |  [~, sortIdx] = sort(fitFunc, 'descend')
29:      |  chromosome = chromosome(sortIdx, :)
30:      |  parent = parent(sortIdx)
31:      |  fitFunc = fitFunc(sortIdx)
32:      |  if fitFunc(1) > acceptableScore
33:      |  |  break
34:      |  end
35:      |  iteration = iteration + 1
36:      end
37:      bestChromosome = chromosome(1, :)
38:   end
```

Рисунок 2.6 – ГА отримання послідовності виконуваних операцій

Використовувані для алгоритму функції (*fitnessFunction*, *crossover mutation*) і функція, яка будує по хромосомі результуюче зображення, та яку можна використовувати в надалі для отримання оптимального результату, не наведено через великого кількості команд, полягають в різної обробці різних операцій ГА.

На початку алгоритму (рядки 2..11) виконується ініціалізація початкової популяції. Так як порогова обробка – це єдина обов'язкова операція, тому що напівтонове зображення повинно бути перетворено в бінарне для обчислення функції пристосованості, то спочатку хромосоми всіх особин складаються із неї, а пороговий рівень обчислюється випадково.

Далі (рядки 12..37) відбувається ітеративне виконання ГА. Використовується елітарна селекція (рядки 14.. 25), для якої з всією популяції вибирається частина elitePart , яка залишається на наступну ітерацію і тільки від якої про- роздягаються нащадки. З ймовірністю probCrossover у пари батьків створюється новий нащадок, інакше у першого батька відбувається народження нового нащадка з мутацією.

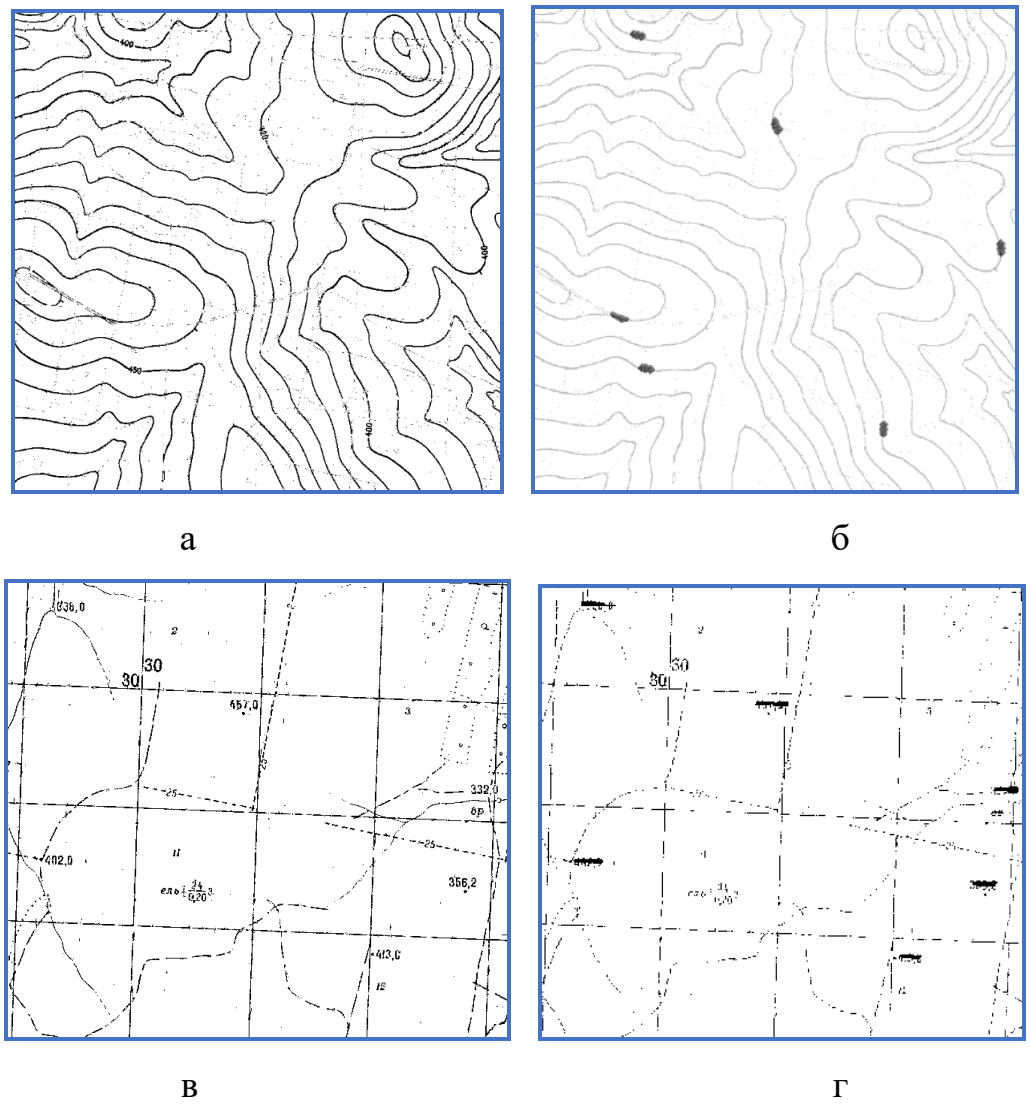
Далі (рядки 26..37) відбувається обчислення функції пристосованості, хромосоми сортуються в спадаючому порядку по ній для майбутнього формування новою еліти особин. Якщо особина з найкращим показником функції пристосованості подолала необхідний рівень *aceptableScore*, то ГА зупиняється . Якщо рівень не досягнуто, то ітерації продовжуються, поки не буде досягнуто максимальне їх кількість maxIteration.

Результат застосування оптимальної хромосоми – бінарне зображення (маска локалізації), містить в собі встановленими пікселі, які належать областям, відповідним деяким об'єктам. Приклад такого застосування наведений на рисунку 2.7. Таким чином, даний алгоритм дозволяє локалізувати місця, в яких потім можна реалізувати алгоритми розпізнавання образів.

Визначити часову складність алгоритму отримання результуючих бінарних зображень заздалегідь неможливо, єдине, що можна дати загальну оцінку O($S \cdot (N_{genes})^2$), де N_{genes} – кількість генів оптимальної хромосоми. Час виконання окремих операцій має складну залежність, наприклад, для морфологічних операцій розмір структурних елементів при прямому підході до обчислення впливає на час квадратично, тобто O($S \cdot M^2$), де M –

характеристичний розмір СЕМО, але при розкладанні у послідовність кількох операцій починає впливати лінійно, тобто O($S \cdot M$).

Час роботи операцій фільтрації залежить скоріше від кількості об'єктів після порогової обробки, тому що їх зберігання виконується за допомогою окремих властивостей, тому можна сказати, що будь-яка така операція виконуються за $O(S)$, так як можна сказати, що кількість об'єктів на зображенні обмежена його площею.



а б в г

Рисунок 2.7 – Приклад роботи алгоритму локалізації точкових об'єктів: а – вихідне зображення для "коричневих" об'єктів; б – маска локалізації позначок «поверх» вихідного порівняння; в – вихідне зображення для "чорних" об'єктів; г – маска локалізації позначок «поверх» вихідного порівняння

Час для роботи генетичного алгоритму можна оцінити як $O(S*N_{genes}*P*N_{iter})$, де P – розмір популяції, N_{iter} – кількість ітерацій.

2.3.2 Алгоритм для класифікації площинних об'єктів

Однією із ознак ССЗ є нерегулярність, тому розпізнавання великих ПлО стає проблемою, так як вони можуть бути представлені великою різноманітністю форм та текстур.

Існує підхід до розпізнавання площинних об'єктів як точкових при достатньо великих по розміром навчальних зображення, використовуючи згорткові нейронні мережі (CNN). Так, CNN виробляє розпізнавання текстур об'єктів з аерофотографій у вигляді бінарного кодування зображення; результати при цьому значно нижче для розпізнавання об'єктів, що мають швидше площинний, ніж об’єкти із точковою або лінійною топологією, наприклад, 83% для парків, 67% для центру міста, але 97% для віадуків, 95% для річок, 99% для паркування.

CNN може грати роль фільтра при обробці старих топографічних карт: зі всього зображення виділяються саме зображення будинків, що, допустимо, актуально для історичних досліджень, так як дозволяє автоматизовано вивчати зміни міських просторів. *IoU* при цьому в залежності від обраного методу та розміру ковзного вікна (від 64×64 до 224×224) варіюється від 56 до 92%.

Головним недоліком використання CNN є необхідність тривалого навчання і швидкість роботи вже навченої моделі. Видно, що результати розпізнавання Пло далекі від аналогічних результатів при розпізнавання ТО. Тому можна зробити висновок про використання інших підходів, особливо, коли це стосується зображень II та III класу.

Одним з підходів є використання гістограм: при ньому по ковзаючому вікну обчислюється гістограма (наприклад, яскравості для напівтонового зображення) і далі робляться висновки щодо знаходження об'єкту.

Недоліком даного підходу є те, що такі гістограми по суті є інтегральними характеристиками, і в них можуть проявитися власні особливості, якщо об'єкти розташовані під різними кутами. Наприклад, для розпізнавання мотоциклів і велосипедів на міських фотографіях треба будувати низку окремих дескрипторів, залежних від положення об’єктів: горизонтального, вертикального або діагонального.

Хоча при відображенні ПлО на ССЗ можуть з’явитися значущі шуми, але в цілому слід відзначити статистичну сталість в варіативності відхилення кольору і її частоті навіть в невеликих областях, що дозволяє значно обмежити розмір ковзаючих вікон на відмінність від, наприклад, CNN, для яких розмір вікон для точкової класифікації часто перевищує сотні пікселів..

Для вирішення задачі визначення ПлО на ССЗ потрібно очистити її від об'єктів, знайдених на етапі обробки ТО і ЛО (назвемо безліч їх пікселів Т&Л), які зображуються поверх ПлО , що є для них тілом. Для таких пікселів неможливо в принципі визначити, що знаходилося під образами переднього візуального плану, але можна припустити деяку континуальність і протяжність об'єктів заднього фону. Дана властивість можна назвати принципом локальності, який можна виразити в існуванні такої функції, що визначає клас пікселя за класами пікселів його сусідів і задовольняє наступній умові для майже кожного пікселя (під майже кожним пікселем тут розуміється, що ця умова може виконуватися для O(1) пікселів кожного площинного об'єкта):

$$class(p) = f_{local}(\{class(p') |\, d(p,p') \leq d_{max}\}), \qquad (2.12)$$

де $class(p)$ – клас фонового зображення поточного пікселя p, що визначається через свої координати растру, p' – пікселі, для яких виконується умова сусідства, d – метрика, за якою розраховується відстань між двома пікселями, d_{max} – максимальне значення даної метрики.

Як найпростіший приклад f_{local} можна скористатися функцією моди класів пікселів в околі, відстань Чебишева якої до центру не більше певного

значення. Тоді порушення принципу локальності відбуватиметься, наприклад, коли розміри ПЛО менше 2 d_{max}.

Так як клас ПлО фонового шару визначає його візуальне відображення, то для усунення розривів, утворених видаленням $P_{\text{Т\&Л}}$, скористаємося фільтром Гауса над растром, при цьому пікселі, які не належать $P_{\text{Т\&Л}}$ зберігають свій колір, стільки разів, поки зміни не перестануть відбуватися. Оскільки ЛО і ТО мають невеликі розміри, то досить кінцевої кількості даних ітерацій.

Площинні об'єкти характеризуються швидше своїми кольорами і найпростішою текстурою, то вікно навколо пікселя, що цікавить, можна охарактеризувати набором ідеальних кольорів у потрібній пропорції. Кожному класу ПлО відповідає 3 гістограми (по кожній з колірних координат), що і є ознаковим описом кольору. Даний підхід дозволяє обробляти як одноколірні об'єкти, а й багатобарвні, наприклад, зображення боліт на ЦТК. Тим не менш, даний підхід не дозволяє враховувати при розпізнаванні відносне розташування пікселів, наприклад, гістограма не буде видно, під яким кутом виконана штрихування.

Поточкову класифікацію можна проводити з використанням різних моделей. При цьому може статися ситуація, що піксель не сприйме жодного з існуючих класів. Друга ситуація – класифікація з досить великою вихідною зашумленностью, коли поруч пікселі, що знаходяться, будуть віднесені до різних класів, хоча семантично ставляться до одного об'єкта.

Пропущені пікселі необхідно обробити, використовуючи ітеративний про процес нарощування вже класифікованих пікселів, знаходячи моду серед околиці пікселів. Якщо частота зустрічі моди більше критичного значення, то пікс ялина класифікуємо, якщо ні, то пропускаємо для обробки в наступною ітерації. значення зменшуємо з збільшенням номери ітерації.

Розроблений алгоритм класифікації площинних об’єктів представлений на рисунку 2.8, а приклад його реалізації представлений на рисунку 2.9.

На рисунку 2.9, а зображено вихідне кольорове зображення, а на рисунку 2.9, б бачимо зображення з трьома класами площинних об'єктів: «ліс», «низькорослa рослинність», «рідколісся».

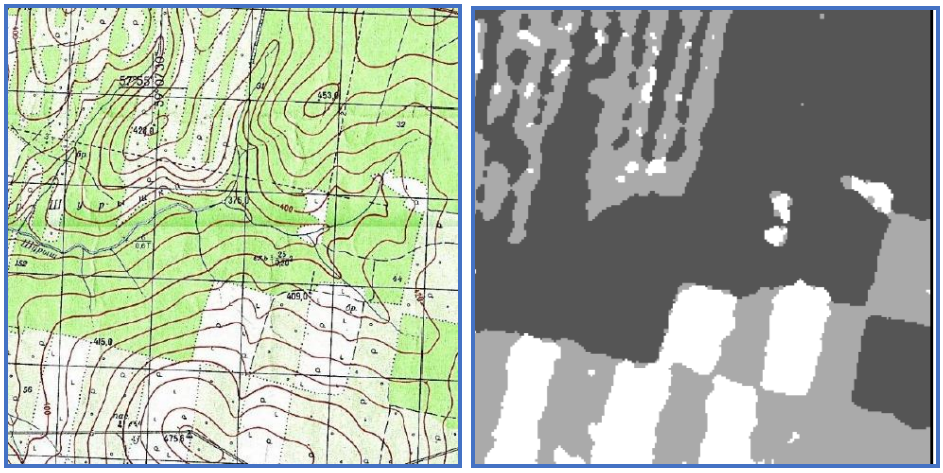
Вхідними параметрами алгоритму є вхідне кольорове зображення, бінарне зображення Т&Л, гістограми класів ПлО з ваговими коефіцієнтами по стовпцях і розмір ковзного вікна. Вихідним параметром є зображення, в якому кожному пікселю поставлений у відповідність один клас ПлО .

```
1:    Function result = classAreal(picture, picturePAndL, model, a)
2:       picture = backGroundReconstruction(picture, picturePAndL)
3:       nx = size(picture, 1)
4:       ny = size(picture, 2)
5:       result = zeros(nx, ny)
6:       x = reshape(repmat(1 : nx, [1, ny]), [1, nx * ny])
7:       y = reshape(repmat(1 : ny, [nx, 1]), [1, nx * ny])
8:       points = [x; y]
9:       nbhd = getNeighborhood(picture, points, a)
10:      edges = 0 : binwidth : 256
11:      rhist = histc(nbhd(1 : 3 : 3 * a * a, :), edges) / (a * a)
12:      ghist = histc(nbhd(2 : 3 : 3 * a * a, :), edges) / (a * a)
13:      bhist = histc(nbhd(3 : 3 : 3 * a * a, :), edges) / (a * a)
14:      result = predict(model, [rhist; ghist; bhist])
15:      iteration = 1
16:      while numel(find(result == 0)) > 0
17:      |  points = find(result == 0)
18:      |  nbhd = getNeighborhood(result, points, a)
19:      |  histogram = histc(nbhd, 0 : size(hClass, 3))
20:      |  histogram = histogram(2 : end, :)
21:      |  for i = 1 : size(points, 2)
22:      |  |  [counter, class] = max(mistogram(:, i))
23:      |  |  if counter > (1 - iteration / 10) * a * a
24:      |  |  |  result(points(1, i), points(2, i)) = class
25:      |  |  end
26:      |  end
27:      |  iteration = iteration + 1
28:      end
29:   end
```

Рисунок 2.8 – Алгоритм класифікації площинних об’єктів

У першій частині алгоритму (рядки 2..14) відбувається первинна класифікація площових об'єктів , використовуючи колір пікселів. Часова складність цієї частини залежить від інтелектуальної моделі.

У другій частині алгоритму (рядки 15... 28) відбуваються подібні операції, але як «зображення» використовується знайдене на попередньому етапі зображення з розміченими класами.



а б

Рисунок 2.9 – Приклад класифікації площинних об'єктів

Далі процес запускається ітеративно, доки всі точки не будуть класифіковані модальним значенням класів сусідніх пікселів. Щоразу у процесі беруть участь лише ті пікселі, які ще не були класифіковані, тому можна сказати, що часова складність цього етапу обмежена. Так як остаточно класифікуються пікселі, що знаходяться на граничному шарі нерозпізнаних областей, то на кожній ітерації кількість розглянутих пікселів скорочується, тому число ітерацій можна оцінити верхньою границею.

2.4 Розпізнавання образів та їх угруповання

Даний етап обробки ССЗ характеризується більш великим впливом специфічності, чим попередні. Якщо попередні розроблені алгоритми можна вважати альтернативами для існуючих алгоритмів обробки зображень, то вже тут характеристики кожного окремого типу ССЗ можуть надавати достатнього впливу, отже з'являється необхідність для розробки окремих алгоритмів або їх варіацій.

Тим не менше передбачається, що можливо створення алгоритмів, які будуть давати гарні результати щодо якості для широкого класу ССЗ, якщо врахувати більше число їх проявів.

2.4.1 Алгоритми для точкових об'єктів

Більше формальні і жорсткі підходи до розпізнавання образів на ССЗ ха рактерни для більше простих образів. Можна використовувати узагальнене перетворення Хафа; але мінусом даного дослідження є те, що навіть для дуже маленьких зображень (порядку 10 Кб) час обробки складає порядку декількох секунд. Також можна розраховувати дескриптори функції автокореляції, входом якої є масив змін координат кордонів символ; навіть для документів з раз- рішенням 300 dpi такий підхід забезпечує точність розпізнавання від 80% для окремих типів символів. У деяких методах розпізнавання символів зводиться до визначення кола, що оптимально покриває символ; хоча даний алгоритм годиться тільки для розпізнавання об'єктів, що репрезентують собою невеликі опуклі символи.

Більше інтелектуальні методи, що використовують готові OCR рішення (наприклад, Google Tesseract , в т. год. вбудований в Комп'ютер Vision Toolbox пакету MatLab) не завжди показують найкращий результат. Сам по собі Tesseract не є досить гнучким інструментом для роботи з ССЗ, оскільки вимагає, щоб символи були щонайменше 20 пікселів у висоту (що можливо тільки для документів починаючи з 300 dpi), не допускаються навіть незначні повороти символу, зрушення або розмиття. Його використання без додаткової попередньої обробки призводить до якості порядку 77-80% для ССЗ, що мають або перетин ТО з іншими об'єктами, або деякі варіації шрифтів.

Використання згорткових нейронних мереж і глибокого навчання теж не є оптимальним рішенням. Використання передбачених нейромереж, наприклад, ResNet-50 або VGGNet для локалізації і Tesseract для розпізнавання також дають спірні результати. У першим випадку це 65-70% для ТО, які можуть перетинатися з іншими об'єктами, у другому випадку 87-94% для ТО, які не перетинаються з іншими об'єктами (через особливості стандартів USGS US Topo для ССЗ зображень).

Враховуючи, що ШНМ прямого поширення з одним прихованим шаром і інші одноетапні інтелектуальні моделі не можуть показати достатніх результатів татів розпізнавання на картах, а багатошарові (й згорткові) нейромережі показують значне зменшення швидкодії, то треба орієнтуватися на двокаскадні моделі. Центри розпізнаваних образів можна отримати із знайдених раніше бінарних зображень, на яких є області локалізації об'єктів. Дані області не відповідають безпосередньо об'єктам: з одного сторони, вони можуть не включати в себе цифр повністю, а з іншого боку можуть перетинатися з декількома цифрами (і іншими об'єктами) одночасно. Морфологічна обробка була здійснено так, що більшість центрів цифр потрапило в ці сегменти.

Використавши зображення ковзаючих вікон з центрами у множині точок цих областей, навчимо перший каскад інтелектуальної моделі. Розміри ковзаючих вікон в даному випадку відрізняються від етапу сегментації, так як розпізнавані об'єкти (наприклад, цифри) можуть бути витягнутими об'єктами (для цифр відношення висоти до ширини приблизно дорівнює 2).

Виходом інтелектуальної моделі є міра впевненості в тому, що в поточному ковзному вікні знаходиться об'єкт одного з K класів. Таким чином, здійснена операція, аналогічна багатометковій сегментації, проведеній раніше, і одне бінарне зображення було перетворено в K напівтонових зображень. Враховуючи, що на ССЗ може бути використано велике число класів, то слід реалізувати порогове відсікання і зберігати матриці в розрядженому форматі.

Далі необхідно розбити отримані області, щоб вони відповідали різним розрядам числа. Здійснити це можна, провівши додаткову операцію по фільтрації з використанням функції заповненості за типом:

$$fullness(x_c, y_c, k) = 1 - 4 \cdot \left(\frac{\sum_{(x,y) \in S(x_c, y_c)} measure(x, y)}{average\ fullness(k)} \right)^2, \qquad (2.13)$$

де (x_c,y_c) – центр ковзного вікна, k – клас об'єкта, $S(x_c,y_c)$– множина точок ковзного вікна з центром у зазначеній точці, $measure(x,y)$ – значення міри належності до узагальненого класу об'єкта, $average\ fullness(k)$ – середнє значення заповненості для даного класу об'єктів. В якості значень растру $measure(x,y)$ можна взяти із напівтонового зображення, отриманого після сегментації за узагальненим класом. Підсумкове значення $fullness(x_c, y_c, k)$ можна також модифікувати, використовуючи інтелектуальну модель. Даний підхід використовується для мінімізації хибного спрацьовування на сторонні об'єкти, які точно не відносяться до шуканому класу. Наприклад, крім ШНМ для розпізнавання окремих цифр потрібна також і ШНМ для отримання заходи впевненості в тому, що в даному ковзному вікні цифра взагалі є. Хоча така інформація буде мати значну кореляцію, але вона дозволить відсікти цифроподібні об'єкти.

Значення функції заповненості повинно бути близько до 1 для центру цифри і наближатися до 0 в точках між цифрами. З допомогою пороговий обробки можна здійснити поділ і виділити сегменти, які будуть точно відповідати певної цифри, що зменшить вплив розташування об'єктів поряд для покращення якості роботи другого каскада інтелектуальної моделі. Таке поділ дозволить розділити цифри лише на візуальному плані: сусідні пікселі можуть бути розпізнані як центри різних цифр. Назвемо такі сегменти, які складаються з пікселів, віднесених до різним класам об'єктів, суперсегментами. Формально суперсегмент можна визначити як безперервне безліч пікселів на індексованому зображення, де значення кольори пікселя визначається індексом класу. Наприклад, суперсегменти, відповідальні цифрі 3, містять пікселі, сприймаються, як центри двійок, трійок та п'ятірок. Це відбувається тому, що ділянки деяких цифр, особливо з обліком спотворень, схожі на ділянки інших цифр (наприклад, нижня частина цифри 3 схожа на верхню частина цифри 2 і на нижню частина цифри 5), тому методи розпізнавання могли здійснити помилку в класифікації. Дана помилка

виникає в будь-кому випадку, особливо на ССЗ, тому існування до додаткової обробки є необхідним. Для того щоб збільшити точність, необхідно вибрати правильне значення з безлічі значень суперсегментів. Для вибору будемо використовувати заходи впевненості в тому, що піксель є центром цифри, отримані на попередній щем етапі, і відстань від пікселя до центру суперсегменту . Усього на вході другого каскада буде десять параметрів, відповідні обчисленим заходам, на виході буде ймовірність того, що даний суперсегмент відповідає однієї з 10 цифр. При цьому відбувається згортка суперсегменту, що складається з кількох точок з безліччю значень в одну точку з єдиним значенням. Зазначимо, що другий рим каскадом може бути і не інтелектуальна модель, а статистична, наприклад, функція моди від значень номерів класів в суперсегменті. Для посилення впливу локальності вхід другого каскада відкоригуємо з використанням відстані пікселя до центру суперсегменту :

$$input_2(i,n) = \cdot \sum_{(x,y)\in S(i,n)} \frac{output_1(x,y)}{1 + k_R \cdot D\big((x,y),(x_{ci},y_{ci})\big)}, \tag{2.14}$$

де $S(i,\ n)$ – множина пікселів з координатами (x, y) i-го суперсегменту, відповідних класу n, $output_1(x,y)$ – вихідне значення міри класу n для точки (x, y), (x_{ci},y_{ci})– координати центру i-го суперсегмента, $D\big((x,y),(x_{ci},y_{ci})\big)$ – евклідова відстань між поточною точкою та центром, k_R – коефіцієнт урахування відстані. Після обчислення вхідні параметри нормалізуються.

Стоїть відзначити відмінність двокаскадного алгоритму від ШНМ з двома шарами. У розробленому алгоритму на вхід першого каскада йдуть зображення ковзаючих вікон зі всього простору растру, а на другий каскад вже тільки дані, одержувані від суперсегментів без ковзаючих вікон. Обидва каскада навчаються з використанням однакових вихідних даних. Загальний алгоритм розпізнавання цифр представлений на рисунку 2.10, а приклад обробки наведено на рисунку 2.11.

```
1:   Function result = recognition(segPicture, locPicture, a, avrFullness)
2:      ax = a(1)
3:      ay = a(2)
4:      label = bwlabel(locPicture)
5:      stats = regionprops(label, 'Orientation')
6:      phi = [stats.Orientation] * pi() / 180
7:      [x, y] = find(locPicture)
8:      nPoints = numel(x)
9:      points = [x, y]'
10:     input2 = zeros(ax * ay, nPoints)
11:     measure = zeros(ax * ay, nPoints)
12:     nClasses = numel(avrFullness)
13:     output = zeros(nClasses, nPoints)
14:     for i = 1 : nPoints
15:     |  input(:, i) = getNeighborhood(segPicture, points(:, i),
                                          [ax, ay], phi(label(x(i), y(i))))
16:     |  measure(:, i) = sum(input(:, i))
17:     end
18:     fullness = 1 - 4 * (measure ./ avrFullness - 1) .^ 2
19:     filter(1 : nPoints) = sim(filternet, input)
20:     output(:, 1 : nPoints) = sim(firstnet, input)
21:     multiPicture = zeros(size(locPicture, 1), size(locPicture, 2), nClasses)
22:     for i = 1 : nPoints
23:     |  multiPicture(x(i), y(i), :) = output(:, i) .* ...
        |                                 filter(i) .* fullness(:, i)
24:     end
25:     for i = 1 : nClasses
26:     |  multiPicture(:, :, i) = bwareaopen(multiPicture(:, :, i) > 0.5, 2))
                                  .* multiPicture(:, :, i)
27:     end
28:     [value, indexedPicture] = max(multiPicture, [], 3)
29:     indexedPicture = indexedPicture.* (value > 0)
30:     indexedPicture = filter(indexedPicture)
31:     [sSegment, nSS] = bwlabel(indexedPicture > 0, 8)
32:     stats = regionprops(sSegment > 0, 'Centroid',
                            'PixelIdxList', 'PixelList')
33:     cntr = [stats.Centroid]
34:     xc = round(cntr(1 : 2 : end))
35:     yc = round(cntr(2 : 2 : end))
36:     input2 = zeros(nClasses, nSS)
37:     for i = 1 : nSS
38:     |  idx = stats(i).PixelIdxList
39:     |  pt = stats(i).PixelList
40:     |  for j = 1 : numel(indexes)
41:     |  |  input2(indexedPicture(idx(j)), i) =
42:     |  |  input2(indexedPicture(idx(j)), i) +
43:     |  |  value(idx(j))/(1+kR*((xc(i)-pt(j,1))^2 +(yc(i)-pt(j,2))^2)^0.5)
44:     |  end
45:     |  input2(:, i) = input2(:, i) / sum(input2(:, i))
46:     end
47:     output2 = sim(secondnet, input2)
48:     [~, output2] = max(output2, [], 1)
49:     result = struct('x', num2cell(xc), 'y', num2cell(yc),
                        'phi', num2cell(phi(L(xc, yc))), 'v', num2cell(output2))
50:  end
```

Рисунок 2.10 – Алгоритм двокаскадного розпізнавання точкових образів

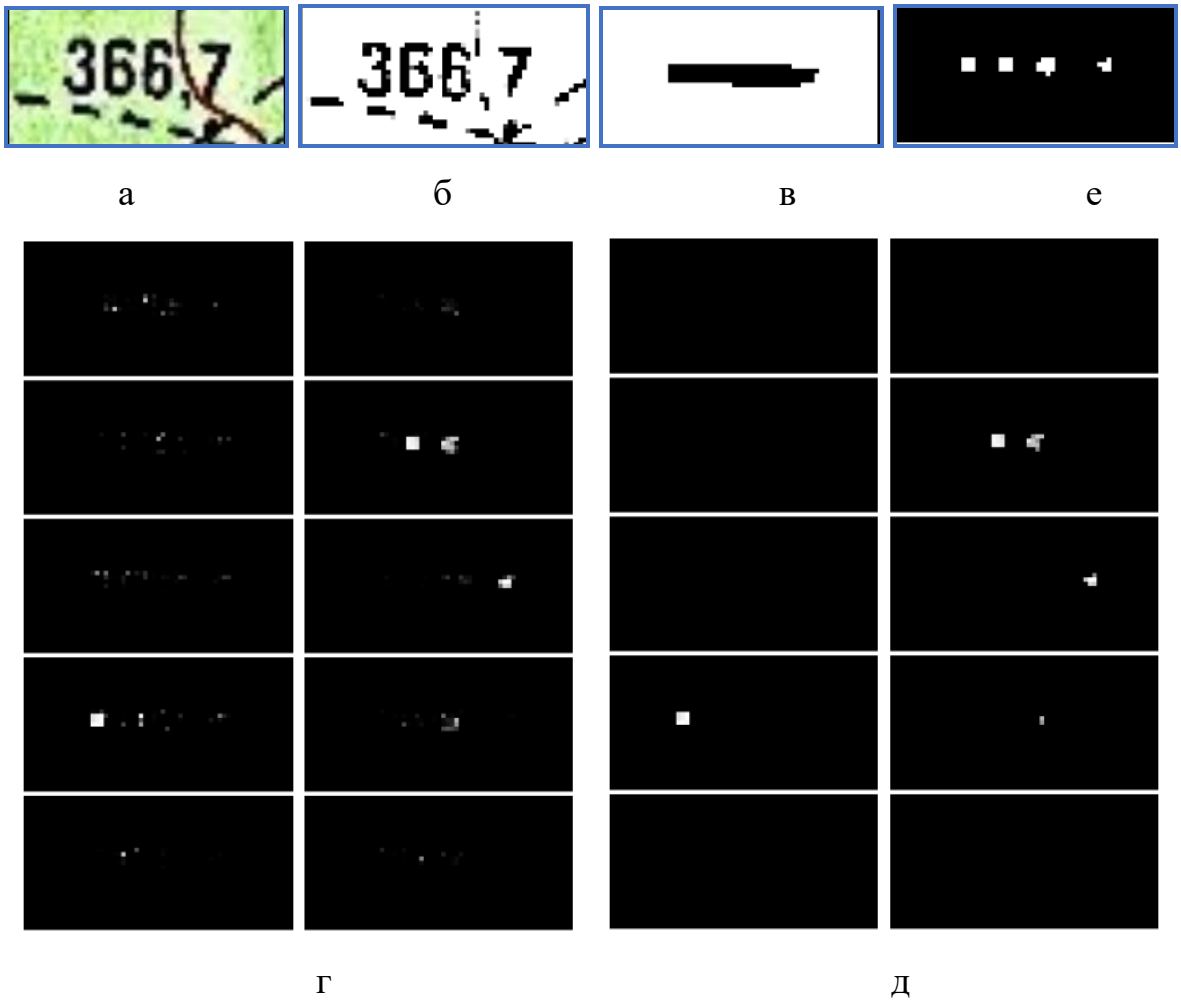


Рисунок 2.11 – Приклад розпізнавання точкових об'єктів:
б – зображення після багатометкової сегментації для класу "число";
а – вихідне зображення; в – бінарне зображення локалізації образу;
г – вихід першого каскаду інтелектуальної моделі (зліва цифри 0-4, справа: 5-9); д – вихід першого каскаду інтелектуальної моделі з урахуванням заповненості; е – індексоване зображення суперсегментів

Вхідними параметрами алгоритму є напівтонове зображення, отримане після етапу сегментації, бінарне зображення, отримане після етапу локалізації, розміри ковзного вікна, середні значення заповненості символів в класі, інтелектуальні моделі обробки даних.

Вихідним параметром є безліч розпізнаних образів, заданих четвіркою значень, де перші три значення відповідають за положення в просторі растру і кут повороту ТО, а останнє значення є ідентифікатором класу.

У рядках 7..24 відбувається обробка та розрахунок першим каскадом. У рядках 25..30 відбувається формування індексованого зображення на основі сформованих напівтонових зображень багатометкової сегментації і фільтрація для розділення образів. У рядках 31..49 відбувається обробка суперсегментів другим каскадом.

Після розпізнавання окремих розрядів їх потрібно згрупувати. Враховуючи, що може статися помилкове спрацьовування на класи інших ТО, необхідно провести угруповання, яке відфільтрує непотрібні групи об'єктів. Таку фільтрацію можна зробити за допомогою кластерного аналізу, дослідивши кількість цифр у кластері, їх взаємне розташування та властивості. Відмінність від кластерного аналізу у класичному розумінні у тому, що з угрупованні символів у слова чи цифр у числа стає важливий порядок.

У загальному випадку угруповання виконується агломеративними і дивізійними методами, при яких образи групуються з обліком ієрархії і специфіки формування груп ТО. У нашому випадку розглянемо рішення завдання угруповання послідовно розташованих образів без більше складною внутрішньої структури всередині групи.

Спочатку здійснимо первинне угруповання з допомогою кластеризації (алгоритм первинної кластеризації наведений на рисунку 2.12).

Будемо називати вісь, вздовж якої розташовуються цифри, великою, а перпендикулярну їй – малою. Основу рішення завдання послідовною угруповання можна потім використовувати при рішенні більше складною завдання, змінюючи місцями велику і малу вісь (якщо ТО в групі розташовуються по двовимірний сітці) або використовуючи додаткові осі (якщо ТО мають більше складне розташування у групі, наприклад, радіальне навколо спільного центру або вздовж кривої).

```
1:    Function group = primaryGrouping(digit, nBucket)
2:       nDigits = size(digit, 2)
3:       maxX = max(digit.x)
4:       bucket = cell(nBucket, 1)
5:       nGroups = 0
6:       for i = 1 : nDigits
7:       |  flag = false
8:       |  k = ceil(Digit(i).x / maxX * nBucket)
9:       |  k = (k==1)*2 + (k > 1 && k < nBucket)*k + (k==nBucket)*(nBucket-1)
10:      |  for j = [bucket{k - 1 : k + 1}]
11:      |  |  ro = ((digit(i).x-group(j).x)^2 + (digit(i).y-group(j).y)^2)^0.5
12:      |  |  psi = atan2(digit(i).y-group(j).y, digit(i).x-group(j).x)
13:      |  |  if abs(digit(i).phi - group(i).phi) < pi() / 4 &&
14:      |  |  |  (abs(ro * cos(psi - phi)) <= 20) &&
15:      |  |  |  (abs(ro * sin(psi - phi)) <= 5) && ~flag
16:      |  |  |  group(j).childs = group(j).childs + 1
17:      |  |  |  group(j).ids(group(j).childs) = i
18:      |  |  |  group(j).v(group(j).childs) = digit(i).v
19:      |  |  |  group(j).x = mean(digit(group(j).ids).x)
20:      |  |  |  group(j).y = mean(digit(group(j).ids).y)
21:      |  |  |  flag = true
22:      |  |  |  break
23:      |  |  end
24:      |  end
25:      |  if ~flag
26:      |  |  nGroups = nGroups + 1
27:      |  |  group(nGroups) = struct('childs', 1, 'x', digit(i).x,
                                'y', digit(i).y, 'phi', digit(i).phi,
                                'ids', [i], 'v', [digit(i).v])
28:      |  |  k = ceil(digit(i).x / maxX * nBucket)
29:      |  |  bucket{k} = [bucket{k}, nGroups]
30:      |  end
31:      end
32:   end
```

Рисунок 2.12 – Алгоритм первинної кластеризації

Розглядаючи деяку поточну цифру, будемо дивитися, з яким вже сформованим кластером виконуються умови близькості вздовж осей. При цьому вздовж великий осі можна припускати більші відхилення (порядку 30-40 пік селів), а вздовж малої осі невеликі (2-3 пікселя). Дані значення залежать від роздільної здатності зображення (dpi), тому при обробці ССЗ даний параметр необхідно вказувати. Якщо ні одного кластера з виконанням умов не знайдено, то це означає, що ця цифра належить до нового кластеру.

Для розв'язання задачі кластеризації в цілому необхідно порівняти всі об'єкти попарно, тому тимчасова складність таких алгоритмів повинна відповідати $O(n^2)$, де n – число об'єктів. В алгоритмі первинної кластеризації представлений випадок, коли ТО на ССЗ зображено рівномірно, що зустрічається в більшості випадків. Тоді для прискорення можна розділити зображення на декілька горизонтальних смуг (геометричне хешування), і якщо зберігати інформацію про вміст всіх смуг, часову складність можна скоротити. Якщо здійснити подібну операцію і для вертикальних смуг, і

шукати необхідні кластери лише у перетині потрібних «осередках» сітки, то часова складність залежить від способу виконання операції перетину.

Після первинної кластеризації можна виконати ряд операцій фільтрації, застосовність яких залежить від специфіки ССЗ. Розглянемо кілька різних варіантів, які можуть траплятися.

При наявності додаткових обмежень на кількість ТО в кластері можна також зробити фільтрацію невеликих кластерів, перебувають менше, чим з мінімального числа об'єктів, так як такі кластери, швидше всього, є наборами з хибно певних об'єктів. Наприклад, при обробці ЦТК не може існувати підписи горизонталі, що складається з однієї цифри, а при обробці карт гірських районів мінімальне кількість можна покласти рівним трьом.

Так як деякі числа можуть не об'єднатися в один кластер, але потрібно провести додаткову операцію агломерації кластерів з допомогою схожою операції, яка проводилася для об'єднання цифр в кластери, але для центрів кластерів, а не цифр. Після цієї операції можна виконати фільтрацію занадто великих кластерів, так як вони представляють собою різні послідовні утворення, які не відносяться до розв'язуваної задачі.

Здебільшого «хибні» ТО представляють собою різні лінії, що розташовані поряд, які розпізнаючий каскад сприйняв, як цифри. Зазвичай це нулі або одиниці, так як природні лінії і інші об'єкти представляють собою або прямі лінії, або невеликі замкнуті криві, схожі на коло. Також «хибні» цифри можуть виникнути в областях між сусідніми справжніми цифрами, у таких цифр зазвичай дуже маленьке значення площі.

Виходячи з цього, проведемо фільтрацію початкових «нулів» (якщо немає можливість існування початкових нулів у записі числа) і фільтрацію занадто близьких цифр одна до одної. Це означає, що як мінімум одна з них є помилковою. Визначити порядок цифр можна при відомому куті повороту числа за допомогою сортування вздовж великої осі.

2.4.2 Алгоритми для лінійних об'єктів

Аналогічно розпізнаванню і угрупованню ТО, лінійні об'єкти також можуть оброблятися як більш формальними, так і більш інтелектуальними методами. При цьому з'являється специфіка, яка не властива ТО: ЛО можуть бути достатньо великими на області ССЗ, тому можуть зазнавати велику кількість розривів, але при цьому саме їх розпізнавання є достатньо простим завданням: відрізнити суцільну лінію від штрихової, пунктирної або якоїсь іншої набагато простіше, ніж здійснити розпізнавання різних варіантів ТО, для цього можна скористатися алгоритмом, аналогічному наведеному на рисунку 2.8. Тому тут будемо вважати, що ЛО в семантичному плані представляють собою суцільні лінії.

Існує два різних підходи до сприйняття ліній: в першому вони мають деяку двовимірну природу (наприклад, мають ширину або товщину, яка може бути текстурована), у другому вони представляють собою швидше одновимірну сутність з одиничною шириною. Перший підхід не дає значної переваги при володінні більшою інформацією про об'єкти – часто дана інформація є зашумленою, що змушує використовувати або дуже прості методи, типу піксельного трекінгу, або складні методи, наприклад, FCNN. Мінус даного підходу є у тому, що якщо розглядати лінійний об'єкт двовимірно, то тоді ускладнюється угруповання таких об'єктів , так як вони можуть мати варіативну структуру. Тому зазвичай навіть двовимірні ЛО спочатку піддають скелетизації, тобто витонченню або виділенню кістяка. При скелетизації за околом Мура утворюються лінії, що перебувають з меншої кількості пікселів, чим за околом фон Неймана. Зберігання і обробка таких зображень займає менше часу при незначному зменшенні точності, тому є кращим варіантом.

Другий підхід набагато більше розвинений, так як дозволяє розробити алгоритми, що використовують широкий спектр математичних методів без використання інтелектуальних моделей. Відмова від використання

інтелектуальних моделей тут обґрунтовується тим, що рішення завдання угруповання для ЛО не має властивісті локальності: неможливо по існуючому зображення ковзного вікна визначити які саме лінійні сегменти потрібно з'єднати.

Строго геометричний підхід, який розглядає тільки кінцеві ділянки ЛО і їх властивості не може вирішити завдання повністю, навіть при значному кількості правильних з'єднань на усім зображення. Основні ідеї геометричного підходу спираються на обчислення відстаней і узгодження кутів між з'єднаними ділянками в ковзному вікні. Розроблені алгоритми носять ітеративний характер, що призводить до збільшення часу роботи алгоритмів. Протилежний підхід розглядає лінії в цілому і застосовується для всього зображення, при цьому лінії сприймаються як безліч точок, які взагалі не зобов'язані бути пов'язаними. Так, з'єднувати лінійні сегменти (під лінійними сегментами маються на увазі криволінійні, так як більше важлива «одномірна» природа цих сегментів) можна з використанням тріангуляції Делоне або діаграм Вороного, градієнтного векторного потоку або відстані Фреше. Нажаль, такі методи також не вирішують поставлених проблем і адекватно працюють тільки на очищених зображеннях з високим дозволом.

Запропонований алгоритм відрізняється використанням яклокальною інформації про кінцеві ділянки ЛО, так і інформації про їх перетин з іншими ЛО і використання функції максимального правдоподібності. У даному прикладі розглядається клас ЛО, які в семантичному плані не повинні мати перетинів один з другом. При можливій наявності перетинів ЛО мають зазвичай простішу структуру (наприклад, розмітка регулярною сітки) або зображені більше широкими лінями для більше чіткого відображення структури (карти або фотографії з зображенням доріг). Віддається перевага менше регулярним кривим щодо прямих відрізків: їх обробка також може бути зроблена більше простими методами, наприклад перетворенням Хафа .

Кінцевими точками (КТ) скелетизованою кривий вважаються точки, мають тільки одного сусіда. Якщо криві не мають перетинів, то вони можуть

бути відтворені тільки з лінійних сегментів, мають рівно дві КТ. У зворотному випадку повинні існувати точки розгалуження, мають більше двох сусідів. Якщо на нашому зображенні присутні точки розгалуження, то їх слід видалити. Наприклад, ЛО з трьома КТ і однієї точкою розгалуження буде перетворено в три ЛО, кожен з яких буде утримувати дві КТ.

Для кожного лінійного сегменту поставимо ідентифікатор. Його введення обумовлено необхідністю не розглядати в початку можливість з'єднання КТ одного сегменту, що може виникати на малих сегментах. Кожна КТ крім своїх безпосередніх координат може мати ще значенням кута дотичної в цих точках. Визначити дотичну для скелітизованого сегменту по визначенню неможливо через жорстку дискретизації кривих. Для вирішення цього питання можна або векторизувати лінійні сегменти, або розглядати їх «як є» у вигляді безлічі точок. Перший підхід поганий тим, що при параметричній задачі відрізків у вигляді функцій неможливо зрозуміти вплив похідної цих функцій для статичного зображення, при цьому похідна КТ може прийняти будь-яке значення.

При розгляді лінійних сегментів у вигляді множини точок кут дотичної можна визначити, використовуючи координати точок кривої поблизу КТ.

Враховуючи, що лінійний сегмент навіть у невеликому вікні може мати деяку кривизну (кривизну дискретної кривої можна визначити аналогічно визначенню дотичної через кілька сусідніх точок), то дана формула є зразковою, проте її застосування краще, ніж спосіб, в якому кут нахилу щодо до КТ визначається по однієї точці, яка лежить на деякому видалення від КТ , або яка є десятий точкою від КТ . При наявності кривизни запропонований алгоритм буде більше стабільним.

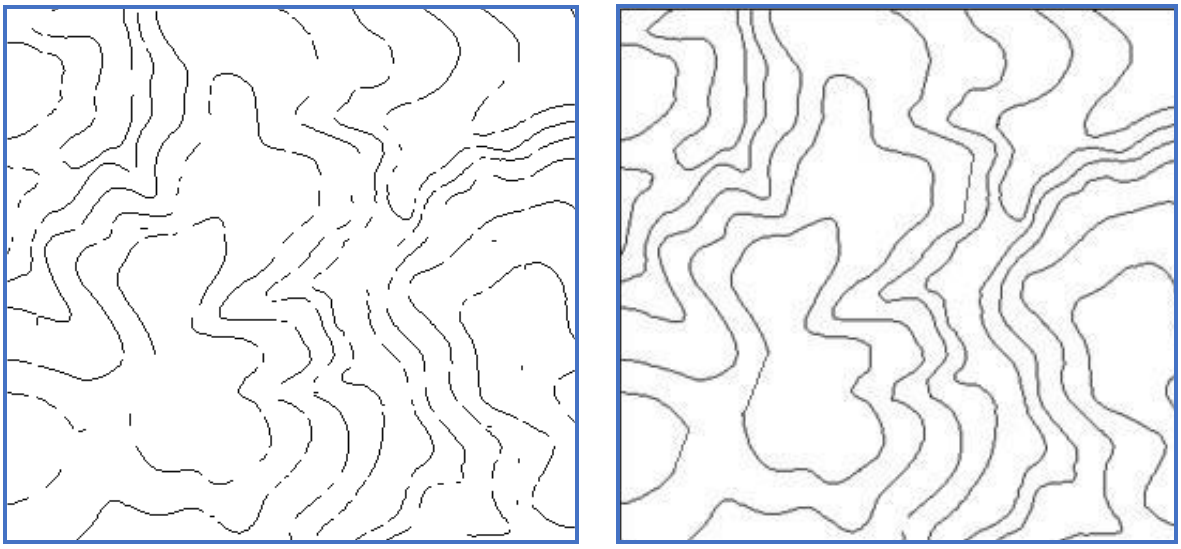
Під перетином розуміємо накладення матриці відрізка з'єднання КТ з вже наявною матрицею лінійних об'єктів. Обробка полягає у видаленні околів кінцевих точок і дилатації скелетизованої матриці ЛО. Дана обробка необхідна щоб уникнути хибного накладання відрізка з ЛО, які він з'єднує, і щоб запобігти пропусканню перетину, виникає через те, що відрізки мають

товщину в один піксель через дискретність піксельної матриці, можуть не мати фактичного перетину.

До теперішнього етапу залишилося дуже мало КТ, які об'єднані в небільші кластери (якщо кластер складається всього з двох КТ, то перевіряємо їх на з'єднання безпосередньо), для яких вже не можна однозначно визначити вірне зі- єднання, використовуючи геометричний підхід.

Отримати і проаналізувати такі кластери можна з допомогою попарного порівнювання.

Приклад роботи алгоритму використання розроблених відношень представлений на рисунку 2.13.



а б

Рисунок 2.13 Приклад роботи алгоритму угруповання горизонталей:
а – вихідне зображення; б – результат угруповання

Отже, проведений аналіз способів розпізнавання ССЗ з урахуванням їх характерних особливостей. Представлено загальна методика аналізу та обробки ССЗ, визначено проблеми, виникаючі на кожному етап, і розроблений набір загальних алгоритмів, які можна застосовувати до широкого класу ССЗ.

Розроблена загальна методика тяжіє до послідовного і комплексного аналізу і обробці ССЗ, а не концентрується на окремих етапах, що дозволяє розробнику алгоритмів глибше замислюватися о природі зображує мих об'єктів.

Розроблені алгоритми містять як можна менше число констант, існуючі порогові значення по можливості залежать від номери ітерації алгоритмів, а те, що вибирається в якості констант (наприклад, розміри ковзаючих вікон), пропонується для оптимізування в ході експериментів;

Розроблені алгоритми допускають необхідність додаткових досліджень для підбору операцій всередині алгоритму: якщо для алгоритму локалізації точкових об'єктів вдалося побудувати генетичний алгоритм, підбираючий морфологічні операції автоматично, то для угруповання точкових об'єктів та лінійних об'єктів підбір виконуваних операцій залишається прерогативою розробника програмного забезпечення.

3 ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА МЕТОДУ ПІДВИЩЕННЯ ТОЧНОСТІ АНАЛІЗУ ТА ОБРОБКИ СКЛАДНОСТРУКТУРНИХ ЗОБРАЖЕНЬ

3.1 Дослідження методу при обробці точкових об'єктів

Експериментальна перевірка розроблених алгоритмів виконувалася з допомогою пакету прикладних програм MATLAB R 2022 а під управлінням 64-розрядної операційної системи Windows 10 з 16 ГБ оперативної пам’яті з використанням відеокарти AMD Radeon 530 (4 ГБ). Орієнтованість мови MATLAB на виконання операцій з матрицями і векторами, а також наявність значної кількості вбудованих функцій і засобів візуалізації дозволила виконувати операції над зображеннями в більш зручному та наочному вигляді.

Помилками роботи алгоритмів із зазначенням часу їх виправлення можуть бути:

- неправильне розпізнавання, може бути виправлено за час, приблизно1 сек, та не залежить від кількості помилок;
- хибне спрацьовування, коли за числову позначку вказується деякий інший знайдений образ, може бути виправлена видаленням на протязі 1 сек.;
- пропуск позначки, час виправлення всіх таких позначок загалом залежить не тільки від кількості, а й від розміру зображення.

Видно, що помилки останнього типу є найбільш трудомісткими для оператора ЕОМ, тому їх кількість повинна бути мінімізована на рівні сегментації та локалізації.

У прикладі відома вся розмічувальна інформація: бінарна карта з чорними пікселями і таблиця з даними про позначки. Координати центру числа і його розміри необхідні для етапу локалізації, координати цифр і їх

значення потрібні для етапу розпізнавання, значення числа потрібно для етапу групування.

Першим етапом дослідження є перевірка моделей, які виконують сегментацію. Так як на даному етапі критичний пропуск необхідних пікселів, який може призвести до некоректної роботи алгоритмів наступних етапів, то як критерій якості будемо використовувати F_β -міру зі значенням $\beta = 2$.

Для навчання моделей для сегментації було вибрано по 1000 точок на кожному тестовому зображенні. Дані точки вибиралися центрами ковзних вікон, у тому числі проводилася вибірка ознак.

Результати за моделями представлені у таблицях 3.1 – 3.3.

Таблиця 3.1 – Результати при використанні формальних методів, $\alpha = 1$

Формула	$F\beta$, %	T, с/Мп
$0 \le R,G,B \le threshold = 160$	96,5±0,2	0,16
$0 \le \text{mean}(R,G,B) \le threshold = 150$	96,3±0,2	0,17

Таблиця 3.2 – Результати використання нейромережевих моделей

Розмір ковзного вікна, пікс.	Структура мережі	$F\beta$, %	T, с/Мп
1	3/3/1	96,9±0,2	0,60
	3/6/1	97,1±0,2	0,72
3	27/7/1	97,0±0,2	2,61
	27/13/1	97,1±0,2	2,85
	27/27/1	97,3±0,2	3,17
	27/54/1	97,2±0,2	4,56

Таблиця 3.3 – Результати під час використання інших моделей, $\alpha = 3$

Метод	Параметри	$F\beta$, %	T, с/Мп
DT	7 аркушів	95,7±0,4	0,99
35 листов	35 аркушів	95,1±0,4	1,08
SVM	лінійний	94,3±0,5	1,86
LDA	–	96,4±0,3	2,24

За результатами бачимо, що ускладнення моделей призводить до збільшення часу роботи алгоритму сегментації, але не завжди приводить до значущого покращення критерію якості. Це можна пояснити тим, що в даному дослідженні важливим вважається тільки один клас. При багатокласовій сегментації показники якості в середньому будуть нижче, тому там має сенс ускладнювати обробляємі моделі. У якості базової моделі для продовження дослідження була оббрана штучна нейронна мережа, котра відповідає такій структуі: 27/13/1.

Наступним етапом є локалізація. Для реалізації генетичного алгоритму були створені наступні функції, використовувані для отримання масочних зображень по даними розташування чисел:

- безпосереднього виконання генетичного алгоритму;
- обчислення функції пристосованості;
- реалізації генетичних операцій мутації і схрещування;
- обчислення центроміри хромосоми;
- видалення порожніх генів.

Хромосома зберігалася в вигляді масиву генів, кожен з яких мав таку структуру: ім'я операції і параметри операції. Кількість генів $N_{gen} > 5$.

У якості параметрів операцій могли бути назви СЕМО або границі для фільтрації.

Параметри генетичного алгоритму були обрані такими (:

- розмір популяції – 100 хромосом;
- максимальна кількість ітерацій – 100;
- частка елітних хромосом, що переходять у наступне покоління – 30%;
- ймовірність схрещування – 10%;
- ймовірність мутації – 90%;
- ймовірність вставки нового гена - $5 / N_{gen}$;
- ймовірність видалення гена - $0,05 (1 -1 / N_{gen})$;
- ймовірність модифікації гена - $0,95 - 4,95 / N_{gen}$;

- можливість взяти ген для мутації з лівої частини – 50%;
- можливість взяти ген для мутації з правої частини – 50%.

ГА для отримання найкращої хромосоми запускався на одному зображенні з тренувальних даних, так як на них присутня достатня кількість відміток глибин: 926, всього цифр 1764. ≈ 85 тис. пікселів.

Результати ГА при отриманні оптимальної хромосоми на цьому зображенні продемонстровані в таблиці 3.4 та рисунку 3.1.

Як критерій якості Recall використовувалося відношення знайдених критичних точок до їх кількості.

Таблиця 3.4 – Результати проведених ітерацій генетичного алгоритму

№	Nгенів	Recall , %	FF, %	№	Nгенів	Recall , %	FF, %
1	2	98,9	22,0	50	13	96,8	60,4
5	3	97,6	25,5	60	16	96,6	62,9
10	6	95,2	35,4	70	16	96,4	63,7
20	8	96,3	41,4	80	18	96,3	64,4
30	9	96,2	46,6	90	18	96,8	64,7
40	12	96,8	58,5	100	17	96,8	65,7

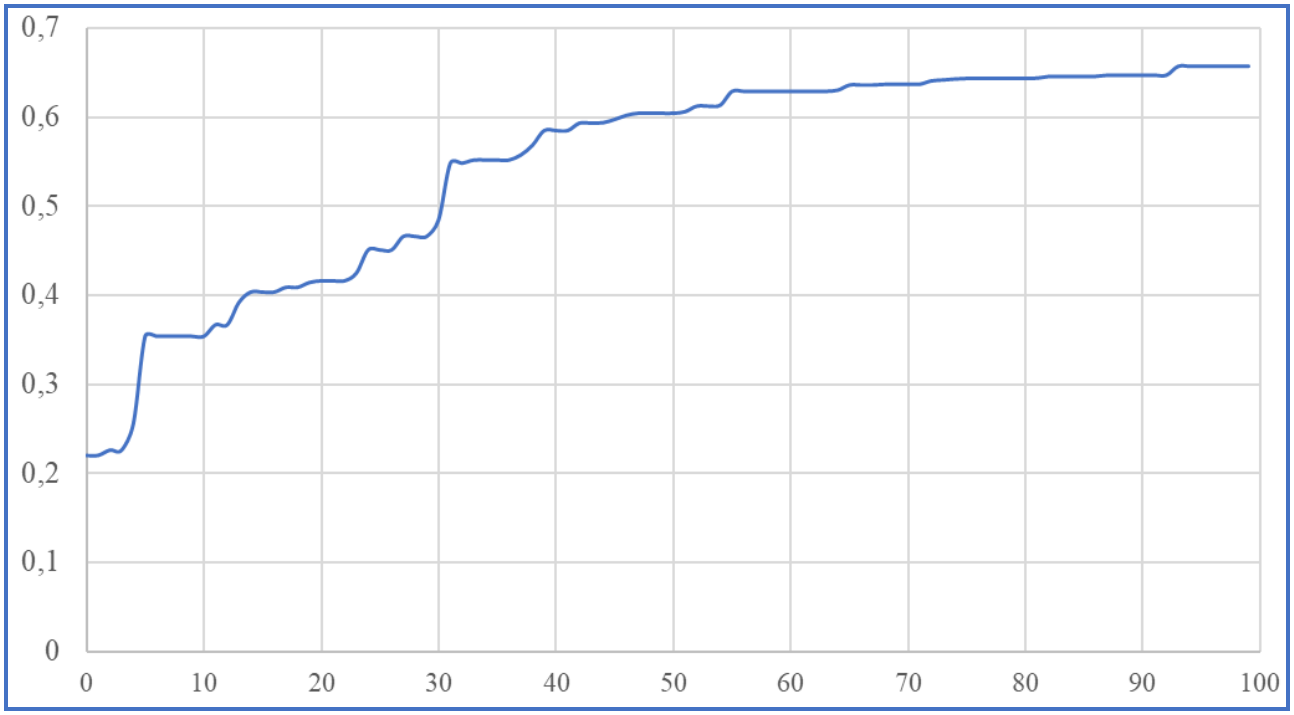


Рисунок 3.1 – Графік залежності функції пристосованості від номера ітерації ГА

У результаті сформована хромосома складається з наступних операцій:

- закриття, СЕ – горизонтальна лінія завдовжки 3;
- закриття, СЕ – вертикальна лінія завдовжки 3;
- відкриття, СЕ – квадрат розміру 2×2;
- закриття, СЕ – диск радіуса 2;
- бінаризація з порогом 0,39;
- закриття , СЕ - вертикальна лінія довжиною 6;
- фільтрація більшої півосі [5,4; 23,2];
- фільтрація за розміром сегмента в інтервалі [4, 268];
- відкриття, СЕ – вертикальна лінія завдовжки 3;
- дилатація, СЕ – горизонтальна лінія завдовжки 5;
- закриття, СЕ – квадрат розміру 2×2;
- фільтрація меншої півосі в інтервалі [5,3; 21,1];
- відкриття, СЕ – вертикальна лінія завдовжки 4;
- ерозія, СЕ – квадрат розміру 3×3;
- фільтрація за розміром сегмента в інтервалі [3, 466];
- дилатація, СЕ – горизонтальна лінія завдовжки 2;
- закриття, СЕ – вертикальна лінія завдовжки 2.

Ця хромосома була запущена на всіх 20 зображеннях. Отримані такі результати:

- швидкість обробки 0,16 с/Мп ,
- значення функції пристосованості дорівнює 64,4%±1,0%,

п– овнота дорівнює 96,1%±0,7%.

Дані результати означають, що з усіх відміток глибин лише 3,9% будуть втрачені, що є досить хорошим варіантом, так як ССЗ зазвичай мають значний ступінь надмірності, тому значення в точках цих відміток може бути відновлено з достатньою точністю. Значення функції пристосованості в 64,4% (враховуючи входження в цей критерій повноти) означає, що помилковопозитивних помилок буде не більше 50% кількості пікселів, що належать PictureIdeal .

Наступним етапом є розпізнавання цифр. На 10 тренувальних зображеннях були навчені наступні моделі:

- ШНМ прямого поширення з одним прихованим шаром,
- дерево рішень,
- kNN -класифікатор,
- машина опорних векторів.

Для kNN і SVM кількість тренувальних даних було скорочено для з відповіді часу роботи щодо інших моделей.

Розбивка на тренувальний, валідаційний та тестовий датасети відбувалася у пропорції 2:1:1.

У якості навчальних даних виступали зображення цифр розміру 13×11. У першому випадку в якості центрів образів використовувалися тільки центри цифр, во другому випадку до ним додавалися 4 сусідніх пікселя (всього 5), в третьому випадку ще 4 сусідніх по діагоналі пікселя (всього 9). Таким способом досліджувалося вплив аугментації на точність зазначених інтелектуальних моделей.

ШНМ навчалися з допомогою методу сполучених градієнтів, вихідний шар мав функцію softmax. Дерева рішень обмежувалися в 100, 1000 і 10000 розгалужень. kNN -класифікатор запускався для k, рівних 1, 3 і 5 сусідів. Використовувалося лінійне ядро для SVM .

Для поділу сегменту з числом на кілька сегментів також була навчена нейромережа з вихідним бінарним класом «цифра/не цифра»: позитивними прикладами класу вибиралися пікселі з квадрата 3×3 щодо центру, негативними – вершини і середини сторін квадрата 7 ×7 щодо центру.

У таблиці 3.5 представлені результати тестування першого каскада розпізнавання. Повнота розраховувалася щодо центрального квадрата 5×5 цифри: якщо в даної області був присутній хоч один піксель, що належить до правильному класу, то це збільшувало TP. Точність розраховувалася щодо всіх пікселів, зазначених до даному класу.

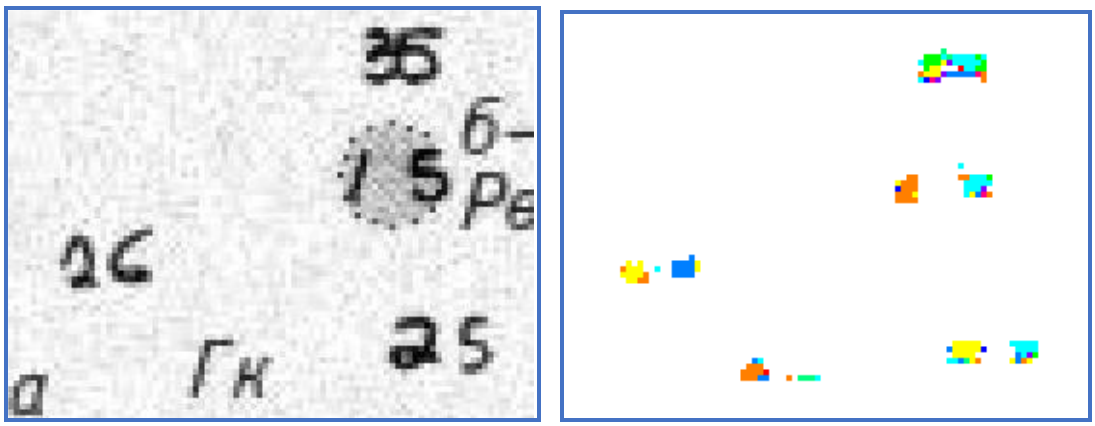
Таблиця 3.5 – Результати тестування першого каскада розпізнавання

Метод розпізнавання	Recall , %			Precision , %			T , с/ Мп
	Навчальна вибірка			Навчальна вибірка			
	1	5	9	1	5	9	
ANN , 143/50/10	77,0	91,3	93,0	18,2	26,3	28,8	0,55±0,02
ANN , 143/100/10	78,7	92,0	93,4	18,3	27,0	29,9	0,64±0,03
ANN , 143/200/10	79,0	92,7	94,0	18,8	27,1	32,8	0,74±0,03
ANN , 143/400/10	78,4	93,4	94,2	18,6	27,2	33,4	0,98±0,05
DT , 100	87,7	90,9	91,5	8,9	9,5	10,0	0,15±0,02
DT , 1000	90,7	93,9	93,6	8,3	10,3	10,6	
DT , 10000	90,7	95,3	95,5	8,3	10,0	11,1	
kNN , 1	90,6	92,1	92,2	9,2	9,7	11,0	1,1±0,1
kNN , 3	86,5	86,3	87,6	9,9	13,2	12,9	
kNN , 5	87,2	87,5	85,5	10,5	12,8	14,2	
SVM	81,3	80,0	75,6	10,7	12,0	12,6	1,8±0,2

Із аналізу таблиці 3.5 видно, що ШНМ і дерево рішень зберігають достатньо високий рівень повноти, але точність набагато вище у нейромереж. Відзначимо, що такий рівень точності (~30%) буде покращено після другого каскада і подальшої фільтрації на рівні угруповання. Повнота при збільшенні ауґментації значно росла тільки у ШНМ, точність збільшувалася у всіх моделей.

На рисунку 3.2 показаний приклад роботи першого каскада алгоритму для обраною ШНМ, має архітектуру 143/400/10, навченої з використанням повної аугментації. Рисунок 3.2, б зображений в індексному просторі:

- фон зображений білим,
- знайдений образ цифри «1» – помаранчевим,
- знайдений образ цифри «2» – жовтим,
- знайдений образ цифри «5» – ціановим,
- знайдений образ цифри «6» – блакитним.



а б

Рисунок 3.2 – Приклад роботи першого каскада алгоритму:
а – оригінальне зображення; б – отримані суперсегменти

Видно, що позначення ґрунту « Гк » також визначилося як цифра "1", що можна пояснити схожістю зображення в ковзному вікні 13×11. Так як цифри в числі «35» виявилися розташовані занадто близько друг до другові, то їх сегменти не були розділені.

Перед роботою другого каскада розпізнавання були проведено наступні операції над вийшли суперсегментами:

- видалення з них сегментів, перебувають з одного пікселя (у даної операції використовувалася околиця Мура, для решти – фон Неймана) ;
- видалення суперсегментів , що перебувають менше, чим з 5 пікселів;
- розбивка сегментів, перебувають більше, чим з 28 пікселів, з допомогою розрізу посередині.

Навчання інтелектуальних моделей відбувалося аналогічно попередньому каскаду. У якості навчальних даних були зроблено операції першого каскада розпізнавання над тренувальними зображення. Для кожного суперсегменту були знайдені частки входження в нього 10 шуканих класів.

Результати проведеного тестування показані в таблиці 3.6. Крім інтелектуальних моделей використовувалася також максимізація по частці. Так як на дано ном етапі обробляються більше високорівневі об'єкти з невеликий розмірністю вхідних параметрів, то час роботи всіх моделей виявилося приблизно однаковим і рівним 0,2 с/ Мп.

Таблиця 3.6 – Результати тестування другого каскада розпізнавання

Метод розпізнавання	**Recall, %**	**Common Precision, %**	**Identity Precision, %**
ANN , 10/10/10	84,4	82,2	92,8
ANN , 10/20/10	84,8	82,6	93,3
ANN , 10/100/10	85,0	82,9	93,6
DT , 10	59,2	57,7	65,1
DT , 20	66,2	64,5	72,8
DT , 100	83,6	81,4	91,9
DT , 1000	83,3	81,2	91,6
kNN , 1	83,2	81,1	91,5
kNN , 3	84,3	82,1	92,7
SVM	82,0	79,9	90,2
max	79,8	77,8	87,8

Останнім етапом є угруповання цифр в числа. Після первинної клатсеризації, реалізованою по алгоритму 2.8, була також проведено фільтрація утворених кластерів, яка частково відображає також і комплексний аналіз даних:

- видалення відміток, що мають значення менше 10 (складаються з однієї цифри), і не розташовуються на блакитному тлі, що означає мілководдя;

- видалення позначок зі значеннями 99 (складаються з 3 і більше цифр);

- видалення позначок зі значеннями, що відрізняються більше ніж на 15 від сусідніх позначок, знайдених за допомогою тріангуляції Делоне.

У таблиці 3.7 показано підсумкові результати застосування фільтрацій. Критерії якості показані для чисел загалом, а не для окремих цифр. Швидкість роботи, включаючи всі етапи фільтрації, становила 0,14 с/ Мп .

Таблиця 3.7 – Результати застосування фільтрацій

Етап	Recall, %	Common Precision, %	Identity Precision, %
Первинна кластеризація	77,4	70,6	78,4
Видалення позначок з 1 цифри на неголубом тлі	77,4	80,9	6,6
Видалення позначок з більше, ніж 2 цифр	77,4	82,9	87,7
Видалення позначок по порівнянні зі сусідніми	77,4	85,0	89,1

У результаті виходить, що з всіх позначок з допомогою отриманого алгоритму можна вірно визначити 89%, середній час на обробку однієї карти склав 18 секунд.

3.2 Дослідження методу при обробці площинних об'єктів

Для перевірки алгоритмів обробки площинних об'єктів використовувалися аерофотознімки з змагання DeepGlobe 2018 Land Cover Classification [12]. У датасеті кожному з 803 зображень розміру 2448×2448 була поставлено в відповідність маска, пікселі якої належали до одному з семи класів: urban, agriculture, rangeland, forest, water, barren, unknown (рисунок 3.3). Параметри гістограм класів ПлО визначалися для тренувального набору зображень тільки для тих пікселів, які не знаходилися на границі кількох класів.

Складність обробки таких зображень полягає не тільки в природній варіації розташованих на них об'єктів, але і розмітці даних достатньо поганої якості. При аналізі даних достатньо часто є зображення, на яких однаковим по виду просторам дані різні мітки або, навпаки, різні по виду об’єкти віднесено до одного класу. Так як об'єкти на зображенні можуть мати сильну варіацію в відображенні, то пропонується не використовувати підхід «один

клас – одна гістограма», а створити масив тренувальних даних, на основі яких сформувати інтелектуальну модель для подальшої класифікації.

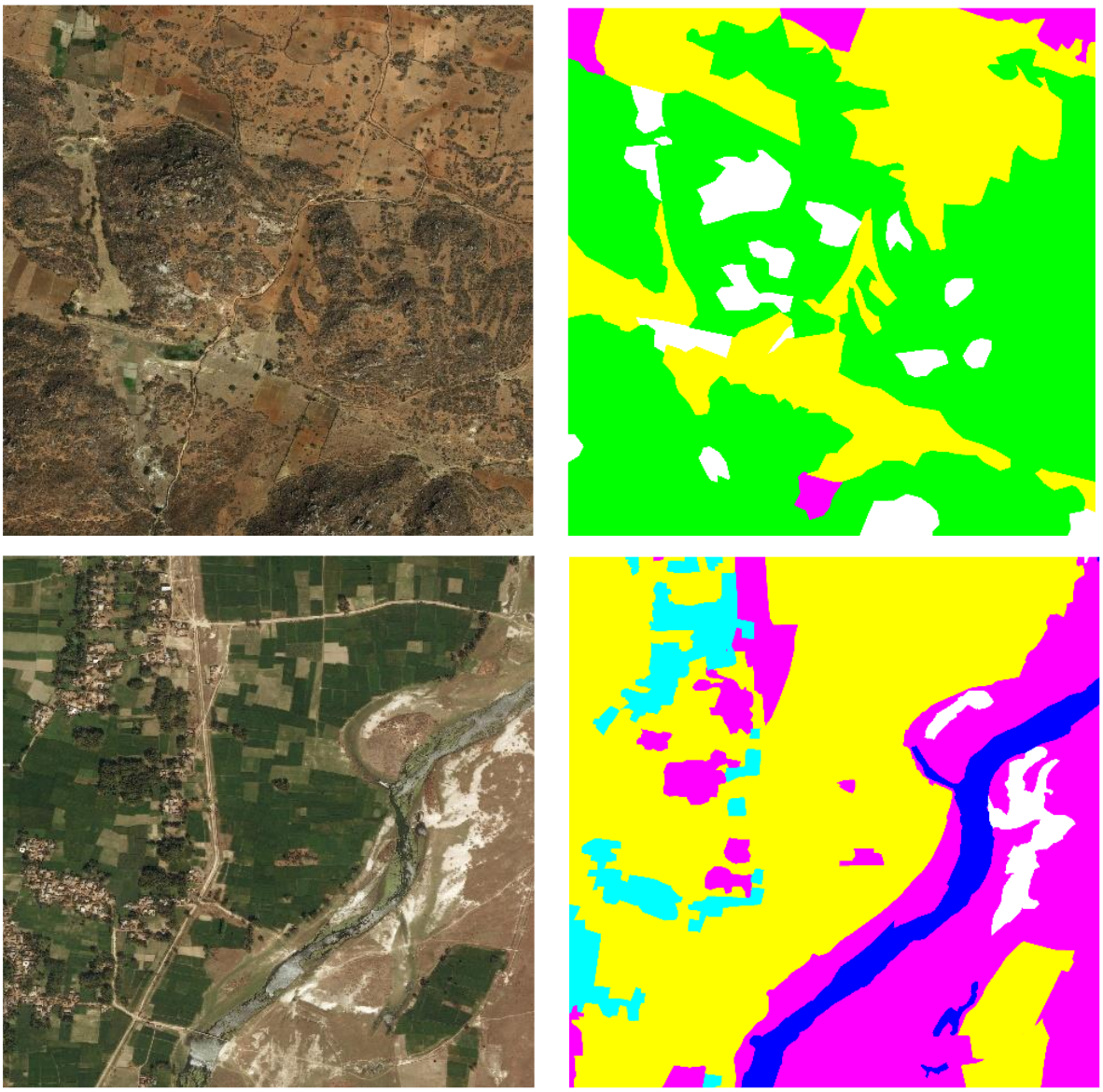


Рисунок 3.2 – Приклади зображень та відповідних масок DeepGlobe 2018

Експериментальне дослідження було засновано на отриманні тренувальних даних, навчанні інтелектуальної моделі та налаштуванні параметрів постобробки та тестування.

Функція GetData використовувалася для отримання тренувальних і валідаційних даних. У якості тренувальних зображень використовувалися 723 (90%) випадково обраних з вихідного датасета пар, що складаються із знімку та розміченої маски. Для того, щоб уникнути навчання на вікнах,

пікселі яких могли бути віднесені до різних класів, для кожного бінарного зображення класу маски була зроблена ерозія: серед кожного присутнього на масці класу випадково вибиралося по 100 точок, які були центрами для ковзаючих вікон. Околи були отримані з допомогою функції GetNeighborhood.

Гістограми будувалися для кожного з колірних просторів, розбиваючи їх на 32 інтервали, всього 96 ознак на об'єкт з допомогою функції img2hist . Завдяки особливостям вибірки вдалося зменшити сильну незбалансованість вихідних даних (таблиця 3.8).

Таблиця 3.8 – Розподіл класів оригінального датасету і тренувальних даних

Датасет	urban	agriculture	rangeland	forest	water	barren	unknown
Загальний	9,35	56,76	10,21	13,75	3,74	6,14	0,04
Вибірка	21,62	24,41	17,60	6,27	15,59	14,51	-

Приклад гістограм для класу forest для різних розмірів ковзаючих вікон показаний на рисунку 3.3. Видно, що гістограми для різних розмірів вікон достатньо схожі, але для маленького розміру вони є менше гладкими.

Так як ознаки для a = 11 і a = 25 приблизно схожі, а час, витрачений на обробку однієї точки, залежить від розміру вікна квадратично, то вибираємо a = 11.

У якості інтелектуальної моделі була обрано ШНМ прямого поширення з одним прихованим шаром (100) нейронів, функція активації сигмоїда), вихідний шар містив 6 виходів. Як оцінки ШНМ використовувалася перехресна ентропія, навчання відбувалося за допомогою методу сполучених градієнтів.

Навчання вироблялося на протязі 2,5 годин, підсумкове значення перехресної ентропії дорівнювало 0,20, що є прийнятним для ССЗ. На валідаційної вибірці (80 зображень з 723) був підрахований *mIoU*, який виявився дорівнює 30,8%.

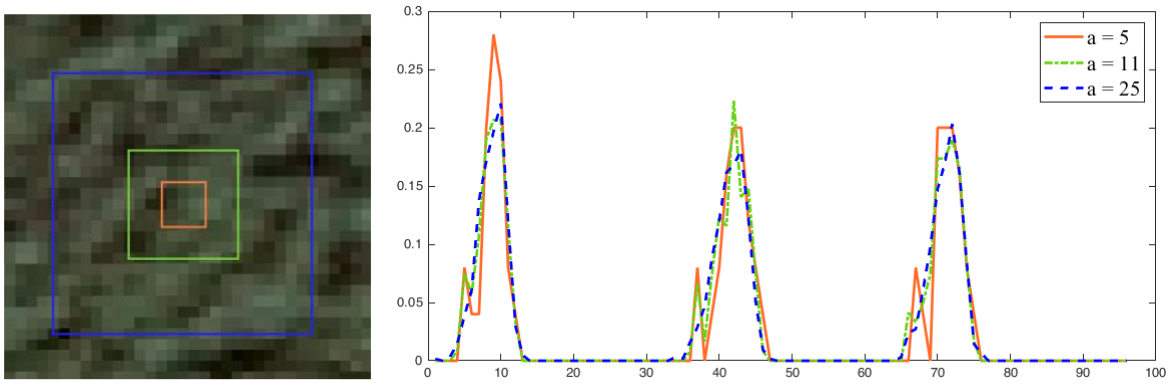
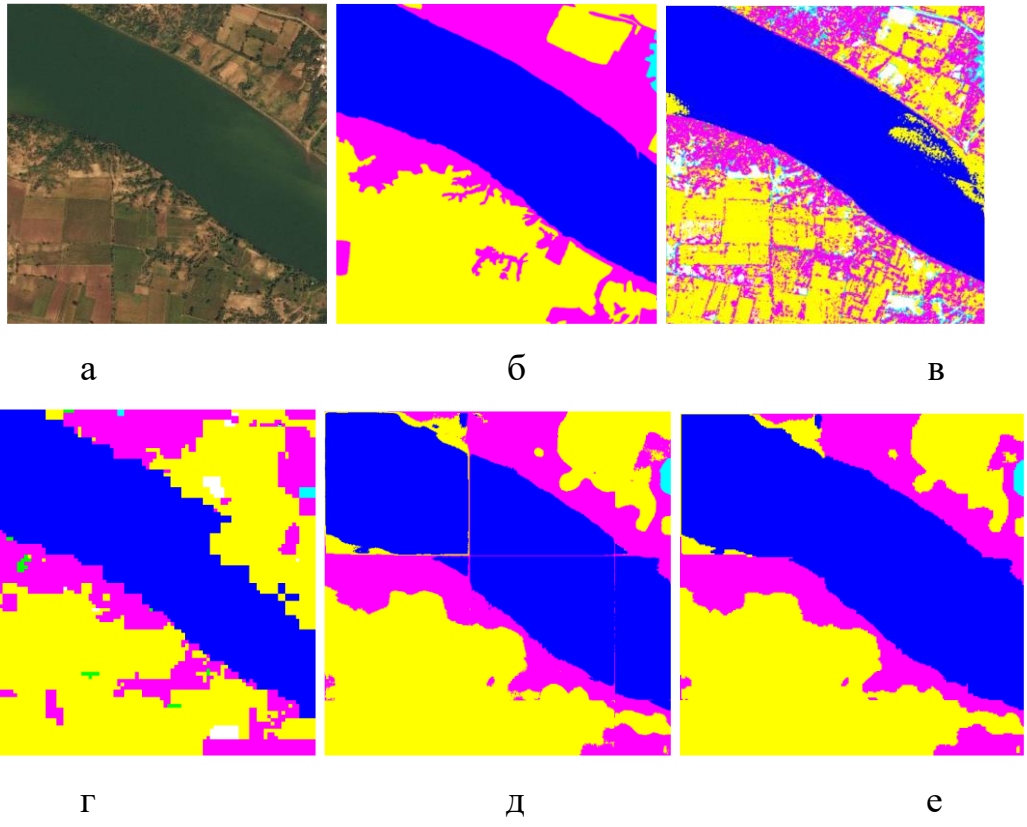


Рисунок 3.3 – Окіл випадкової точки класу forest та гістограми ковзних вікон

Тренувальна вибірка використовувалася у т.ч. год. для підбору параметрів постобробки.



а б в

г д е

Рисунок 3.4 – Ілюстрація роботи збудованого алгоритму:
а – вихідне зображення; б – вихідні розмічені дані; в – результат роботи ШНМ; г – відфільтроване зображення; д – результати роботи DeepLabv3+; е – відфільтроване зображення

Постобробка полягала у виконанні двох операцій:

- корекція вихідних значень ШНМ за допомогою коефіцієнтів;
- фільтрація за допомогою видалення невеликих сегментів та сегментація.

Ілюстрація роботи алгоритму показано на рисунку 3.4.

Використання розмірів вікон порядку кілька десятків призводить до зникнення деталей, але забезпечує гарну узагальнюючу здатність. Тим не менше слід відзначити, що вказаний підхід до постобробки може бути покращений, якщо збільшити кількість обробляючих операцій та/або кількість параметрів, що підбираються.

Отже, для дослідження алгоритмів обробки Пло були використані матеріали конкурсу DeepGlobe 2018 Land Cover Classification. Експеримент полягав в перевірці можливості використання нейромереж простої структури і наступною постобробки замість обробки за допомогою глибоких нейромереж.

У результаті дослідження була передбачена ШНМ зі структурою 96/100/6. З допомогою підбору були знайдені коригувальні коефіцієнти виходу нейромережі і розміри ковзаючих вікон фільтрів. З допомогою постобробки критерій якості для ШНМ піднято з 30,8 до 56,1%.

ВИСНОВКИ

Сукупність отриманих у кваліфікаційній роботі результатів дозволило вирішити актуальне науково-технічне завдання, спрямоване на підвищення точності аналізу та обробки складноструктурних зображень.

В результаті проведених досліджень отримані такі наукові та практичні результат.

1. Проведений аналіз сучасного стану аналізу та обробки растрових зображень з точки зору концепції складноструктурного зображення. Проведено класифікацію зображень за категоріями відображуваних об'єктів і з'ясовано ознаки складноструктурності. Були досліджені основні методи і моделі, що використовуються для попередньої обробки цифрових зображень, розпізнавання образів, розуміння зображення та принципи оцінювання цих методів.

2. Проведені дослідження підходів до локалізації та визначення типів сегментованих об'єктів. Після виконання попередніх алгоритмів були отримані напівтонові зображення, значення в яких означають міру належності пікселя вихідного зображення до певного класу об'єктів. Необхідно розділити ці зображення на окремі образи, тобто зробити їх локалізацію в просторі растру, підкласифікацію (наприклад, відділення текстових міток одного розміру, несучих одну функцію, від інших) і виявити їх більше абстрактні властивості, наприклад, просторовий тип об'єкта (точковий, лінійний, площинний), кут нахилу (для точкових об’єктів), товщину ліній (для локальних об’єктів) тощо.

3. Проведені дослідження підходів до до розпізнавання образів та їх угруповання з урахуванням їх характерних особливостей. Представлено загальна методика аналізу та обробки ССЗ, визначено проблеми, виникаючі на кожному етап, і розроблений набір загальних алгоритмів, які можна застосовувати до широкого класу ССЗ. Побудуваний генетичний алгоритм,

підбираючий морфологічні операції автоматично. Розроблена загальна методика тяжіє до послідовного і комплексного аналізу і обробці ССЗ, а не концентрується на окремих етапах, що дозволяє розробнику алгоритмів глибше замислюватися о природі зображує мих об'єктів.

4. Удосконалений та досліджений метод підвищення точності аналізу та обробки складноструктурних зображень за рахунок сегментації, локалізації, розпізнавання та групування точкових, лінійних і площинних об'єктів, що засновані на комплексному використанні відомих методів аналізу та обробки зображень. Для дослідження алгоритмів обробки ПлО були використані матеріали конкурсу DeepGlobe 2018 Land Cover Classification. Експеримент полягав в перевірці можливості використання нейромереж простої структури і наступною постобробки замість обробки за допомогою глибоких нейромереж. У результаті дослідження була передбачена ШНМ зі структурою 96/100/6. З допомогою підбору були знайдені коригувальні коефіцієнти виходу нейромережі і розміри ковзаючих вікон фільтрів. З допомогою постобробки критерій якості для ШНМ піднято з 30,8 до 56,1%.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Коваленко А. А., Чхеідзе В. О. Сучасний стан проблеми аналізу та обробки складноструктурних зображень. Системи управління, навігації та зв'язку. Полтава : Національний університет «Полтавська політехніка імені Юрія Кондратюка», 2025. Вип. 2(80). С. 132–136.

2. Ding, P. A light and faster regional convolutional neural network for object detec-tion in optical remote sensing images / P. Ding et al. // ISPRS Journal of Photogrammetry and Remote Sensing. – 2018. – V. 141. – Pp. 208-218.

3. Kouril, D. Labels on Levels: Labeling of Multi-Scale Multi-Instance and Crowded 3D Biological Environments / D. Kouril et al. // IEEE Transactions on Visualization and Computer Graphics. – 2019. – V. 25(1). – Pp. 977-986.

4. Koutnik, J. Evolving Large-Scale Neural Networks for Vision-Based Reinforce-ment Learning / J. Koutnik et al. // GECCO 2013, Proc. of the Genetic and Evolutionary Computation Conference. – 2013. – Pp. 1061-1068. – doi:10.1145/2463372.2463509.

5. Kuo, T. Deep Aggregation Net for Land Cover Classification / Proc. of 2018 IEEE/CVF Conference CVPRW. – 2018. – p. 247-2474. – doi: 10.1109/CVPRW.2018.00046.

6. LeCun, Y. Gradient-based learning applied to document recognition / Y. LeCun, L. Bottou, Y. Bengio, P. Haffner // Proceedings of the IEEE. – 1998. – V. 86(11). – Pp. 2278-2323.

7. Leyk, S. Improving feature extraction of composite cartographic information in low-quality maps / S. Leyk, R. Boesch // Cartography and Geographical Information Science. – 2009. – 36(1). – Pp. 71-79.

8. Li, C. A reconstruction method for broken contour lines based on similar contours / C. Li et al. // ISPRS International Journal of Geo-Information. – 2019. – V. 8(1). – No. 8. – 14 p. – doi:10.3390/ijgi8010008.

9. Li, H. Intelligent Map Reader: A Framework for Topographic Map

Understanding With Deep Learning and Gazetteer / H. Li et al. // IEEE Access. – 2018. – V. 6. – Pp. 25363-25376. – doi:10.1109/ACCESS.2018.2823501.

10. Liao, P.-S. A fast algorithm for multilevel thresholding / P.-S. Liao, T.-S. Chen, P.-C. Chung // Journal of Information Science and Engineering. – 2001. – V. 17. – Pp. 713-727.

11. Moreno-Garcia, C. F. New trends on digitisation of complex engineering drawings / C. F. Moreno-Garcia, E. Elyan, C. Jayne // Neural Computing and Applications. – 2019. – V. 31(6). – Pp. 1695-1712. – doi:10.1007/s00521-018-3583-1.

12. Pinto, A. T. From archived historical aerial imagery to informative orthophotos: A framework for retrieving the past in long-term socioecological research [Technical Note] / A. T. Pinto et al. // Remote Sensing. – 2019. – V. 11(11). – No. 1388. – 15 p. – doi:10.3390/rs11111388.

13. Uhl, J. H. Automated Extraction of Human Settlement Patterns From Historical Topographic Map Series Using Weakly Supervised Convolutional Neural Networks / J. H. Uhl et al. // Access IEEE. – 2020. – V. 8. – Pp. 6978-6996

14. Wei, H. DEANet: Dual Encoder with Attention Network for Semantic Segmentation of Remote Sensing Imagery // Remote Sens. – 2021. – № 13:3900. – doi:10.3390/rs13193900

15. Xydas, C. Buildings Extraction from Historical Topographic Maps via a Deep Con-volution Neural Network / C. Xydas et. al // 17th International Conference on Computer Vision Theory and Applications, VISIGRAPP 2022. – 2022. – V. 5. – Pp. 485-492. – doi:10.5220/0010839700003124

16. Yan, X. A graph convolutional neural network for classification of building pat-terns using spatial vector data / X. Yan et al. // ISPRS Journal of Photogrammetry and Remote Sensing. – 2019. – V. 150. – Pp. 259-273. – doi:10.1016/j.isprsjprs.2019.02.010

17. Zhang, Zh. MKANet: An Efficient Network with Sobel Boundary Loss for Land-Cover Classification of Satellite Remote Sensing Imagery // Remote Sens. – 2022. – №. 18:4514. – doi:10.3390/rs14184514