

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ

Кваліфікаційна наукова
праця на правах рукопису

Тимченко Борис Ігорович

УДК 004.932.72'1

ДИСЕРТАЦІЯ

**НЕЙРОМЕРЕЖЕВІ МЕТОДИ АНАЛІЗУ ПЛАНАРНИХ
ЗОБРАЖЕНЬ В СИСТЕМАХ АВТОМАТИЗОВАНОГО
СКРИНІНГУ**

122 — Комп'ютерні науки
Інформаційні технології

Подається на здобуття наукового ступеня
доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів
і текстів інших авторів мають посилання на відповідне джерело.

_____ Б. І. Тимченко

Науковий керівник: **Антощук Світлана Григорівна**,
доктор технічних наук, професор

Одеса — 2021

АНОТАЦІЯ

Timchenko B. I. Нейромережеві методи аналізу планарних зображень в системах автоматизованого скринінгу. — Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 — Комп'ютерні науки. — ІКС ОНПУ, Одеса, 2021.

У **вступі** обґрунтовано актуальність удосконалення систем автоматизованого скринінгу (виявлення та класифікації нетипових об'єктів або процесів) шляхом створення моделей наборів даних та методів їх генерації, а також моделей нейронних мереж та методів багатозадачного навчання з метою підвищення достовірності класифікації та сегментації планарних зображень без підвищення витрат часу. Визначено об'єкт, предмет, задачі і методи дослідження; показано зв'язок з науково-дослідними роботами; наведено наукову новизну та практичне значення одержаних результатів; висвітлено особистий внесок здобувача.

В **першому розділі** дисертаційної роботи проведено аналіз проблем аналізу планарних зображень (які не мають виміру глибини, та в яких можна знехтувати масштабом об'єктів) в системах автоматизованого скринінгу на основі нейромережевих технологій.

Проаналізовано існуючі системи автоматизованого скринінгу в предметних областях медицини, метеорології та віддаленого зондування та особливості їх функціонування шляхом аналізу планарних зображень. Показано, що ефективність роботи таких систем безпосередньо залежить від достовірності класифікації та сегментації зображень, оскільки хибно-позитивні результати можуть привести до передчасного реагування, а хибно-негативні

результати - до пропуску наявних проблем та пізнього реагування.

Крім того, не менш важливим фактором, що впливає на ефективність скринінгових систем є витрати часу на реалізацію процесу навчання, що в свою чергу пов'язано з якістю попередньо створених навчальних вибірок даних. Але, в задачах скринінгу, отримання достатньої кількості даних та проведення якісного аnotування професіоналами є, як правило, затратним, а іноді неможливим через малу кількість та різний вигляд навчальних прикладів, суб'єктивність під час формування аnotацій в наборах даних. Ці фактори негативно впливають на достовірність класифікації та сегментації при навчанні глибинних нейронних мереж стандартними методами. Для усунення такого впливу запропоновано при побудові моделей класифікації та сегментації вирішити задачу навчання з урахуванням частково-помилкових аnotацій.

Показано, що в таких умовах доцільне використання глибинних нейронних мереж, через можливість автоматизації процесу навчання та застосування методів багатозадачного навчання, які підвищують достовірність класифікації та сегментації.

Проведений аналіз наявних моделей наборів даних та методів їх генерації показав обмеженість моделей наборів даних, які б дозволяли формувати частково-помилкові аnotації, які характерні для реальних задач автоматизованого скринінгу.

Таким чином, вирішення важливої науково-практичної задачі підвищення достовірності класифікації та сегментації в задачах аналізу планарних зображень для систем автоматизованого скринінгу шляхом удосконалення моделей нейронних мереж та методів їх багатозадачного навчання, особливо за умови частково-помилково аnotованих даних навчальної вибірки.

У другому розділі розроблено метод генерації наборів даних для забезпечення можливості тестування нейронних мереж при навчанні з використанням частково-помилкових аnotацій та запропоновано параметричну

формалізацію моделі набору даних із частково-помилковими анотаціями, які характерні для реальних задач автоматизованого скринінгу.

Оскільки одержання достовірних анотацій класифікації та сегментації для тестувальних вибірок часто є неможливим або дуже затратним, запропоновано використання модельних наборів даних для оцінки достовірності моделей та методів навчання глибинних нейронних мереж. Такі набори даних дають можливість проводити навчання та тестування в контролюваних умовах шляхом штучного додавання помилок до анотацій навчальної вибірки даних (як правило, тестова вибірка залишається з достовірними анотаціями).

Запропонована модель набору даних \mathcal{M} має наступне представлення:

$$\mathcal{M} \in \{\mathcal{X}_b, \mathcal{X}_{tex}, N, S_{img}, S_{obj}, \Delta_{max}, N_{obj}, P_e, P_d, S_e, S_d\} \quad (1)$$

де \mathcal{X}_b - набір зображень фону, \mathcal{X}_f - набір об'єктів, \mathcal{X}_{tex} - набір зображень текстур об'єктів, N - кількість зображень в генерованому наборі даних, S_{img} - розмір генерованих зображень в пікселях, S_{obj} - середній розмір об'єкта в пікселях, Δ_{max} - максимальне відхилення розміру об'єкта в відсотках, N_{obj} - максимальна кількість об'єктів на зображені, P_e та P_d - ймовірності зменшення та збільшення маски кожного з об'єктів, S_e та S_d - допустимі масштаби збільшення та зменшення масок всіх об'єктів.

Останні чотири параметри введено для контролюваного створення помилок в анотаціях.

Сформульовано **перший пункт наукової новизни**: вперше запропоновано параметричну формалізацію моделі набору даних із частково-помилковими анотаціями, які характерні для реальних задач автоматизованого скринінгу, що дозволило розробити метод генерації навчальних, тестових та валідаційних наборів даних.

На основі запропонованої параметричної моделі (9) розроблено метод генерації наборів даних із частково-помилковими анотаціями, який містить

наступні кроки:

1. Вибрати випадкове зображення фону: $x_{bg} \sim \mathcal{X}_b$
2. Вибрати кількість об'єктів на зображені: $N_{obj} \sim \mathcal{U}(1, N_{obj})$
3. Провести ініціалізацію маски сегментації: $M = 0$ так що $M \in \mathcal{R}^{C \times S_{img} \times S_{img}}$
4. Виконати наступні кроки n_{obj} разів:
 - 4.1 Вибрати розміри об'єкта: $s \sim \mathcal{U}(S_{obj} - \Delta_{max}, S_{obj} + \Delta_{max})$
 - 4.2 Вибрати координати розміщення об'єкта:

$$i_f \sim \mathcal{U}(0, S_{img} - s)$$

$$j_f \sim \mathcal{U}(0, S_{img} - s)$$

4.3 Вибрати зображення об'єкта $x_{fg} \sim \mathcal{X}_f$ та відповідний клас об'єкта $c_{fg} \sim \mathcal{Y}_f$

4.4 Змінити розмір зображення об'єкта за допомогою білінійної інтерполяції:

$$\hat{x}_{fg} = R_{bilinear}(x_{fg})$$

4.5 Вибрати зображення текстури $x_{tex} \sim \mathcal{X}_{tex}$

4.6 Модифікувати зображення об'єкта за допомогою текстури:

$$\hat{x}_{fg} = x_{fg} \circ x_{tex}[i_f : i_f + s, j_f : j_f + s]$$

4.7 Розмістити зображення об'єкта на зображені фону:

$$x_{bg}[i_f : i_f + s, j_f : j_f + s] = (1 - x_{fg}) \circ x_{bg} + \hat{x}_{fg}$$

4.8 Сформувати маску сегментації об'єкта:

$$M_{seg} = x_{fg} > \theta_{seg}$$

де θ_{seg} - поріг бінаризації вихідного зображення об'єкта. Для набору даних MNIST $\theta_{seg} = 0.2$, для набору даних FashionMNIST $\theta_{seg} = 0.1$.

4.9 Модифікувати маску сегментації відповідно до необхідного рівня помилок:

$$M_{seg} = \begin{cases} M_{seg} \oplus K^{S_d \times S_d} & \text{якщо } p_d \sim \mathcal{U}(0, 1) < P_d \\ M_{seg} \ominus K^{S_e \times S_e} & \text{якщо } p_e \sim \mathcal{U}(0, 1) < P_e \end{cases}$$

де $K^{S_d \times S_d}$ - матриця ядра $K^{S_e \times S_e}$ - матриця ядра ерозії.

4.10 Розмістити модифіковану маску сегментації об'єкта на загальному зображенні маски сегментації:

$$M[c_{fg}, i_f : i_f + s, j_f : j_f + s] = \max \{M[c_{fg}, i_f : i_f + s, j_f : j_f + s], M_{seg}\}$$

4.11 Зберегти зображення x_{bg} та маску M

5. Завершити генерацію

Таким чином, удосконалено метод генерації наборів даних із частково-помилковими анотаціями на основі параметричної моделі, що за рахунок що за рахунок генерації анотацій: частково-помилкових для тренувальної вибірки та достовірних - для тестової, дало можливість виконувати тестування впливу рівня помилок анотацій на роботу нейромережевих методів сегментації та класифікації.

Метод дозволяє отримати безліч наборів даних із схожими характеристиками та використовувати непараметричні статистичні методи (бутстрепінг) для оцінки моделей в умовах відсутності реальних тренувальних даних. Метод становить **другий пункт наукової новизни**.

В третьому розділі дисертаційної роботи розроблено модель нейронної мережі та метод багатозадачного навчання для одночасного підвищення достовірності класифікації та сегментації без зниження оперативності.

Для реалізації методів багатозадачного навчання, запропоновано удосконалити модель глибинних нейронної мережі з використанням архітектури енкодер-декодер (UNet, LinkNet) введенням додаткового декодера з шаром нормалізації. Таким чином, вдосконалена модель складається з

енкодера та двох декодерів (для задач сегментації та класифікації відповідно).

Модель глибинної нейронної мережі представлена наступним виразом:

$$v_1, v_2 \dots v_n = F_{encoder}(x, \theta_{enc}) \quad (2)$$

$$M_{seg} = F_{seg}((v_1, v_2 \dots v_n), \theta_{seg}) \quad (3)$$

$$C_{cls} = F_{cls}((v_n), \theta_{cls}) \quad (4)$$

де θ_{enc} - набір параметрів енкодера, θ_{seg} та θ_{cls} - набори параметрів декодерів сегментації та класифікації відповідно, F_{seg} та F_{cls} - нейронні мережі декодера сегментації та класифікації відповідно. Запропонована модель складає **третій пункт наукової новизни**.

Завдяки введенню додаткового декодера класифікації з шаром нормалізації, з'явилася можливість реалізації методів багатозадачного навчання глибинних нейронних мереж та передбачення результатів.

Запропоновано метод багатозадачного навчання нейронних мереж в умовах частково-помилкових анотацій навчальних даних, який спирається на використання задач, пов'язаних з оригінальною. Показано, що для задачі сегментації існує близька задача класифікації, для якої анотації навчальних даних є якіснішими, ніж для вихідної задачі сегментації.

Метод багатозадачного навчання складається з двох етапів.

Етап 1: генерація похідної задачі. В контексті багатозадачного навчання, задача класифікації зводиться до визначення набору сегментованих зразків (англ. multiple instance learning) на планарному зображені, замість маркування кожного з об'єктів для всіх класів. При цьому анотація вихідного планарного зображення являє собою множину з одним, чи декількома визначеними об'єктами.

Дляожної з задач (класифікація та сегментації) окремо обчислюється функція втрат. Для навчання декодера сегментації використовується обме-

жена зверху функція втрат $L_{seg}]$, в той час як для декодера класифікації - звичайна L_{cls} , а загальне значення функції втрат визначається як середнє арифметичне між індивідуальними значеннями:

$$L_{total} = \frac{L_{seg}] + L_{cls}}{2} \quad (5)$$

Відповідно, загальний градієнт функції втрат буде також середнім арифметичним градієнтів складових частин $\nabla L_{seg}]$ і ∇L_{cls} :

$$\nabla L_{total} = \frac{\nabla L_{seg}] + \nabla L_{cls}}{2} \quad (6)$$

Таким чином, забезпечується наявність ненульових градієнтів для оновлення параметрів від хоча б однієї функції втрат для кожного вхідного прикладу.

Етап 2: Введення обмежень. Для зменшення впливу помилкової частини анотацій запропоновано ввести обмеження другого роду (зверху) до функції втрат для задачі сегментації з менш якісними анотаціями даних. Це дозволило при навчанні на декількох задачах зменшити вплив градієнтів функції втрат на прикладах з помилковими анотаціями:

$$\mathcal{L}] = \min(L, \theta)$$

де θ - поріг обмеження функції втрат.

Введені обмеження представлені наступним виразом:

$$\nabla \min(L, \theta) = \begin{cases} 1, & \text{якщо } L \in (-\infty, \theta] \\ 0, & \text{якщо } L \in (\theta, \infty) \end{cases}$$

Метод прогнозування містить наступні етапи:

Етап 1. Нормалізація. Нехай $C_{cls} \in \mathcal{R}^C$ та $M_{seg} \in \mathcal{R}^{C \times H \times W}$ - результати декодерів класифікації та сегментації відповідно, значення яких знаходяться на проміжку $(-\infty, +\infty)$ (логіти).

Для отримання результатів на проміжку $[0, 1]$ використовується логістична сигмоїдна функція активації:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Етап 2. Ренормалізація. Ренормалізація полягає у зважуванні карт сегментації за допомогою нормованих логітів класифікатора. Першим кроком є трансформація логітів сегментації та класифікації в некалібровані оцінки на проміжку $[0, 1]$:

$$\begin{aligned}\hat{M}_{seg} &= \sigma(M_{seg}) \\ \hat{C}_{cls} &= \sigma(C_{cls})\end{aligned}$$

Ці оцінки мають ті самі розмірності, що й оригінальні маска та класи, для зручності репрезентації операцій додано додаткові розмірності до вектору класів: $\hat{M}_{seg} \in \mathcal{R}^{C \times H \times W}$ та $\hat{C}_{cls} \in \mathcal{R}^{C \times 1 \times 1}$

Зважування карти сегментації відбувається за допомогою добутку Адамара між матрицями \hat{M}_{seg} та \hat{C}_{cls}

$$M_{refined} = \hat{M}_{seg} \circ \hat{C}_{cls}$$

В умовах відсутності розмітки для прикладів в задачах сегментації, можливе ітеративне уточнення цієї розмітки. Для цього обчислюється уточнена карта ознак класифікації шляхом добутку Адамара між нормованим за допомогою сигмоїдної функції виходом декодера сегментації та логітами класифікації:

$$M_{unsup} = \hat{M}_{seg} \circ C_{cls} \tag{7}$$

Далі, для отримання результату класифікації виконується сумація елементів M_{unsup} з нормалізацією за сумою елементів оригінальної ненормалізованої карти локалізації:

$$C_{unsup} = \frac{\sum_{h=0}^H \sum_{w=0}^W M_{unsup(h,w)}}{\sum_{h=0}^H \sum_{w=0}^W M_{seg(h,w)} + c} \tag{8}$$

На основі запропонованих моделі нейронної мережі та методу її навчання, розроблено метод багатозадачного передбачення. Це дозволило підвищити достовірність класифікації та сегментації планарних зображень без зниження оперативності.

За допомогою моделі (9) протестовано запропоновані методи в різних умовах, виконаний аналіз внеску окремих компонентів та проведено аналіз стійкості запропонованого методу до різних рівнів помилок в анотаціях. В середньому, підвищення коефіцієнта Дайса відносно базової моделі склало 13%.

Сформульовано **четвертий пункт наукової новизни**: удосконалено методи багатозадачного навчання та передбачення результатів на основі удосконаленої моделі згорткових нейронних мереж шляхом об'єднання класифікації та сегментації і введення обмеження другого роду (зверху) при обчисленні функції втрат сегментації, що дозволило підвищити достовірність сегментації та класифікації в задачах автоматизованого скринінгу.

У **четвертому розділі** розроблені інструментальні засоби, що реалізують запропоновані рішення. Проведено випробування розробленого методу в рамках експериментів як на синтетичних даних, що були згенеровані за допомогою запропонованої моделі, а також експерименти в реальних задачах: скринінг діабетичної ретинопатії, скринінг меланоми, та скринінг хмарних утворень.

Інструментальні засоби розроблено мовою програмування Python з використанням фреймворку автоматичного диференціювання PyTorch. На основі розроблених інструментальних засобів створено ефективні програмні модулі, які інтегровано з “хмарними” сервісами для вирішення ресурсомістких задач навчання нейронних мереж, що забезпечує високу обчислювальну потужність та швидкість прогнозування в задачах автоматизованого скринінгу.

Для задачі автоматизованого скринінгу при сегментації патернів орга-

нізації хмар на супутникових знімках (в рамках проекту “Understanding Clouds from Satellite Images” на платформі для змагань з наук про дані Kaggle) було використано запропоновані моделі нейронних мереж, а також методи їх навчання та прогнозування результатів. Підвищення достовірності (міра Дайса) відносно базової моделі склало 3.9%.

Для задачі автоматизованого скринінгу при класифікації стадій діабетичної ретинопатії (в рамках проекту “APROS 2019 Blindness Detection” на платформі для змагань з наук про дані Kaggle) було використано запропоновані методи багатозадачного навчання та прогнозування результатів. Підвищення достовірності (F1-міра) відносно базової моделі склало 2.1%.

Для задачі розпізнавання уражень шкіри при скринінгу меланоми (у рамках проекту SIIM-ISIC Melanoma Classification на платформі для змагань з наук про дані Kaggle) було використано запропонований локалізації важливих для класифікації ознак зображення. Використання запропонованого методу дозволило спростити процес контролю навчання нейронних мереж, що допомогло попередити перенавчання і підвищити достовірність класифікації на 3.5%.

Розроблені в роботі методи та інструментальні засоби отримали впровадження в навчальний процес ОНПУ та програмний продукт SafetyRadar компанії VITech Lab, основним призначенням якого є скринінг наявності елементів засобів індивідуального захисту на людях в умовах будівельних майданчиків, або лікарень та лабораторій.

Ключові слова: аналіз зображень, модель даних, глибинні нейронні мережі, багатозадачне машинне навчання, сегментація, класифікація, функції втрат.

Список публікацій здобувача за темою дисертації.

- on Data Science and Intelligent Analysis of Information / Springer. 2018. C. 20–29. (Index Copernicus)
<https://www.springerprofessional.de/en/race-from-pixels-evolving-neural-network-controller-for-vision-b/16003024>
7. Tymchenko B, Hramatik A., Tulchyi H., Antoshchuk S. Making money: Evolving neural network for stock prediction // VI українсько-німецька конференція «Інформатика. Культура. Техніка». 2018. C. 34-35
8. Tymchenko B., Antoshchuk S. Evolution strategy for policy search in robotics // Сучасні Інформаційні Технології / Одеський Національний Політехнічний Університет. 2018.
<http://dspace.opu.ua/jspui/handle/123456789/8056>
9. Tymchenko B., Halchonkov O. Online lane detection algorithm for line scan camera // Сучасні Інформаційні Технології / Одеський Національний Політехнічний Університет. 2017. C. 86-89
<http://dspace.opu.ua/jspui/handle/123456789/3351>
10. Tymchenko B. Global position system sensor model for robotics simulator // Праці Одеського політехнічного університету. 2017. № 3. C. 88–93.
<http://dspace.opu.ua/jspui/handle/123456789/7939>
11. Tymchenko B., Samodelok V., Putilina D., Galchonkov O. The robust control system for skid elimination in dynamic road environments // Електротехнічні та комп’ютерні системи. 2016. № 23. C. 107–112.
<http://dspace.opu.ua/jspui/handle/123456789/1176>
12. Komleva N., Cherneha K., Tymchenko B., Komlevoy O. Intellectual approach application for pulmonary diagnosis 2016 IEEE First International Conference on Data Stream Mining Processing (DSMP). 2016. C. 48–52.
<https://ieeexplore.ieee.org/document/7583505/>
13. Cherneha K., Tymchenko B., Komleva N. Decision support system for

automated medical diagnostics // Електротехнічні та комп'ютерні системи. 2016. № 23. С. 65–72. (Index Copernicus)

http://dspace.opu.ua/xmlui/handle/123456789/1140

ABSTRACT

Tymchenko B. I. Neural Network Methods for Planar Image Analysis in Automated Screening Systems. — Qualification scientific work in the form of manuscript.

Thesis for doctor of philosophy degree in speciality 122 — Computer science. — ICS ONPU, Odesa, 2021.

In **introduction**, the relevance of improving automated screening systems (screening is defined as detection and classification of atypical objects or processes) by creating models of datasets and methods for their generation, as well as models of neural networks and multi-tasking learning methods to increase the accuracy of classification and segmentation of planar images without increasing time. The object, subject, tasks and methods of research are defined; the connection with research works is shown; the scientific novelty and practical significance of the obtained results are given; the personal contribution of the applicant is covered.

In the **first section** of the dissertation the analysis of problems of analysis of planar images (which do not have depth measurement, and in which it is possible to discard scale of objects) in automated screening systems on the basis of neural network technologies is carried out.

The existing systems of automated screening in the subject areas of medicine, meteorology and remote sensing and features of their functioning by analyzing planar images are analyzed. It is shown that the effectiveness of such systems directly depends on the accuracy of image classification and segmentation, as false-positive results can lead to premature response, and false-negative to the omission of existing problems and late response. In addition, no less impor-

tant factor influencing the effectiveness of screening systems is the time spent on the implementation of the educational process, which, in turn, is associated with the quality of pre-created samples of training data. However, in the tasks of verifying the receipt of sufficient data and conducting high-quality annotations by professionals is usually expensive, and sometimes impossible due to the small number and variety of training examples, subjectivity in the formation of annotations in data sets. These factors negatively affect the accuracy of classification and segmentation in the training of deep neural networks by standard methods. To eliminate such influence, it is proposed to solve the problem taking into account partially erroneous annotations when constructing models of classification and segmentation.

It is shown that in such conditions it is expedient to use deep neural networks, due to the possibility of automating the learning process and the use of multi-tasking learning that increase the accuracy of classification and segmentation.

The analysis of the existing models of datasets and methods of their generation showed the limitations of these models, which would allow to form partially erroneous annotations, which have characteristic of real automated screening tasks.

Thus, solving an important scientific and practical problem of increasing the accuracy of classification and segmentation in the analysis of planar images for automated screening systems by improving models of neural networks and methods of their multitasking, especially in terms of partially erroneously annotated data samples.

In the **second section** method for generating datasets to enable testing of neural networks in learning using partially erroneous annotations and proposed a parametric formalization of the data set model with partially erroneous annotations, which are characteristic of real automated screening tasks were developed.

Because obtaining reliable classification and segmentation annotations for test samples is often impossible or very costly, it is proposed to use model datasets to assess the reliability of models and methods of learning deep neural networks. Such datasets make it possible to conduct training and testing in controlled conditions by artificially adding errors to the annotations of the training data sample (test samples remains with precise annotations).

Proposed model of datasets \mathcal{M} has the following definition:

$$\mathcal{M} \in \{\mathcal{X}_b, \mathcal{X}_{tex}, N, S_{img}, S_{obj}, \Delta_{max}, N_{obj}, P_e, P_d, S_e, S_d\} \quad (9)$$

where \mathcal{X}_b - background images set, \mathcal{X}_f - objects set, \mathcal{X}_{tex} - object texturing set, N - number of images in the generated dataset, S_{img} - pixel size of generated images, S_{obj} - average size of the objects (pixels), Δ_{max} - maximum deviation of objects size, N_{obj} - maximum number of objects in the image, P_e and P_d - probabilities of mask erosion and dilation for every object, S_e and S_d - kernel sizes for erosion and dilation.

The last four parameters are introduced for controlled annotation errors.

Formulated **first point of scientific novelty**: for the first time a parametric formalization of the dataset model with partially erroneous annotations, which are characteristic of real problems of automated screening, was proposed, which allowed the development of a method of generating training, test and validation data sets.

Based on the proposed parametric model (??), a method for generating data sets with partly-erroneous annotations has been developed, which contains the following steps:

1. Choose random background: $x_{bg} \sim \mathcal{X}_b$
2. Choose number of objects in the image: $N_{obj} \sim \mathcal{U}(1, N_{obj})$
3. Initialize the segmentation mask: $M = 0$ and $M \in \mathcal{R}^{C \times S_{img} \times S_{img}}$
4. For n_{obj} times:
 - 4.1 Sample object size: $s \sim \mathcal{U}(S_{obj} - \Delta_{max}, S_{obj} + \Delta_{max})$

4.2 Sample object coordinates:

$$\begin{aligned} i_f &\sim \mathcal{U}(0, S_{img} - s) \\ j_f &\sim \mathcal{U}(0, S_{img} - s) \end{aligned}$$

4.3 Sample object image $x_{fg} \sim \mathcal{X}_f$ and the corresponding object class $c_{fg} \sim \mathcal{Y}_f$

4.4 Resample the image with bilinear interpolation:

$$\hat{x}_{fg} = R_{bilinear}(x_{fg})$$

4.5 Sample the texture image $x_{tex} \sim \mathcal{X}_{tex}$

4.6 Modify object image with the object texture:

$$\hat{x}_{fg} = x_{fg} \circ x_{tex}[i_f : i_f + s, j_f : j_f + s]$$

4.7 Place object image into the background image:

$$x_{bg}[i_f : i_f + s, j_f : j_f + s] = (1 - x_{fg}) \circ x_{bg} + \hat{x}_{fg}$$

4.8 Form the segmentation mask:

$$M_{seg} = x_{fg} > \theta_{seg}$$

where θ_{seg} - is the source object binarization threshold. For the MNIST dataset $\theta_{seg} = 0.2$, and for FashionMNIST $\theta_{seg} = 0.1$.

4.9 Modify the segmentation mask according to the desired error level:

$$M_{seg} = \begin{cases} M_{seg} \oplus K^{S_d \times S_d} & \text{iff } p_d \sim \mathcal{U}(0, 1) < P_d \\ M_{seg} \ominus K^{S_e \times S_e} & \text{iff } p_e \sim \mathcal{U}(0, 1) < P_e \end{cases}$$

where $K^{S_d \times S_d}$ - dilation kernel matrix and $K^{S_e \times S_e}$ - erosion kernel matrix.

4.10 Place modified object mask into the common mask:

$$M[c_{fg}, i_f : i_f + s, j_f : j_f + s] = \max \{M[c_{fg}, i_f : i_f + s, j_f : j_f + s], M_{seg}\}$$

4.11 Save image x_{bg} and mask M

5. Finish generation

Thus, the method of generating datasets with partially erroneous annotations based on the parametric model was improved, which due to the generation of annotations: partially erroneous for the training sample and accurate - for the test, made it possible to test the impact of annotation errors on neural networks.

The method allows to obtain many data sets with similar characteristics and to use non-parametric statistical methods (such as bootstrapping) to evaluate models in the absence of real training data. The method is **the second point of scientific novelty**.

In the **third section** of the dissertation neural network models and methods of multi-tasking learning were developed to simultaneous increase in accuracy of classification and segmentation without decrease in efficiency.

To implement multi-task learning methods, it is proposed to improve the models of deep neural networks using the encoder-decoder architectures (UNet, LinkNet) by introducing an additional decoder with the normalization layer. Thus, the advanced model consists of an encoder and two decoders (for segmentation and classification tasks, respectively).

The model of the deep neural network is represented by the following expression:

$$v_1, v_2 \dots v_n = F_{encoder}(x, \theta_{enc}) \quad (10)$$

$$M_{seg} = F_{seg}((v_1, v_2 \dots v_n), \theta_{seg}) \quad (11)$$

$$C_{cls} = F_{cls}((v_n), \theta_{cls}) \quad (12)$$

where θ_{enc} - are encoder parameters, θ_{seg} and θ_{cls} - parameters of encoder and decoder, respectively, F_{seg} and F_{cls} - decoder neural networks.

The proposed model is **the third point of scientific novelty**.

With the introduction of an additional classification decoder with a normalization layer, it is possible to implement methods of multi-tasking learning and inference for deep neural networks.

During the practical use of neural network methods in screening tasks, it is established that partially erroneous annotations of training data prevent obtaining high accuracy of classification and segmentation.

The method of multi-task learning in the conditions of partially-erroneous annotations of training data, which is based on use of tasks that are connected with original task is offered. It is shown that for the segmentation problem there is a similar classification problem, for which the annotations of training data are of better quality than for the original segmentation problem.

The proposed method of multi-task learning consists of two stages.

Stage 1. Generating the derivative task. In the context of multi-task learning, the task of classification is defined as a multiple instance learning task, instead of marking each of the objects for all classes. In this case, the annotation of the original planar image is a set with one or more objects.

The loss function is calculated separately for each of the tasks (classification and segmentation). The top-limited loss function $L_{seg}]$ is used to train the segmentation decoder, while the classification decoder uses the normal L_{cls} loss function, and the total value of the loss function is defined as the arithmetic mean between the individual values:

$$L_{total} = \frac{L_{seg}] + L_{cls}}{2} \quad (13)$$

Accordingly, the total gradient of the loss function will also be the arithmetic mean of the gradients of the components $\nabla L_{seg}]$ and ∇L_{cls} :

$$\nabla L_{total} = \frac{\nabla L_{seg}] + \nabla L_{cls}}{2} \quad (14)$$

Thus, the presence of non-zero gradients is provided to update the parameters from at least one loss function for each input example.

Stage 2. Loss function bounding. To reduce the impact of the erroneous part of the annotations, it is proposed to introduce a constraint to the loss function for the segmentation problem with lower quality data annotations. That allowed to reduce the influence of gradients of the loss function on examples with erroneous annotations when learning on several tasks:

$$\mathcal{L} = \min(L, \theta)$$

where θ - loss function threshold.

Thus, the introduced boundaries are represented by the following expression:

$$\nabla \min(L, \theta) = \begin{cases} 1, & \text{iff } L \in (-\infty, \theta] \\ 0, & \text{iff } L \in (\theta, \infty) \end{cases}$$

In the absence of markup for examples in segmentation problems, iterative refinement of this markup is possible. To do this, a refined map of classification features is calculated by the Hadamard product between the output of the segmentation decoder normalized by the sigmoid function and the classification logs:

$$M_{unsup} = \hat{M}_{seg} \circ C_{cls} \quad (15)$$

Next, to obtain the classification result, the summation of the elements M_{unsup} is performed with normalization by the sum of the elements of the original non-normalized localization map:

$$C_{unsup} = \frac{\sum_{h=0}^H \sum_{w=0}^W M_{unsup}(h,w)}{\sum_{h=0}^H \sum_{w=0}^W M_{seg}(h,w) + c} \quad (16)$$

Based on the proposed model of the neural network and the method of its training, a method of multi-taski prediction has been developed. This

allowed to increase the accuracy of classification and segmentation of planar images without increasing time needed.

Method consists of the following stages:

Stage 1. Normalization. Let $C_{cls} \in \mathcal{R}^C$ and $M_{seg} \in \mathcal{R}^{C \times H \times W}$ - logits of classification and segmentation decoders, respectively and lie in the interval $(-\infty, +\infty)$.

To obtain normalized to the interval $[0, 1]$, sigmoid activation function is used:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Stage 2. Renormalization. Renormalization is the weighing of a segmentation map using normalized classifier logits. The first step is the transformation of segmentation and classification logs into uncalibrated estimates between $[0, 1]$:

$$\hat{M}_{seg} = \sigma(M_{seg})$$

$$\hat{C}_{cls} = \sigma(C_{cls})$$

These estimates have the same dimensions as original segmentation mask and classification vector, additional dimensions are added for simplicity: $\hat{M}_{seg} \in \mathcal{R}^{C \times H \times W}$ and $\hat{C}_{cls} \in \mathcal{R}^{C \times 1 \times 1}$

Segmentation map is weighted using the Hadamard product: \hat{M}_{seg} and \hat{C}_{cls}

$$M_{refined} = \hat{M}_{seg} \circ \hat{C}_{cls}$$

Using the model, the proposed methods were tested in different conditions, ablation studies were performed and the analysis of the resistance of the proposed method to different levels of errors in the annotations was performed. On average, the increase in the Dice coefficient relative to the base method was 13 %.

Thus, **fourth point of scientific novelty** is formulated, namely, improved methods of multi-task learning and inference based on an improved model of convolutional neural networks by combining classification and segmentation and introducing a constraint when calculating the segmentation loss function, which allowed to increase the accuracy of segmentation and classification in automated screening tasks.

In **fourth section** developed tools that implement the proposed solutions. The developed method was tested as part of experiments on synthetic data generated using the proposed model, as well as in experiments in real problems: screening of diabetic retinopathy, melanoma screening, and screening of cloud formations.

The tools are developed in the Python programming language using the PyTorch automatic differentiation framework. Based on the developed tools, effective software modules have been created, which are integrated with cloud services for solving resource-intensive learning tasks of neural networks, which provides high computing power and speed of prediction in automated screening tasks.

For the task of automated screening in the cloud organization patterns segmentation in satellite images (within the project "Understanding Clouds from Satellite Images" on the platform for competitions in data science Kaggle) the proposed models of neural networks were used, as well as methods for training and inference. The increase in Dice measure relative to the base model was 3.9%.

For the task of automated screening in the diabetic retinopathy classification (within the project "APTOS 2019 Blindness Detection" on the platform for competitions in data science Kaggle) used the proposed methods of multi-tasking and inference. The increase in F1-measure relative to the base model was 2.1 %.

For the task of recognizing skin lesions in melanoma screening (within

the SIIM-ISIC Melanoma Classification project on the Kaggle Data Science Competition Platform), the proposed localization of image-relevant image features was used. The use of the proposed method simplified the process of monitoring the learning of neural networks, which helped prevent retraining and increase the reliability of the classification by 3.5%.

The methods and tools developed in the work were introduced into the educational process of ONPU and the SafetyRadar software product of VITech Lab, the main purpose of which is to screen the presence of elements of personal protective equipment on people in construction sites, or hospitals and laboratories.

Key words: image analysis, dataset models, deep neural networks, multi-tasking machine learning, segmentation, classification, loss functions.

ЗМІСТ

Вступ	28
Розділ 1. Проблеми аналізу планарних зображень в задачах автоматизованого скринінгу	34
1.1. Аналіз задачі автоматизованого скринінгу планарних зображень	34
1.1.1. Задачі автоматизованого скринінгу в медицині	35
1.1.2. Задачі автоматизованого скринінгу в метеорології	39
1.1.3. Структура інформаційних систем автоматизованого скринінгу	41
1.2. Моделі глибинного машинного навчання	41
1.2.1. Згорткові нейронні мережі	41
1.2.2. Визначні моделі ШНМ для класифікації зображень	43
1.2.3. Визначні моделі ШНМ для сегментації зображень	52
1.2.4. Тестування моделей в задачах автоматизованого скринінгу	58
1.3. Методи багатозадачного глибинного машинного навчання	62
1.3.1. Визначення багатозадачного машинного навчання	63
1.3.2. Багатозадачне навчання в задачах класифікації та сегментації	68
1.4. Висновки до першого розділу	72
Розділ 2. Параметрична модель наборів даних із частково-помилковими анотаціями та метод їх генерації	74
2.1. Проблема частково-помилкових анотацій в наборах даних	74
2.1.1. Природні причини помилок в анотаціях в наборах даних	74

2.1.2. Введення помилок в анотації наборів даних	76
2.2. Параметрична модель набору даних із частково-помилковими анотаціями	78
2.3. Метод генерації наборів даних	80
2.4. Види згенерованих зображень	84
2.5. Висновки до другого розділу	90
Розділ 3. Нейромережева модель аналізу планарних зображень та методи її навчання	91
3.1. Структурна модель багатозадачної штучної нейронної мережі	91
3.1.1. Загальна структура моделі нейронної мережі	92
3.1.2. Структура декодера семантичної сегментації	93
3.1.3. Структура декодера класифікації	94
3.2. Методи навчання та прогнозування багатозадачних ШНМ в умовах частково помилкової розмітки сегментації	96
3.2.1. Генерація похідної задачі до задачі семантичної сегментації	97
3.2.2. Автоматична фільтрація помилкової розмітки	98
3.2.3. Багатозадачне прогнозування результатів семантичної сегментації	102
3.2.4. Локалізація важливих для класифікації ознак в умовах відсутності розмітки для сегментації в навчально-му наборі даних	105
3.3. Висновки до третього розділу	108
Розділ 4. Побудова та випробування системи автоматизованого скринінгу	110
4.1. Інструментальні засоби	110
4.2. Експерименти на синтетичних даних	112
4.2.1. Оцінка прогнозів нейронної мережі	112

4.2.2. Структура нейронних мереж	113
4.2.3. Параметри експериментів	114
4.3. Експерименти на реальних даних	128
4.4. Класифікація та сегментація формаций хмар	128
4.4.1. Опис набору даних Understanding Clouds from Satellite Images	128
4.4.2. Процедура навчання	131
4.4.3. Результати експерименту	132
4.4.4. Класифікація стадії осередків діабетичної ретинопатії	132
4.4.5. Процедура навчання	134
4.4.6. Результати експерименту	136
4.5. Класифікація раку шкіри та локалізація родимок	136
4.5.1. Опис набору даних SIIM-ISIC Melanoma Classification	136
4.5.2. Процедура навчання	138
4.5.3. Результати експерименту	139
4.6. Висновки до четвертого розділу	140
Висновки	142
Список використаних джерел	145
Додаток А. Список публікацій здобувача за темою дисертації . .	157
Додаток Б. Акт впровадження результатів дисертації в компанії VITech Lab	160
Додаток В. Акт впровадження результатів дисертації в навчаль- ний процес ОНПУ	161
Додаток Г. Акт впровадження результатів компанії ПЛАНЕТА ЮГ	162
Додаток Д. Вихідні коди програмних інструментальних засобів .	163

ВСТУП

Актуальність теми зумовлюється стрімким розвитком процесів, які потребують швидкого реагування на нетипові зміни для їх подальшого коригування, поки ціна наслідків є низькою. Задача виявлення таких нетипових змін називається задачею скринінгу, особливу групу серед яких складають засновані на аналізі планарних зображень задачі, які виникають, наприклад, при медичній діагностиці, дистанційному зондуванні Землі, метеорології тощо. Автоматизація скринінгу дозволяє з більшою ефективністю вирішувати проблеми підвищення продуктивності діяльності людини-оператора, підвищити рентабельність відповідних процесів.

При автоматизації скринінгу основними процедурами, що визначають його достовірність є сегментація регіонів інтересу та їх класифікація. Традиційно, в таких системах використовуються класичні методи комп'ютерного зору (пошук границь, бінарна сегментація за інтенсивністю, кореляційно-екстремальний метод, тощо), однак достовірність їх суттєво обмежена через різноманіття можливих спотворень (наприклад, різний вигляд об'єктів одного класу) та складність їх формалізації за допомогою наборів простих ознак. В таких умовах доцільне використання семантичної сегментації (*класифікаційного методу сегментації*), для реалізації якої в теперішній час використовуються глибинні нейронні мережі, завдяки можливості вирішення задач в умовах невизначеності та стійкості до вхідних спотворень з високою достовірністю. Безумовною перевагою застосування глибинних нейронних мереж є можливість використання багатозадачного навчання для підвищення достовірності та оперативності, але його застосування потребує наявності великих наборів достовірно анотованих (роздічених) даних.

Але, в задачах скринінгу, отримання достатньої кількості даних та проведення якісного анатування професіоналами є, як правило, затратним, а іноді неможливим через малу кількість та різний вигляд навчальних прикладів, суб'єктивність під час формування анотацій в наборах даних.

Таким чином, має місце **протиріччя** між широкими можливостями щодо забезпечення достовірності рішення задачі автоматизованого скринінгу нейромережевими методами з одного боку, та обмеженістю існуючих методів отримання наборів анатованих даних для навчання глибинних нейронних мереж.

Для розв'язання цього протиріччя в дисертаційній роботі “Нейромережеві методи аналізу планарних зображень в системах автоматизованого скринінгу” запропоновані відповідні нейромережеві моделі та методи аналізу планарних зображень в задачах автоматизованого скринінгу для підвищення його достовірності.

Робота пов'язана з дослідними проектами:

Робота виконана відповідно до планів наукової і науково-технічної діяльності Одеського Національного Порлітехнічного Університету в рамках держбюджетних науково-дослідних робіт: «Моделі, методи та інструментальні засоби підтримки прийняття рішень з підвищення ефективності гідроаеродинамічних процесів в діючому енергетичному обладнанні» (№ ДР 0115U000413); «Дослідження інформаційних сховищ як моделей предметних областей в системах підтримки прийняття рішень» (№ ДР 0104U002401); «Методи моделювання та робочого діагностування складних цифрових систем і мереж» (номер ДР 0110U008194)

Метою дослідження є підвищення достовірності класифікації та сегментації планарних зображень в системах автоматизованого скринінгу шляхом розробки нейромережевих моделей і методів.

Для досягнення мети дослідження необхідно вирішити такі завдання:

— провести аналіз проблем автоматизованого скринінгу, показати пе-

- реваги та недоліки нейромережевого підходу, та обґрунтувати напрямки досліджень;
- розробити параметричну модель набору даних, зокрема з частково-помилковими анотаціями, які характерні для реальних задач автоматизованого скринінгу;
 - розробити модель нейронної мережі та методи навчання й передбачення для аналізу планарних зображень;
 - розробити інструментальні засоби, які реалізують розроблені нейромережеві моделі та методи, та виконати апробацію і впровадження теоретичних результатів при вирішенні завдань автоматизованого скринінгу.

Об'єктом дослідження є процеси аналізу планарних зображень в системах автоматизованого скринінгу.

Предмет дослідження – нейромережеві моделі та методи класифікації і сегментації планарних зображень в задачах автоматизованого скринінгу.

Методи дослідження. Для вирішення поставлених задач використані методи системного аналізу та теорії множин при розробці нейромережевих моделей; теорія нейронних мереж при розробці відповідних нейромережевих моделей та методів; теорія цифрової обробки зображень та розпізнавання при розробці методів класифікації і сегментації; методи імітаційного моделювання при апробації запропонованих рішень.

Наукова новизна отриманих результатів. Основний науковий результат полягає у розв'язанні важливої науково-практичної задачі підвищення достовірності класифікації та сегментації планарних зображень в системах автоматизованого скринінгу шляхом розробки нейромережевих моделей і методів.

У рамках виконаних досліджень отримано такі наукові результати:

- *вперше* запропоновано параметричну формалізацію моделі набору

- даних із частково-помилковими анотаціями, які характерні для реальних задач автоматизованого скринінгу, що дозволило розробити метод генерації навчальних, тестових та валідаційних наборів даних;
- *удосконалено* метод генерації наборів даних на основі параметричної моделі, що за рахунок генерації анотацій: частково-помилкових для тренувальної вибірки та достовірних - для тестової, що дало можливість підвищити ефективність тестування нейромережевих методів сегментації та класифікації в задачах автоматизованого скринінгу;
 - *удосконалено* моделі згорткових нейронних мереж шляхом додавання додаткового декодера класифікації з шаром нормалізації, що дало змогу побудувати методи багатозадачного навчання та передбачення результатів нейронних мереж, які вирішують задачі сегментації та класифікації одночасно;
 - *удосконалено* методи багатозадачного навчання та передбачення на основі *удосконаленої* моделі згорткових нейронних мереж шляхом об'єднання класифікації та сегментації і введення обмеження другого роду (зверху) при обчисленні функції втрат сегментації, що дозволило підвищити достовірність сегментації та класифікації в задачах автоматизованого скринінгу.

Практичне значення одержаних результатів. Практичне значення розроблених моделей та методів підтверджується їх використанням в розроблених інструментальних засобах для вирішення задач автоматизованого скринінгу. Результати дисертаційної роботи впроваджено: в програмний продукт SafetyRadar компанії VITech Lab (акт впровадження від 13.09.2021), програмні продукти компанії «ПЛАНЕТА ЮГ» (акт впровадження від 15.09.2021) та в навчальний процес кафедри Інформаційних систем Інституту Комп'ютерних систем **Одеського Національного Політехнічного Університету**.

хнічного Університету (акт впровадження від).

На основі розроблених інструментальних засобів створено ефективні програмні модулі, які інтегровано з “хмарними” сервісами для вирішення ресурсомістких задач навчання нейронних мереж, що підвищило оперативність навчання та передбачення в задачах автоматизованого скринінгу на 15%. Впровадження запропонованих в дисертаційній роботі моделей та методів забезпечує підвищення достовірності сегментації та класифікації планарних зображень в задачах автоматизованого скринінгу та скорочує витрати на одержання анотованих наборів даних.

Особистий внесок здобувача

Основні наукові положення, висновки і рекомендації, що містяться в дисертації та виносяться на захист, отримано здобувачем особисто в період з 2016 по 2020 рр. Наукова праця [1] опублікована без співавторів. У статтях, написаних у співавторстві, внесок здобувача полягає в наступному: [2-4] – запропоновано модель нейронної мережі та метод багатозадачного навчання, [5] – запропоновано модель нейронної мережі та метод підбору порогу класифікації, [6-8] – модель нейронної мережі та метод еволюції параметрів, [9] - алгоритм детекції полоси для лінійної камери; [11] – метод ідентифікації параметрів ковзання; [12-13] – метод розпізнавання спектрів.

Апробація результатів дисертації

Основні положення і практичні результати дисертаційної роботи доповідалися та одержали схвалення на таких конференціях: міжнародній науковій конференції з методів розпізнавання патернів “ICPRAM-2020” (м. Валетта, Мальта, 2020), українсько-німецьких науково-практичних конференціях “Інформатика. Культура. Техніка”: ІКТ-2016, ІКТ-2018 (м. Одеса), міжнародних студентських конференціях “Сучасні інформаційні технології”: СІТ-2017, СІТ-2018 (м. Одеса).

Публікації

Основні результати дисертаційної роботи викладено в 13 публікаціях, з

них: 8 статей у наукових фахових періодичних виданнях України з технічних наук, 1 стаття у зарубіжному науковоуму періодичному виданні з напряму, з якого підготовлено дисертацію, що індексується наукометричною базою SCOPUS, 4 публікацій у працях і матеріалах наукових конференцій.

Структура і обсяг роботи

Дисертація складається з вступу, чотирьох розділів, висновків, списку використаних джерел і додатків. Повний обсяг дисертації складає 184 сторінки, в тому числі: 134 сторінок основного тексту, 55 рисунків і 17 таблиць, список використаних джерел з 116 найменувань і 5 додатків.

РОЗДІЛ 1

ПРОБЛЕМИ АНАЛІЗУ ПЛАНАРНИХ ЗОБРАЖЕНЬ В ЗАДАЧАХ АВТОМАТИЗОВАНОГО СКРИНІНГУ

1.1. Аналіз задачі автоматизованого скринінгу планарних зображень

Задача автоматизованого скринінгу постає, коли виникає необхідність ідентифікації процесів, які потребують швидкого реагування на нетипові зміни для їх подальшого коригування, поки ціна наслідків є низькою. Задача виявлення таких нетипових змін називається задачею скринінгу, особливою групу серед яких складають засновані на аналізі планарних зображень задачі, які виникають, наприклад, при медичній діагностиці, дистанційному зондуванні Землі, метеорології тощо. Автоматизація скринінгу дозволяє з більшою ефективністю вирішувати проблеми підвищення продуктивності діяльності людини-оператора, підвищити рентабельність відповідних процесів.

Попри те, що скринінг сприяє ранньому реагуванню, існує можливість помилкової діагностики, а також створення неправдивого почуття упевненості у відсутності проблеми. З цих причин скринінгові дослідження повинні мати достатню чутливість і допустимий рівень специфічності [1]. Також, в задачі автоматизованого скринінгу важливою є локалізація ознак можливих проблем, для подальшого оцінювання людьми. [2]

Характерним для задачі скринінгу є клас планарних зображень, особливістю якого є постійний масштаб об'єктів, та можливість знектувати перспективними спотвореннями [3].

Одними з найбільш важливих прикладних областей, в яких використо-

вується автоматизований скринінг, є медицина та метеорологія, що розглянуті далі.

1.1.1. Задачі автоматизованого скринінгу в медицині. Серед багатьох застосувань скринінгу в медицині, особливо нагальною є потреба в методах скринінгу для агресивних прихованих хвороб в регіонах, де в більшості населення немає доступу до професійного лікаря. Так, агресивними хворобами, для яких утруднене діагностування на ранніх стадіях є, наприклад діабетична ретинопатія та рак шкіри.

Скринінг раку шкіри. Рак шкіри є найбільш розповсюдженим видом злоякісних пухлин, і саме меланома є причиною більшості смертей від раку. Світова проблема захворюваності на меланому стрімко зростала за останні 50 років і стала проблемою, з якою намагаються боротися багато вчених з різних країн.

Меланома є п'ятим за поширеністю раком серед чоловіків та шостим за поширеністю раком серед жінок [4]. Подібно до інших типів раку, ранні та легкі стадії візуально навряд чи можна розрізнати. В наш час, дерматологи оцінюють кожну родимку пацієнта, щоб виявити незвичні осередки або такі, що, швидше за все, є злоякісними. Якщо меланому помітити вчасно, її можна вилікувати незначними оперативними втручаннями.

Огляд літератури Недавні дослідження в галузі автоматичного виявлення злоякісних утворень пов'язані з найсучаснішими підходами до глибокого навчання в розпізнаванні зображень. Набагато менше робіт використовують класичне машинне навчання та сконструйовані дослідниками ознаки.

Тут наведено найвпливовіші роботи в цій галузі. Так, Mustafa та ін. [5] створив підхід з підібраними вручну ознаками (GrabCut для сегментації раку) та методом опорних векторів для класифікації ракових уражень. Також, Nasiri et al. [8] згенерував похідні зображення за допомогою декількох

алгоритмів і використав на них метод k-найближчих моделей сусідів для вирішення завдання.

Брінкер та ін. [6] проводив експерименти із попередньо навченими на наборі даних ImageNet згортковими нейронними мережами, такими як ResNet-50 [7], для класифікації ранніх стадій меланоми. Автори використали 4204 перевірених біопсією зображення меланоми та звичайних родинок. Крім того, були інтегровані новітні на той момент методи глибокого навчання: різні темпи навчання для різних частин нейронної мережі, зменшення темпу навчання на основі функції косинуса, стохастичний градієнтний спуск з перезапуском для того щоб уникнути локальних мінімумів.

Коделла та ін. [8] запропонували систему сегментації та класифікації меланоми за дермоскопічними зображеннями шкіри. Для класифікації хвороб вони застосували ансамбль останніх методів машинного навчання, включаючи глибокі залишкові мережі, згорткові нейронні мережі тощо. Вони довели, що ансамблі здатні давати кращі результати, ніж моделі окремо.

Насірі та ін. [9] запропонували класифікацію уражень шкіри за допомогою глибокого навчання для раннього виявлення меланоми в системі міркувань на основі прецедентів (англ. case-based reasoning). Цей підхід був використаний для отримання схожих вхідних зображень із бази даних прецедентів запропонованої системи DePicT Melanoma Deep-CLASS для підвищення точності рекомендацій щодо запитуваної проблеми (наприклад, зображення родинки). Їх метод, що заснований на глибоких згорткових нейронних мережах, генерує ознаки з зображень, щоб використовувати їх у процесі пошуку в базі даних. Інтеграція цього підходу до DePicT Melanoma CLASS значно покращила ефективність класифікації зображень та якість рекомендаційної частини системи.

Дослідження в галузі багатозадачного навчання також проводили Сонг та ін. [10]. Вони запропонували нейронну мережу, яка може одночасно виконувати завдання детекції, класифікації та сегментації уражень шкіри, не

вимагаючи додаткових етапів попередньої обробки або подальшої обробки. Подібну роботу представили Чен та співавтори [11], вони використали багатозадачну мережу U-Net для задачі детекції та сегментації. Янг та ін. [12] запропонував більш складну багатозадачну модель, яка одночасно вирішує завдання сегментації уражень та дві незалежні задачі бінарної класифікації, використовуючи спільноті та відмінності між завданнями.

Скринінг діабетичної ретинопатії. Діабетична ретинопатія є одним із найбільш загрозливих ускладнень діабету, при якому пошкодження сітківки викликає сліпоту. Вона пошкоджує кровоносні судини тканини сітківки, викликаючи витік рідини та погіршення зору. Поряд із захворюваннями, що призводять до сліпоти, такими як катаракта і глаукома, ретинопатія є одним із найпоширеніших захворювань, згідно зі статистикою США, Великобританії та Сінгапуру.

Лікарі встановили чотири стадії діабетичної ретинопатії:

- Легка непроліферативна ретинопатія, найраніша стадія, коли можуть виникати лише мікроаневризми;
- Помірна непроліферативна ретинопатія, стадію якої можна описати втратою здатності кровоносних судин до транспортування крові через набряк з прогресуванням захворювання;
- Важка непроліферативна ретинопатія призводить до обмеженого кровопостачання сітківки через підвищений набряк великої кількості кровоносних судин;
- Проліферативна діабетична ретинопатія - це запущена стадія, коли фактори росту, що виділяються сітківкою, активізують проліферацію нових кровоносних судин, зростаючи вздовж оболонки сітківки в склоподібному тілі, заповнюючи око.

Щонайменше 56% нових випадків можна уникнути за допомогою належного та своєчасного лікування та скринінгу очей. Однак, початкова

стадія цього захворювання не має помітних для пацієнта ознак, і виявити його на ранній стадії є справжньою проблемою. Більш того, добре навчені діагности іноді не можуть вручну оцінити стадію за діагностичними зображеннями очного дна пацієнта.

Огляд літератури

Багато дослідницьких зусиль було присвячено проблемі раннього виявлення діабетичної ретинопатії. Перш за все, дослідники намагалися використовувати класичні методи комп’ютерного зору та машинного навчання, щоб забезпечити відповідне рішення цієї проблеми.

Наприклад, Прія та ін. [13] запропонували підхід на основі комп’ютерного зору для виявлення діабетичної ретинопатії за допомогою кольорових зображень очного дна. Автори створили набір ознак із вихідного зображення, використовуючи класичні методи обробки зображень, і використали метод опорних векторів для бінарної класифікації. Їх метод досяг чутливості 98%, специфічності 96% та точності 97% на тестовому наборі з 250 зображень.

Крім того, інші дослідники намагалися використати інші моделі для багатокласової класифікації, наприклад, застосовуючи аналіз головних компонент до зображень та використовуючи дерево рішень, Баєсові класифікатори або метод найближчих сусідів [14] з найкращими результатами 73.4% точності та 68.4% для F-міри, використовуючи набір даних із 151 зображення з різною роздільною здатністю.

Зі зростанням популярності підходів, заснованих на глибокому навчанні, з’явилися методи, які застосовують глибокі згорткові нейронні мережі до цієї проблеми. Пратт та ін. [15] розробили архітектуру нейронної мережі та використали аугментацію даних, яка може ідентифікувати складні ознаки захворювання, пов’язані із задачею класифікації, такі як мікроаневризми, ексудат та крововиливи в сітківку ока, і, отже, автоматично діагностувати стадію захворювання. Цей метод досяг чутливості 95% і точності 75% на 5000 валідаційних зображень. Крім того, є й інші роботи про

використання глибоких штучних нейронних мереж від інших дослідників [16, 17].

Asiri та ін. провели аналіз значної кількості доступних методів та наборів даних, висвітливши їх плюси та мінуси [18]. Крім того, автори вказали на проблеми, які слід вирішити при розробці та вивчені ефективних та надійних алгоритмів глибокого навчання для різних проблем діагностики діабетичної ретинопатії, та звернули увагу на напрямки подальших досліджень.

Інші дослідники також намагалися здійснити трансферне навчання за допомогою згорткових нейронних мереж. Хагош та ін. [19] спробував навчити InceptionNet V3 для класифікації 5 класів з нейронною мережею, натренованою на наборі даних ImageNet і досяг точності 90.9%. Сарки та ін. [20] провів дослідження з навчання різних архітектур, зокрема ResNet50, Xception, DenseNets та VGG за допомогою попереднього навчання на наборі даних ImageNet і досяг найкращої точності 81.3%. Обидві групи дослідників використовували набори даних, які надавали APTOS та Kaggle.

1.1.2. Задачі автоматизованого скринінгу в метеорології. Однією з найцінніших особливостей визначення кліматичної моделі Землі є поведінка хмар. Однак дослідження їх поведінки є однією з найскладніших частин, оскільки вимагає досконалого розуміння всіх процесів в атмосфері. Класифікація різних типів організації хмар допомагає покращити розуміння цих хмар, що, в свою чергу, допоможе нам побудувати кращі кліматичні моделі.

Дослідники з Інституту метеорології імені Макса Планка зібрали найбільший набір даних, що складається з приблизно 10 000 фотографій хмар в видимому спектрі з супутників Terra і Aqua MODIS. Завдяки краудсорсинговій спільноті Zooniverse, вони створили анотований набір даних, в якому хмари позначені чотирма типами масок: цукор, квітка, гравій, ри-

ба (англ. Sugar, Flower, Gravel, Fish). Однак, через недосконалу процедуру маркування фотографій, та через те, що маркування було зроблене не-професіоналами, розмітка є частково помилковою. Так, маски сегментації класів містять багато пікселів, що належать до фону, та деякі хмари не мають масок сегментації.

Огляд літератури Щоб описати широкий спектр дослідницьких робіт, розглянуто роботи, що використовують як супутникові зображення в видимому спектрі, так і в інших діапазонах: мультиспектральні та інфрачервоні.

Один із методів виявлення хмар був розроблений Чжу та співавтори [21], вони запропонували метод під назвою Fmask (функція маски) для виявлення хмар та їхніх тіней на зображеннях із супутника Landsat 7. Fmask використовує підходи, засновані на правилах на основі фізичних властивостей хмар, щоб відокремити потенційні хмарні регіони від чистого неба. В якості вхідних даних метод використовує інформацію із семидіапазонних датчиків Enhanced Thematic Mapper (ETM) та Enhanced Thematic Mapper Plus (ETM+), якими обладнаний супутник Landsat. Альтернативний підхід був запропонований Харб та ін. [22], автори проаналізували мультиспектральні дані середньої роздільної здатності з супутників програми CBERS. Даний алгоритм використовує набір математичних операцій над спектральними смугами, щоб покращити видимість хмар та їхніх тіней.

Ці методи дають точні результати, але в значній мірі залежать від моделей датчиків (оскільки вони є методами, заснованими на правилах), а пропоновані рішення не є масштабованими до інших типів датчиків.

Ху та співавтори [23] представили більш загальне рішення, використовуючи методи комп'ютерного зору для виявлення декількох низькорівневих ознак, таких як колір, особливості текстури тощо. Для оцінки піксельних масок автори використали класичні алгоритми машинного навчання. Озкан та ін. [24] застосували глибокі нейронні мережі (такі як

Feature Pyramid Network) для сегментації хмар із низькоорбітальних RGB-зображень супутників Gokturk-2 та RASAT.

1.1.3. Структура інформаційних систем автоматизованого скринінгу. Інформаційні системи автоматизованого скринінгу ...

1.2. Моделі глибинного машинного навчання

Глибинне машинне навчання - це частина більш широкого сімейства моделей машинного навчання, заснованих на штучних нейронних мережах (ШНМ). Методи глибинного машинного навчання моделюють високорівневі абстракції за допомогою графу з декількох шарів, що побудовані з лінійних, чи нелінійних перетворень.

Штучні нейронні мережі — це обчислювальні системи, натхнені біологічними нейронними мережами. ШНМ ґрунтуються на сукупності з'єднаних вузлів, що називають штучними нейронами (аналогічно до біологічних нейронів у головному мозку тварин). Кожне з'єднання (аналогічне синапсу) між штучними нейронами може передавати сигнал від одного до іншого. Штучний нейрон, що отримує сигнал, може обробляти його, й потім сигналізувати іншим нейронам, приєднаним до нього [25].

1.2.1. Згорткові нейронні мережі. Революція в розпізнавання обrazів була зроблена за допомогою згорткових нейронних мереж (ЗНМ). Раніше, для задач розпізнавання використовувалися фільтри, які обиралися вручну, а після них використовувався простий класифікатор. Велика перевага нейронних мереж, полягає в тому, що потрібні лише тренувальні дані. На основі даних, фільтри і класифікатори навчаються автоматично. Це стало особливо потужним методом в завданнях розпізнавання зображень.

Дані з зображень захоплюється за допомогою операції згортки. Використовуючи згорткові ядра для сканування цілого зображення, потрібно

вивчити порівняно небагато параметрів відносно повнозв'язних штучних нейронних мереж. Окрім безпосередньо операції згортки, сучасні ЗНМ використовують нелінійні функції активації, операції підвибірки та нормалізації.

Активації. Функції активації забезпечують нелінійність нейронної мережі, виконуючи нелінійне перетворення карти ознак, отриманої від згорткового, або повнозв'язного шару.

Найбільш відомими функціями активації в даний час є [26]:

- Сигмоподібна (Sigmoid) [27]
- Гіперболічний тангенс (Tanh) [28]
- Лінійний ректифікатор (ReLU) [29]
- Нещільний лінійний ректифікатор (LeakyReLU) [30]
- SoftPlus [31]
- Доповнена тотожна функція (Bent identity) [32]
- Swish [33]
- Mish [34]

Графіки зазначених функцій зображені на рис. 1.1.

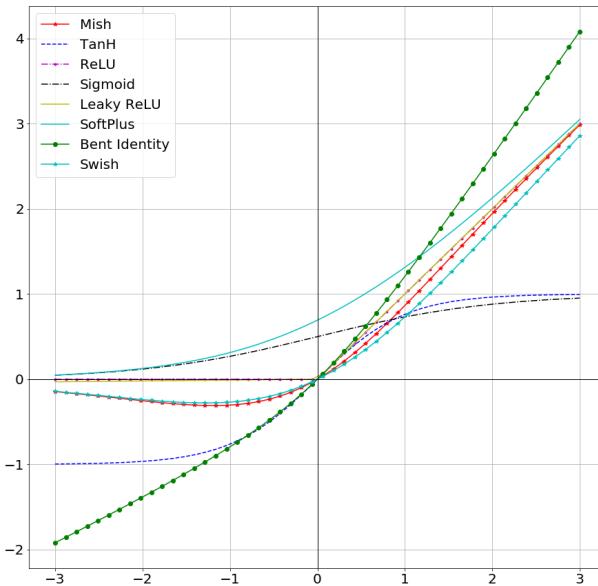


Рис. 1.1: Графіки популярних функцій активації [34]

Підвибірки. Шари підвибірки використовуються для зменшення просторової роздільної здатності та агрегації просторових ознак. В сучасних нейронних мережах використовуються шари підвибірки з функцією максимуму та середнього арифметичного. Також існують шари підвибірки з іншими функціями, але вони не набули загального застосування [35, 36, 37].

Нормалізації. В сучасних нейронних мережах, шари нормалізації використовуються для стабілізації процесу навчання. Розподіл входів кожного шару змінюється під час навчання, оскільки змінюються параметри попередніх шарів. Це уповільнює навчання, вимагаючи нижчих темпів навчання та ретельної ініціалізації параметрів.

Нормалізація проводиться над картами ознак, або їх частинами, приводячи їх до розподілу з середнім значенням 0 та стандартним відхиленням 1. Різні типи нормалізацій які застосовуються в нейронних мережах включають:

- Пакетна нормалізація (batch normalization)[38]
- Пошарова нормалізація (layer normalization)[39]
- Поекземплярна нормалізація (instance normalization) [40]
- Групова нормалізація (group normalization) [41]

Візуалізація різних типів нормалізації зображена на рис. 1.2. Тут N - кількість карт ознак в пакеті, C - розмірність каналів карти ознак, H, W - просторова розмірність одного каналу. Синім виділено набори ознак, на яких обчислюється середнє значення та стандартне відхилення.

1.2.2. Визначні моделі ШНМ для класифікації зображень.

Модель нейронної мережі AlexNet. AlexNet є однією з найперших вдалих спроб застосування штучних згорткових нейронних мереж до зображень у великому масштабі. На відміну від більш ранніх робіт, автори досягли великого прориву на великому наборі даних ImageNet [42].

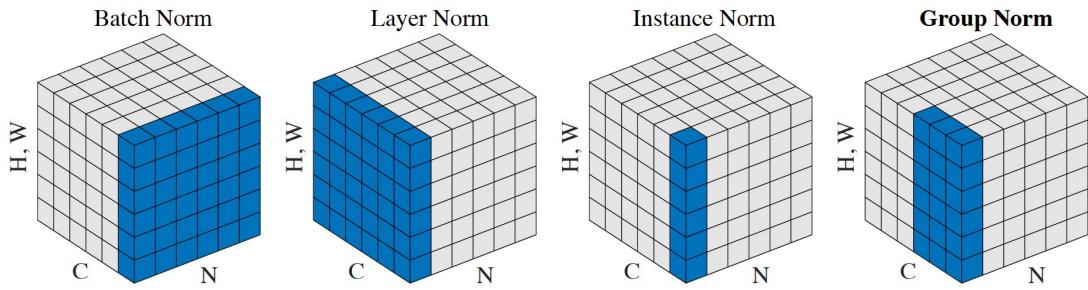


Рис. 1.2: Візуалізація різних типів нормалізації [41]

Архітектура нейронної мережі є логічним продовженням розробок Я. Лекуна [43] для розпізнавання рукописних літер за допомогою згорткових нейронних мереж. AlexNet складається з восьми шарів, перші п'ять з яких - згорткові, а останні три - повнозв'язними шарами. AlexNet використовує функцію активації ReLU, яка демонструвала покращені результати навчання в порівнянні з сигмоподібною. Для збільшення поля зору, автори використали великий розмір ядра в перших згорткових шарах (11 та 5 пікселів відповідно)

Для того, щоб мати можливість навчання не великої кількості зображень порівняно високої роздільної здатності, автори розробили алгоритм паралельного навчання на декількох графічних процесорах (GPU).

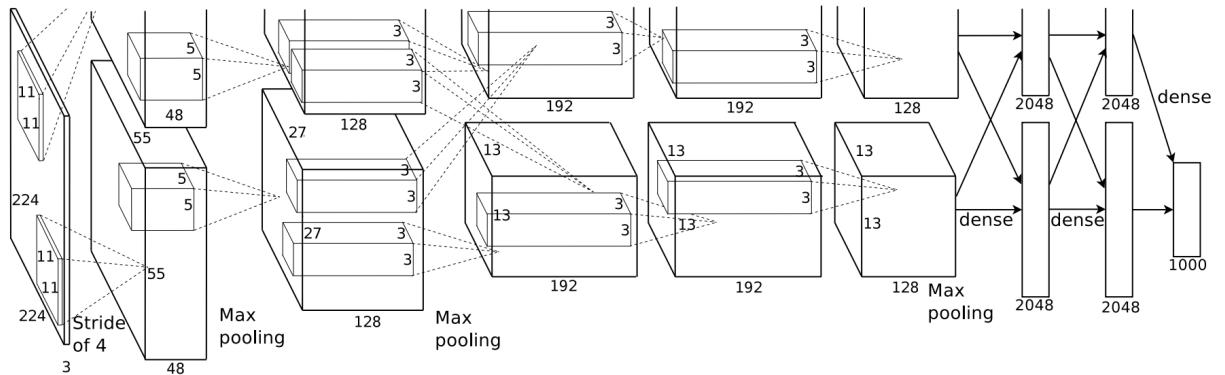


Рис. 1.3: Графічна репрезентація моделі нейронної мережі AlexNet [44]

Також, автори вперше показали можливість використання карт ознак як міри подібності для вхідних зображень та необхідність регуляризації

для запобігання перенавчанню.

AlexNet вважається однією з найвпливовіших статей, опублікованих в галузі комп'ютерного зору, завдяки чому було опубліковано багато інших публікацій із використанням CNN та GPU для прискорення глибокого навчання.

Модель нейронної мережі VGGNet. VGGNet є інкрементальним покращенням архітектури AlexNet, в якому адресовано проблему продуктивності та систематизовано залежність точності від глибини нейронної мережі. Покращення продуктивності в порівнянні з AlexNet досягається заміною великих розмірів ядра згортки (11 і 5 у першому та другому згортковому шарі, відповідно) кількома згортками з розміром ядра 3x3 одна за одною. Також, за рахунок використання більшої кількості нелінійних активацій, підвищується дикримінаційна здатність нейронної мережі. Так, три згортки з ядром 3x3 одна за одною та кроком 1 мають той самий розмір поля зору що й одна згортка з розміром ядра 7x7, але кількість задіяних параметрів становить $3 \cdot (3^2 C^2)$ порівняно з $7^2 C^2$ параметрів для ядер розміром 7x7. Тут C - це кількість вхідних та вихідних каналів згорткового шару.

Автори порівнюють декілька нейронних мереж, які відрізняються лише кількістю шарів. Вихідна таблиця порівняння архітектур зазначена на рис. 1.4.

В даній роботі було продемонстровано, що глибина нейронної мережі вигідна для точності класифікації, а також того, що можна досягти високої точності за допомогою звичайної архітектури згорткової нейронної мережі.

Сімейство моделей нейронних мереж Inception. Мережа Inception [46] стала важливою віхою в розвитку згорткових нейронних мереж. До цієї роботи, згорткові нейронні мережі являли собою послідовні операції згортки, активації та підвибірки. До сімейства Inception входять нейронні

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Рис. 1.4: Порівняння моделей VGGNet з різною кількістю шарів [45]

мережі, що були спеціально розроблені для підвищення точності та продуктивності.

Автори розглядають проблему варіативності розміру об'єктів на зображеннях: вибір правильного розміру ядра для операції згортки стає важким. Більше ядро краще підходить для виділення інформації, яка розповсюджена більш глобально, а менше ядро - для інформації, яка розповсюджується локально.

Для вирішення цієї проблеми, автори використовують декілька операцій згортки з різними розмірами ядра в одному шарі нейронної мережі. Оскільки виконання згорток з великою кількістю каналів є довгою операцією, автори зменшують кількість каналів за допомогою згорток з розміром ядра 1×1 . Один такий шар називається *блоком Inception*, архітектура якого зображена на рис. 1.5

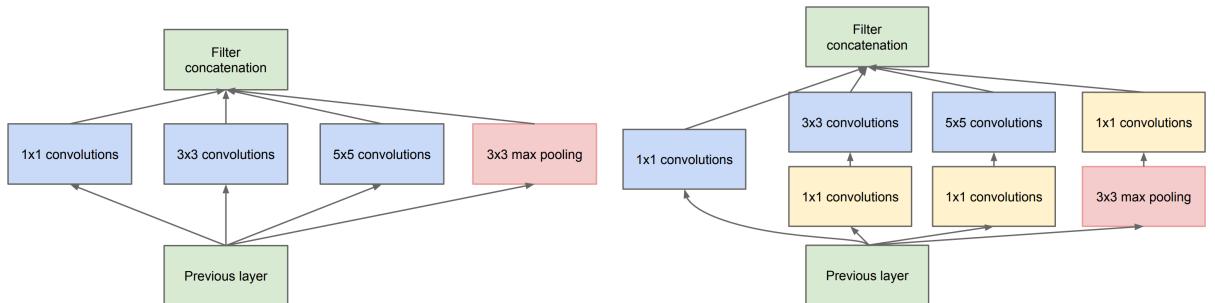


Рис. 1.5: Блок-схема Inception v1 до та після розкладу[46]

Нейронна мережа GoogLeNet [47] складалася з 22 шарів в глибину та була найглибшою на той час. Для того, щоб запобігти затуханню градієнтів, використовувалися додаткові класифікатори на менш глибоких рівнях, результати яких було додано до функції втрат.

В наступних роботах [48], автори продовжують ідею зменшення кількості операцій, для чого вводять розкладання згорток з великим розміром ядра на декілька менших згорток. Так, автори пропонують розкладання згортки з ядром розміру 5×5 на дві згортки з розміром ядра 3×3 , а згортки

з розміром ядра 3×3 на дві згортки розміру 1×3 та 3×1 відповідно. Це дозволяє зменшити кількість параметрів в нейронній мережі при збереженні глибини, зменшуючи необхідні ресурси для обчислення.

Схема блоків Inception v2 та v3 зображене на рис. 1.6

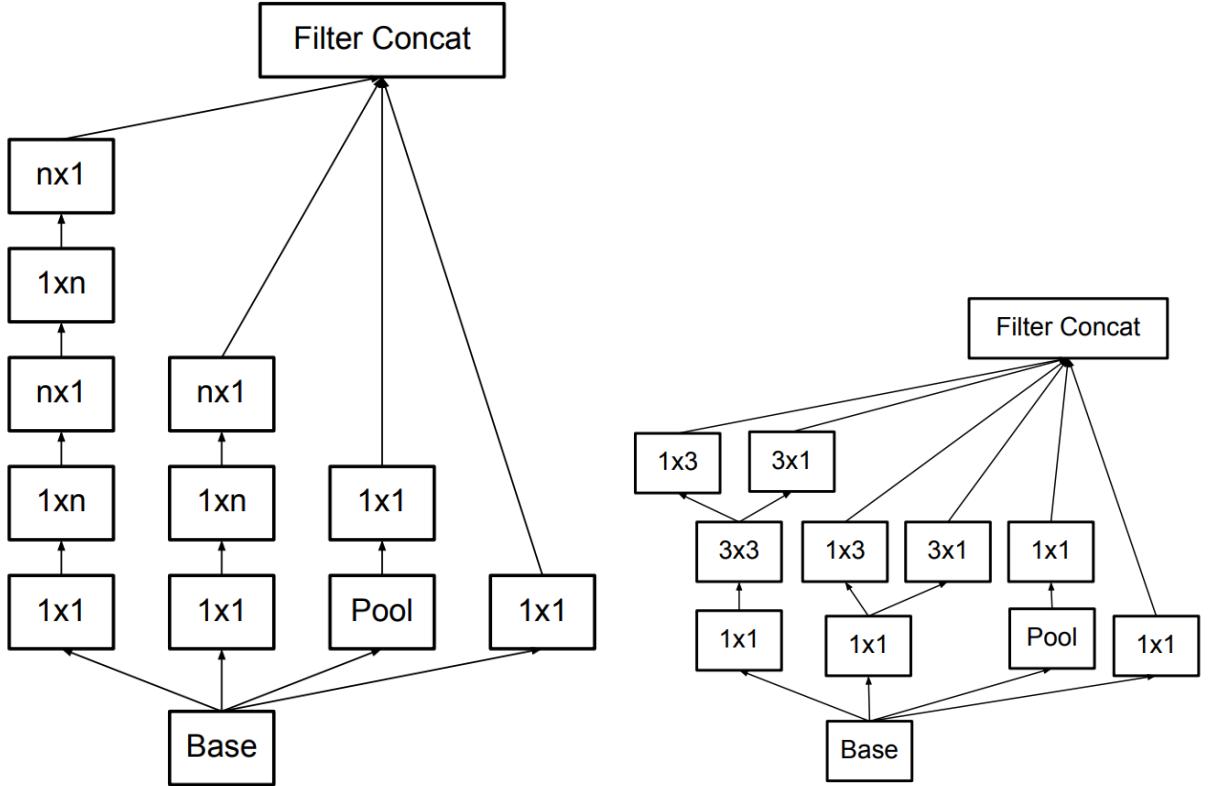


Рис. 1.6: Блок-схема Inception v2 до та після розкладу[48]

В нейронній мережі Inception v2 вперше було застосовано метод пакетної нормалізації [38], що дозволило значно пришвидшити навчання. Також, було показано, що додаткові класифікатори не мають впливу на затухання градієнтів, але мають властивості регуляризації, особливо на останніх стадіях навчання.

Сімейство моделей нейронних мереж ResNet. Сімейство нейронних мереж ResNet було засновано на новому методі залишкового навчання, що був представлений в тій самій роботі, що й архітектура нейронної мережі [7]. Автори розв'язують проблему того, що збільшення глибини ней-

ронної мережі зменшує точність як на навчальному, так і на тестувальному наборі даних.

В роботі показано, що метод стохастичного градієнтного спуску не придатний до оптимізації глибоких нейронних мереж. Так, наведено експеримент, в якому між шарами неглибокої нейронної мережі використано шари тотожного відображення або додаткові згорткові шари з функцією активації. Показано, що для більш глибокої нейронної мережі має існувати рішення з точністю не меншою, ніж для неглибокої нейронної мережі, але, на практиці, точність більш глибокої нейронної мережі залишається нижчою. Для збільшення глибини нейронних мереж без втрати точності, в роботі запропоновано метод залишкового навчання.

Нехай $\mathcal{H}_l(x_{l-1})$ - це відображення, що задається декількома шарами нейронної мережі, де x_{l-1} - це вхідний тензор до першого з цих шарів. Якщо припустити, що декілька нелінійних шарів можуть апроксимувати будь-яку нелінійну функцію, то логічно припустити, що вони можуть апроксимувати і залишкову функцію: $\mathcal{F}_l(x_{l-1}) = \mathcal{H}_l(x_{l-1}) - x_{l-1}$. Тоді, оригінальне відображення стає $\mathcal{H}_l(x_{l-1}) = \mathcal{F}_l(x_{l-1}) + x_{l-1}$. Хоча ці відображення є еквівалентними, алгоритму оптимізації «простіше» таким чином вивчити тотожну функцію, оскільки для цього достатньо наблизити ваги шарів нейронної мережі до нуля.

В роботі автори пропонують блоки нейронних мереж, що складаються з декількох згорткових шарів з функцією активації ReLU та об'єднання з вхідним тензором першого шару. Схема таких блоків зображена на рис. 1.7

Завдяки використанню таких блоків, стала можливою побудова надзвичайно глибоких нейронних мереж. В експериментах, автори запропонували архітектури, що складаються з 18, 34, 50, 101, 152 та 1202 згорткових шарів, та можуть бути оптимізовані стандартними методами (наприклад, методом стохастичного градієнтного спуску).

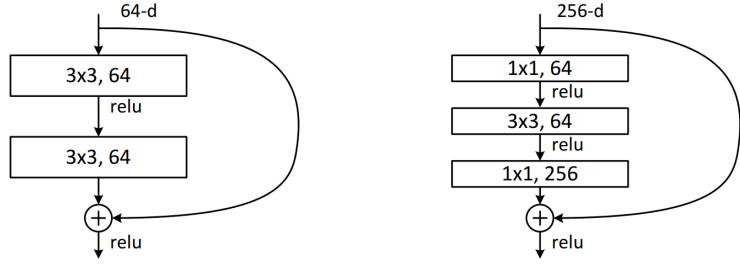


Рис. 1.7: Схема залишкових блоків нейронних мереж ResNet18 та ResNet50 [7]

Сімейство моделей нейронних мереж DenseNet. Архітектура нейронних мереж сімейства DenseNet [49] є логічним продовженням ідей, запропонованих в архітектурі ResNet. В даній архітектурі також використовується метод залишкового навчання, але, на відміну від архітектури ResNet, послідовні шари поєднані кожний з кожним.

Для того, щоб підвищити об'єм інформації, яким обмінюються шари нейронної мережі, автори пропонують ввести пряме з'єднання від кожного шару до всіх наступних:

$$x_l = \mathcal{H}([x_0, x_1, x_3 \dots x_{l-1}]) \quad (1.1)$$

де $[x_0, x_1, x_3 \dots x_{l-1}]$ - конкатенація всіх карт активації з шарів від 0 до $l-1$.

Така зміна архітектури дозволила останнім шарам перевикористовувати ознаки з попередніх, що, в свою чергу, дозволило зменшити кількість параметрів в нейронній мережі та підвищити точність.

Граф нейронної мережі DenseNet зображене на рис. 1.8.

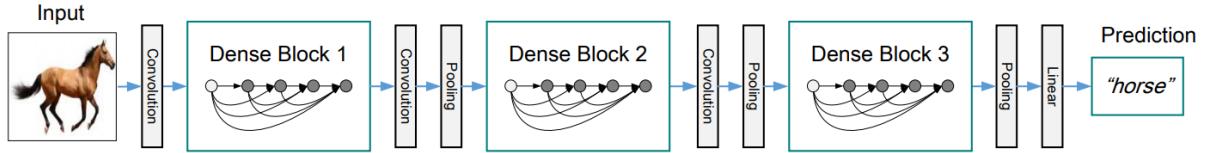


Рис. 1.8: Графічна репрезентація моделі нейронної мережі DenseNet [49]

На практиці, хоча архітектура дозволяє досягти більшої точності з ви-

користанням меншої кількості параметрів, на час виходу роботи, більшість реалізацій, заснованих на популярних фреймворках машинного навчання вимагали більшої кількості оперативної пам'яті через необхідність копіювання карт активацій для кожного з шарів. Таким чином, наївна реалізація архітектури DenseNet потребувала $\mathcal{O}(\frac{L(L-1)}{2})$ оперативної пам'яті, де L - кількість шарів. З розвитком фреймворків машинного навчання, була реалізована можливість перевикористання тензорів без копіювання, що дозволило знизити потребу оперативної пам'яті до $\mathcal{O}(L)$ [50].

Сімейство моделей нейронних мереж ResNeXt. Архітектури сімейства ResNeXt [51] є логічним продовженням методів, застосованих в архітектурах Inception та ResNet. Основною ідеєю цієї архітектури є використання набору з декількох простих однакових трансформацій в просторі низької розмірності в кожному з шарів нейронної мережі.

Так, автори пропонують розділити згортковий шар на декілька паралельних трансформацій, результати з яких агрегуються:

$$\mathcal{H}(x_{l-1}) = \sum_{i=1}^C \mathcal{T}_i(x_{l-1}) \quad (1.2)$$

де \mathcal{T}_i - це будь-яка нелінійна функція, що проєктує вхід x_{l-1} до простору низької розмірності, трансформує його, та проєктує результат назад в простір високої розмірності. Тут C - потужність множини трансформацій.

Графічну презентацію блоку ResNeXt зображено на рис. 1.9

В роботі автори проводять експерименти відносно важливості параметру C відносно інших, та показують, що збільшення C підвищує точність більше, ніж підвищення кількості каналів згорток. Також, в роботі показано, що такий метод здатний створювати кращі репрезентації вхідних даних.

Сімейство моделей нейронних мереж EfficientNet. Сімейство нейронних мереж EfficientNet також є логічним продовженням архітектури ResNet. В даній роботі вперше розглядається параметрична модель та уза-

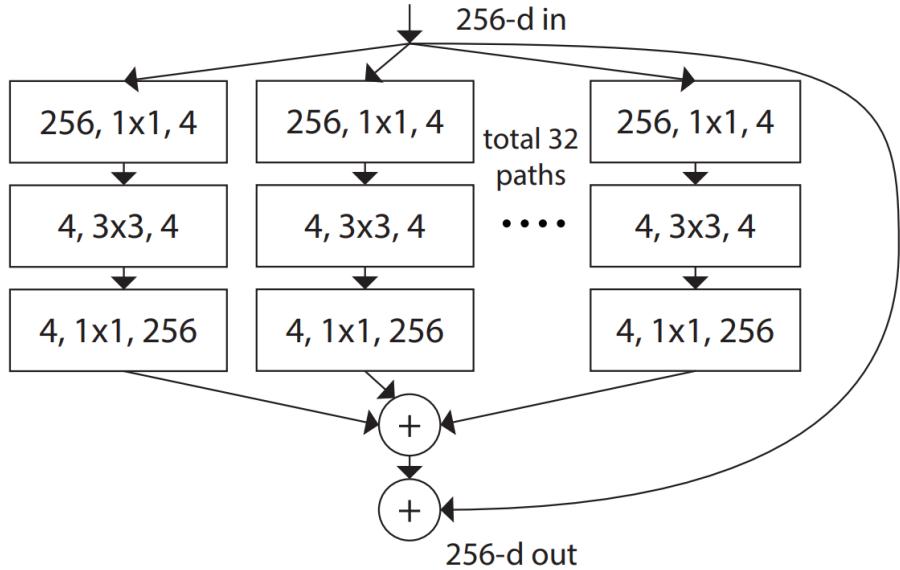


Рис. 1.9: Графічна репрезентація блоку ResNeXt [51]

галальнений метод масштабування згорткових нейронних мереж для збільшення точності під заданий обчислювальні можливості.

Автори пропонують метод, за допомогою якого можна масштабувати одночасно вхідну роздільну здатність зображення, глибину та ширину нейронної мережі. Наведено спостереження, що ці величини не є незалежними.

Завдяки цьому методу масштабування, автори представляють 8 варіантів нейронної мережі, що відрізняється необхідною точністю та продуктивністю: від найменшої до найбільшої.

1.2.3. Визначні моделі ШНМ для сегментації зображень.

Повністю згорткова нейронна мережа. (англ. *Fully-convolutional network, FCN*) [52] - це перша нейронна мережа, що використовувала лише шари згортки, активації та підвибірки, не використовуючи повнозв'язних шарів в задачі семантичної сегментації.

Основною перевагою цього підходу стала можливість навчання та прогнозування на зображеннях різного розміру, без зміни параметрів нейронної мережі.

В даній роботі вперше було застосовано метод трансферного навчання

з задачі класифікації на задачу сегментації. Для цього перші шари нейронної мережі було ініціалізовано за допомогою параметрів, отриманих після тренування на наборі даних ImageNet [42].

Для отримання масок сегментації, повнозв'язні шари було сконвертовано в згорткові, а також було додано інформацію з перших шарів для збереження просторових ознак. Граф нейронної мережі FCN зображене на рис. 1.10.

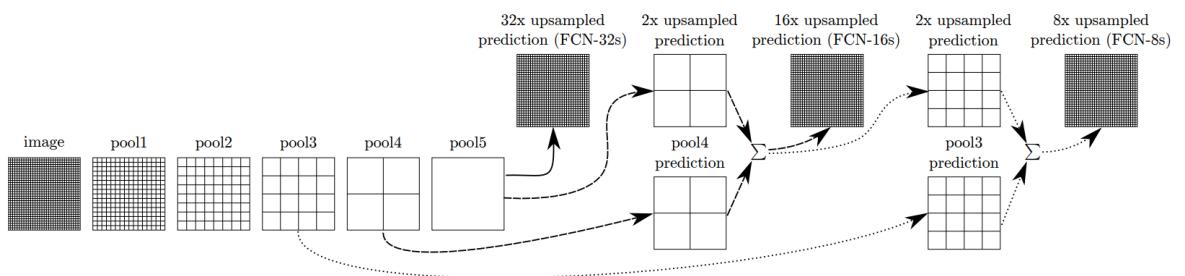


Рис. 1.10: Графічна репрезентація моделі нейронної мережі FCN [52]

Тут, *pool[1-5]* - карти активації після відповідних операцій підвибірки, а *upsampled* - карти активації після білінійної інтерполяції.

Модель нейронної мережі SegNet. SegNet (*сегментаційна мережа*) [53]- перша повністю згорткова нейронна мережа, що використовувала архітектуру типу "енкодер-деокдер" для вирішення задачі семантичної сегментації. В даній архітектурі, енкодер стискає семантичне представлення до більш компактного простору. Роль декодера полягає у відображені з компактного простору з низькою роздільною здатністю у карту активації повного розміру для класифікації кожного пікселя.

Архітектура мережі енкодера топологічно ідентична 13 згортковим шарам мережі VGG16 [45].

Новина SegNet полягає в тому, як декодер підвищує розширення карт активації низького розширення. Так, декодер використовує індекси операцій підвибірки, обчислені на відповідному шарі енкодера, для нелінійного

підвищення розширення. Карти активації підвищеного розширення виявляються розрідженими, для того, щоб зробити їх щільними, виконується декілька операцій згортки. Структура SegNet зображена на рисунку 1.11.

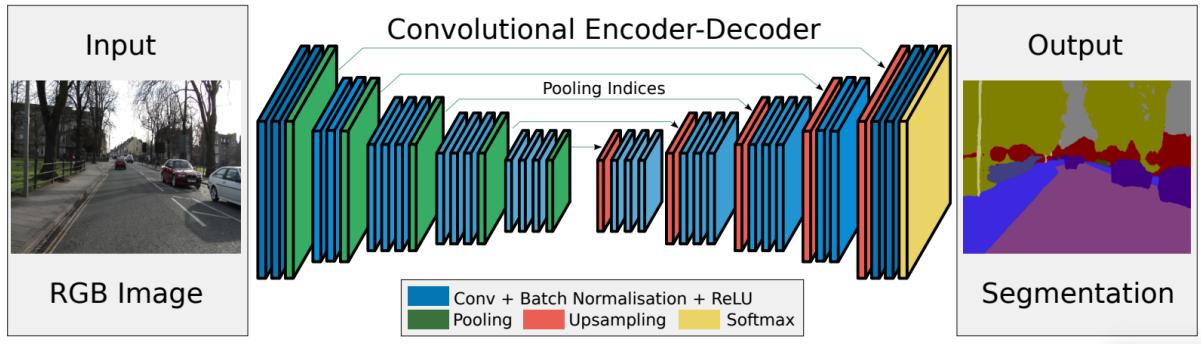


Рис. 1.11: Граф нейронної мережі SegNet [53]

Модель нейронної мережі UNet. U-Net (*U-подібна мережа*) [54] - повністю згорткова нейронна мережа, що складається з симетричних енкодера та декодера, в яких, на відміну від SegNet замість індексів підвибірки, до декодера передаються карти активації енкодера напряму, через операцію конкатенації. Окрім спрощення кодування, така організація мережі дозволила спростити навчання, оскільки передча інформації стала повністю диференційованою.

Також, для забезпечення можливості навчання на малій кількості даних, використовується велика кількість різноманітних аугментацій вхідних зображень, щоб синтетично збільшити розмір набору даних.

Для більш чіткого розподілення близько розташованих об'єктів, автори запропонували спеціальну зважену функцію втрат, що забезпечувала більш чітке формування границь об'єктів.

Хоча, оригінально ця архітектура була запропонована для семантичної сегментації планарних мікроскопічних знімків, подальші дослідження показали її застосовність в широкому спектрі задач семантичної сегментації.

Структура нейронної мережі U-Net зображена на рисунку 1.12.

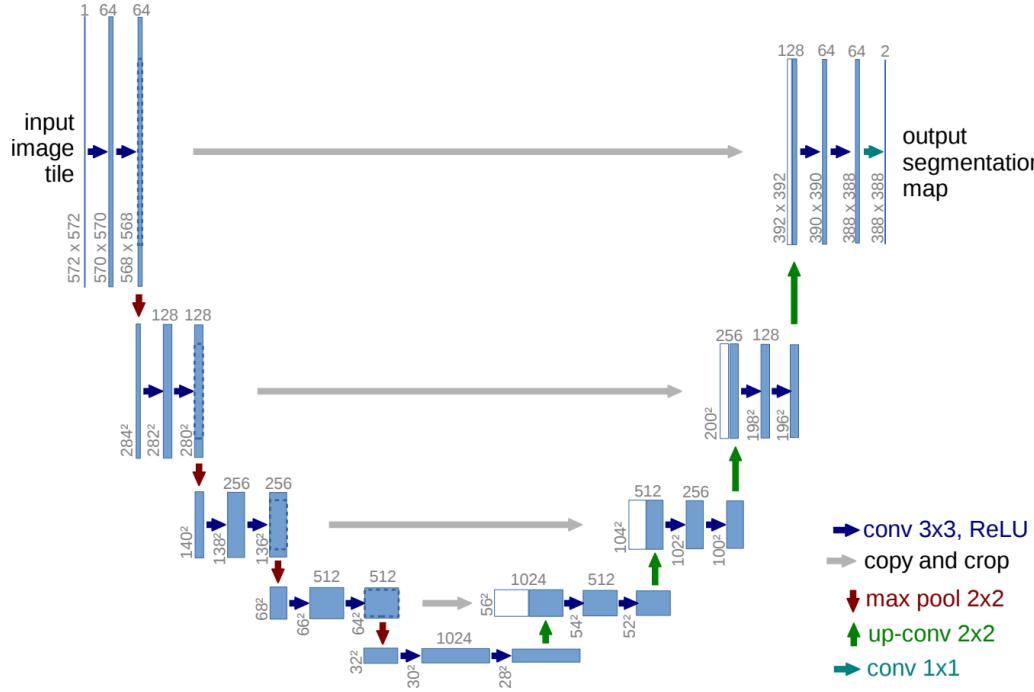


Рис. 1.12: Граф нейронної мережі U-Net [54]

Модель нейронної мережі LinkNet. Архітектура нейронної мережі LinkNet є подальшим розвитком архітектури U-Net, і також складається з енкодера та декодера, що поєднані через проміжні карти активації.

На відміну від U-Net, LinkNet використовує ResNet в якості енкодера, а також для поєднання карт активації енкодера та декодера використовує операцію суми, замість конкатенації. Це дозволило значно зменшити кількість параметрів, підвищити швидкість тренування при збереженні точності сегментації.

Також, автори LinkNet вперше використали індуктивний перенесі з ImageNet виключно до енкодера, залишивши декодер випадково ініціалізованим на момент початку навчання.

Структура нейронної мережі LinkNet зображена на рисунку 1.13.

В подальших роботах, автори зазвичай називають свої архітектури U-Net-подібними, хоча, насправді, вони є близчими до LinkNet. [56, 57, 58]

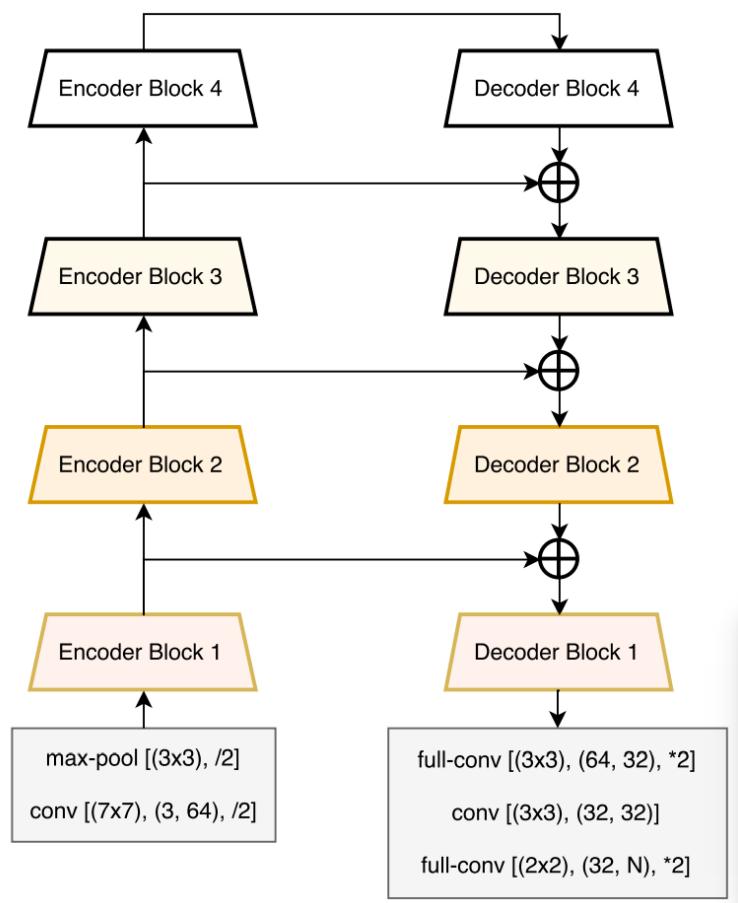


Рис. 1.13: Граф нейронної мережі LinkNet [55]

Модель нейронної мережі Feature Pyramid Network. Вперше розроблена для задач детекції об'єктів, Feature Pyramid Network (FPN) [59], була успішно застосована для задач семантичної сегментації [60].

Структура FPN складається з двох шляхів: на одному з них зменшується просторова та розмірність збільшується глибина карт ознак (шлях знизу вверх), на іншому - навпаки (шлях зверху-вниз). Шлях знизу-вверх представляє собою звичайну згорткову нейронну мережу (зазвичай, з сімейства ResNet). Шлях зверху-вниз представляє собою послідовність згорток та операцій підвищення розширення.

Для отримання карти активації, з якої буде отримано результат семантичної сегментації, всі рівні декодера оброблюються двома операціями згортки, після чого приводяться до однакового просторового розширення. З цієї карти ознак отримується фінальна карта сегментації за допомогою операції згортки.

Структура нейронної мережі FPN для семантичної сегментації зображена на рисунку 1.14.

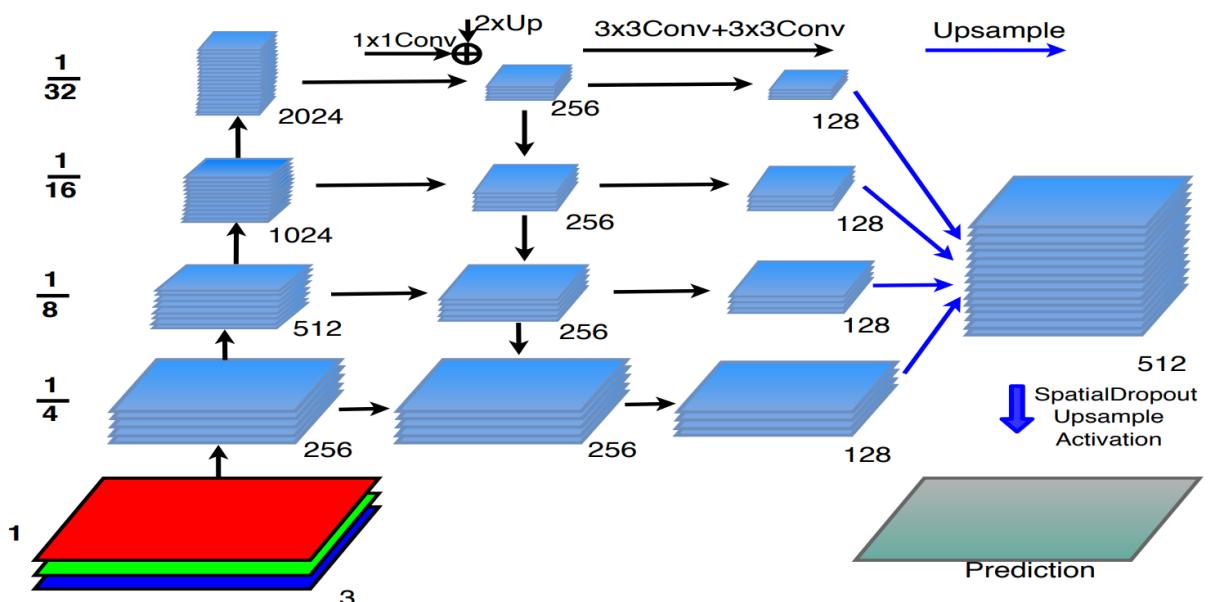


Рис. 1.14: Граф нейронної мережі FPN [60]

1.2.4. Тестування моделей в задачах автоматизованого скрінінгу.

Набори даних в задачах автоматизованого скрінінгу. Як і в загальній задачі машинного навчання, при розробці систем автоматизованого скрінінгу розрізняють три етапи, для кожного з яких потрібний окремий набір даних: етап навчання (тренування), тестування та валідації.

Навчальний набір даних - це набір прикладів та відповідних анотацій, що використовуються під час навчального процесу, і використовується для зміни параметрів моделі (наприклад, класифікації, або сегментації) [61].

В задачах глибинного навчання, алгоритм навчання використовує набір навчальних даних, щоб вивчити оптимальні параметри моделі (відповідно до критерію оптимальності), які забезпечують хорошу здатність моделі до передбачення результатів. Метою навчання є створення навченої моделі, яка добре узагальнює нові, невідомі дані. Більшість методів, які використовують тренувальні набори даних для підбору параметрів моделі схильні до перенавчання, тобто до вивчення виключно точок з цього набору, без здатності до узагальнення на нових даних.

Валідаційний набір даних - це набір прикладів та відповідних анотацій, що використовуються для налаштування гіперпараметрів (включно зі структурою) моделей та методів їх навчання. Прикладом гіперпараметрів для штучних нейронних мереж може бути кількість шарів в кожному шарі, або варіанти з'єдніння цих шарів. Він повинен відповідати такому ж розподілу ймовірностей, що й набір навчальних даних та генеральна сукупність даних.

Щоб уникнути перенавчання моделей до тренувального набору даних, валідаційний набір даних використовується для порівняння їх ефективності та вибору найкращої. Цей набір даних є частиною навчальних даних,

які використовуються для тестування, але не як частина навчання, та не як частина фінального тестування.

Оскільки процедура вибору найкращої моделі може привести до перенавчання до валідаційного набору даних, достовірність передбачень вибраної мережі слід підтвердити, вимірювши її на третьому незалежному наборі даних, який називається тестовим набором даних.

Набір тестових даних - це набір прикладів з відповідними анотаціями, який не залежить від набору навчальних даних, але обраний з тієї ж генеральної сукупності, що й тренувальний набір даних. Якщо модель, що має високу достовірність на наборі навчальних даних, також має високу достовірність на тестовому наборі даних, то перенавчання є мінімальним. Більш висока достовірність на тренувальному наборі даних ніж на тестовому, зазвичай сигналізує про перенавчання. При використанні перехресної перевірки, тестовий набір даних може не використовуватися.

Для оцінки достовірності передбачень, можуть використовуватися різні метрики, що обчислюються між прогнозами моделей та анотаціями з відповідного набору даних.

З точки зору достовірності розмітки, набори даних можна розділити на достовірно-анотовані, з частково-помилковими анотаціями та недостовірні набори даних.

Так, в достовірно-анотованих наборах даних кожна з анотацій відповідає істинному класу об'єкта, або пікселя. В наборах даних з частково-помилковими анотаціями деякі з анотованих пікселів, або зображень не відповідають істинному класу об'єкта. В недостовірних наборах даних, анотації є результатом шумового процесу та не містять корисної інформації, з якої можна було б зробити висновок про істині класи об'єктів.

Більшість реальних наборів даних мають частково-помилкові анотації, повністю достовірні та недостовірні анотації зустрічаються дуже рідко в наборах даних для реальних задач та, зазвичай, є предметом особливих

досліджень, однак проблема частково-помилкових анотацій недостатньо представлена в літературі.

Також, отримання достовірних анотацій для валідаційного і тестового наборів даних часто ускладнюється, оскільки анотації збираються за допомогою людей-анотаторів, які допускають помилки. В деяких випадках, немає можливості отримати набір даних з реальної предметної області для оцінки методу. В такому разі, дослідники використовують генерацію синтетичних даних, відповідно до характеристик, притаманних прикладній області.

Так, автори [62] запропонували генерацію набору коротких відео-послідовностей з цифр, зо рухаються для перевірки можливостей рекурентної моделі до передбачення наступної позиції в послідовності.

Автори [63] розширили попередній набір даних на задачу семантичної сегментації з додаванням зашумлення вхідних даних.

В інженерній роботі [64] запропоновано параметричну модель генерації наборів даних для оцінки моделей та методів семантичної сегментації. Об'єкти даного набору даних є рукописними цифрами MNIST, а фон генерується з випадкових текстур, при цьому, будь-яке зашумлення розмітки відсутнє.

Методи оцінки якості класифікації та сегментації зображень в задачі автоматизованого скринінгу. Для тестування моделей та методів машинного навчання, необхідно визначити метрики, за якими оцінюються достовірність передбачень. В задачах класифікації та сегментації використовуються наступні метрики.

Помилки 1 і 2 роду та матриця невідповідностей

В задачах математичної статистики, помилка першого роду - це хибне відхилення правильної гіпотези (хибно-позитивний результат, FP), тоді як помилка другого роду - це прийняття хибної гіпотези (хибно-негативний

результат, FN). Дані поняття використовуються, коли потрібно прийняття бінарного рішення на основі деякого критерію, що може мати похибку.

В задачах контролюваного навчання, зокрема бінарної класифікації та сегментації, за нульову гіпотезу зазвичай приймається приналежність елемента вибірки до класу, що зустрічається в виборці частіше. Також, в задачах багатокласової класифікації використовується розширення поняття помилок першого і другого роду на задачу з більшою кількістю гіпотез - матриця невідповідностей. Матриця невідповідностей - це матриця, в строках якої зазначені зразки прогнозованого класу, а кожен із стовпців представляє зразки справжнього класу. Така матриця дозволяє оцінити, чи допускає система невідповідності між класами.

Влучність, чутливість та специфічність

Влучність, чутливість та специфічність є похідними мірами якості бінарного класифікатора та розраховуються на основі значень в матриці невідповідностей. Влучність (англ. precision) - вимірює частку істинно-позитивних зразків серед знайдених.

$$Precision = \frac{TP}{TP + FP} \quad (1.3)$$

Чутливість, або повнота (англ. sensitivity, recall) - вимірює частку істинно-позитивних зразків серед усіх позитивних зразків.

$$Recall = \frac{TP}{TP + FN} \quad (1.4)$$

Специфічність (англ. specificity) - вимірює частку істинно-негативних зразків серед усіх негативних зразків.

$$Specificity = \frac{TN}{TN + FP} \quad (1.5)$$

Між влучністю і повнотою, та чутливістю і специфічністю існує обернена залежність, коли можливо підвищити одну ціною зниження іншої. Чезрез наявність такої залежності, на практиці, ці міри не використовуються

окремо. Замість цього використовують агреговані метрики, що дозволяють оцінити якість класифікатора в цілому.

Міра F1 та індекс подібності Дайса-Соренсена

F1 та індекс подібності Дайса-Соренсена (англ. *Dice score*) є еквівалентними мірами, що, зазвичай використовуються в задачах класифікації (F1-міра) та сегментації (Dice score).

Міра F1 визначається як середнє гармонічне між влучністю та повнотою:

$$F1 = \frac{2}{precision^{-1} + recall^{-1}} = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} \quad (1.6)$$

Індекс подібності Соренсена визначається на множинах, і, в задачах сегментації, визначається на піксельних масках сегментації:

$$D_c = 2 \cdot \frac{A \cap B}{|A| + |B|} \quad (1.7)$$

Також, для бінарних даних $A \cap B$ відповідає TP , а $|A| + |B|$ відповідає всій множині об'єктів, тобто еквівалентно $TP + FP + FN$. Таким чином, коефіцієнт Соренсена є еквівалентним мірі F1.

1.3. Методи багатозадачного глибинного машинного навчання

Одним з варіантів розв'язання задачі машинного навчання на наборах даних з частково-помилковими анотаціями є багатозадачне машинне навчання.

Багатозадачне машинне навчання - це один з підходів машинного навчання, в якому одночасно вирішуються кілька навчальних задач, використовуючи спільноті та відмінності між ними. Кожна з таких задач може бути загальною задачею, такою як навчання з учителем (наприклад, задача

класифікації, або регресії), навчання без вчителя (кластеризація), напівавтоматичне навчання, завдання навчання з підкріпленням. Передбачається, що всі ці навчальні задачі або хоча б частина з них пов'язані одна з одною.

В цьому випадку виявлено [65], що спільне навчання на цих задачах може призвести до значного підвищення продуктивності в порівнянні з навчанням на кожній задачі окремо. Отже, багатозадачне навчання направлено на підвищення якості узагальнення кількох пов'язаних задач. Далі будуть розглядатися виключно задачі контролюваного навчання.

Подібно до людського навчання, корисно вивчати кілька навчальних задач разом, оскільки знання, що містяться в одній задачі, можуть бути використані іншими. Наприклад, людина, що вивчає математику та математичну статистику, може використати цей досвід для вивчення інших схожих галузей, наприклад, машинного навчання.

1.3.1. Визначення багатозадачного машинного навчання. Для п навчальних задач T_1, T_2, \dots, T_n , де всі задачі або їх підмножина пов'язані, багатозадачне навчання (БЗМН) має на меті допомогти покращити вивчення параметрів моделі для T_i , використовуючи інформацію, що міститься в усіх п або деяких з них задачах.

В літературі визначається декілька видів БЗМН:

Однорідне БЗМН: Для кожної з задач прогнозується лише один вихід. Наприклад, розпізнавання цифр MNIST зазвичай використовується для оцінки алгоритмів БЗМН. В цьому разі, воно розглядається як 10 завдань бінарної класифікації.

Неоднорідне БЗМН: Для кожної з задач прогнозується свій набір виходів. Наприклад, сучасні задачі детектування об'єктів потребують одночасного прогнозування класу об'єкта (задача класифікації), його положення на зображені та розмірів (задача регресії). Для задач контролюваного навчання, кожна задача T_i містить в собі набор даних D_i , що складається

з t елементів...

Дослідники вважають, що коли різні задачі використовують однакові набори вхідних даних, БЗМН зводиться до задачі класифікації або регресії з кількома виходами. Однак, новіші дослідження визначають БЗМН на одному наборі вхідних даних, або незалежно від кількості наборів вхідних даних.

В методах БЗМН, заснованих на глибоких нейронних мережах, виділяють дві групи відповідно до способу розподілу параметрів між різними задачами.

Трансферне навчання. Трансферне навчання – це проблема машинного навчання, яка фокусується на збереженні знань, отриманих під час вирішення однієї задачі з подальшим застосуванням її результатів до іншої, але близької до неї. З практичної точки зору, повторне використання або передача інформації з раніше засвоєних завдань для вивчення нових може значно підвищити ефективність їх розв'язання [?]. Деякі дослідники [66, 67] вважають багатозадачне машинне навчання підмножиною трансферного навчання.

На відміну від трансферного навчання, в багатозадачному навчанні немає різниці між задачами, та завдання стоїть підвищити продуктивність на всіх задачах одночасно.

Жорсткий розподіл параметрів. Зазвичай, архітектури згорткових нейронних мереж для багатозадачного навчання складаються зі спільногого енкодера, який виділяє ознаки на вхідному зображені (shared features), та окремих згорткових і/або повнозв'язних шарів для кожної з задач.

В найпершій роботі [68], що використовує цей принцип, було запропоновано архітектуру нейронної мережі Task-Constrained Deep Convolutional Network (TCDCN) для детекції ключових точок лиця використати додаткові задачі регресії пози голови та класифікації атрибутів. Архітектуру

TCDCN зображене на рис. 1.15.

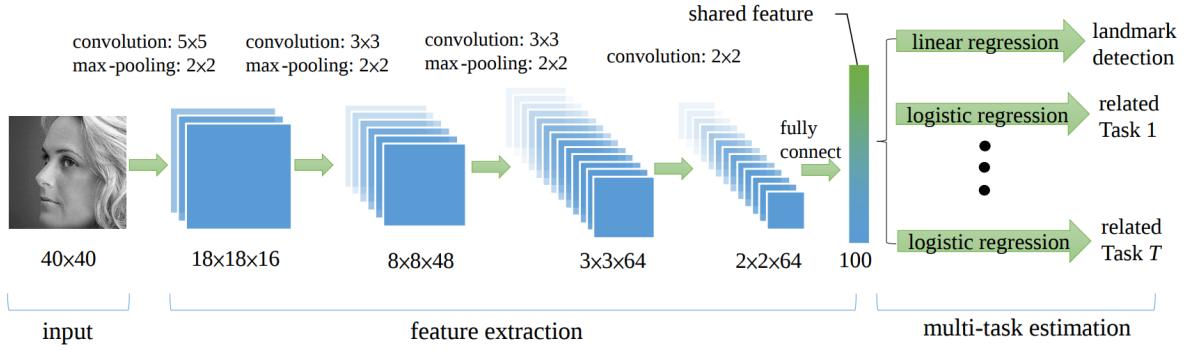


Рис. 1.15: Архітектура нейронної мережі TCDCN [68]

Модифікацією попереднього підходу є багатозадачні каскадні мережі ??, в яких результати одних задач використовуються як доповнення до вхідної інформації для наступних. Така організація шарів дозволяє підвищити точність на більш складних завданнях при правильному підборі черги задач за складністю. Загальну архітектуру зображене на рис. 1.16. Схожий підхід також використовується в деяких архітектурах нейронних мереж для розв'язання задачі детекції об'єктів [69, 70, 71]

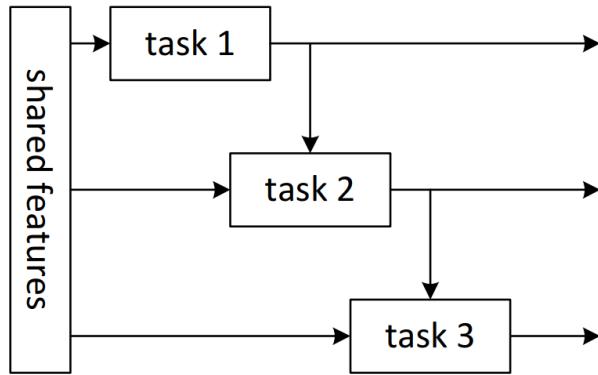


Рис. 1.16: Архітектура багатозадачної каскадної мережі [72]

Також, можлива інтеграція модулів уваги як до енкодера, так і до специфічних до задачі шарів. Таким чином, обчислення ознак відбувається не лише за рахунок параметрів енкодера, а і через специфічні до задачі модулі, що розташовані всередині мережі. Це дозволяє покращити репрезентації ознак для конкретних задач.

Можливою проблемою при багатозадачному навчанні на семантично-віддалених задачах є конфлікт градієнтів [73]: градієнти для різних задач мають протилежні напрямки. Для розв'язання цієї проблеми було запропоновано декілька підходів [74, 75, 73], найпростішим з яких є навчання на задачах, що не викликають конфлікту [73].

М'який розподіл параметрів. Одним з варіантів архітектур, що менш схильні до утворення конфліктних градієнтів є методи архітектури з м'яким розподілом параметрів.

В методах, що використовують м'який розподіл параметрів використовується декілька енкодерів для кожної з задач, між шарами яких відбувається обмін інформацією.

Наприклад, в роботі Cross-stitch networks (перехресні мережі) [76] представлена архітектура багатозадачної нейронної мережі, в якій кожній задачі відповідає окремий енкодер, але входом кожного шару кожного з енкодерів є лінійна комбінація виходів попереднього шару всіх енкодерів. Коефіцієнти лінійної комбінації також вивчаються за допомогою градієнтного спуску разом з іншими параметрами нейронної мережі. На рисунку 1.17 зображені архітектуру блоку лінійної комбінації для шарів двох нейронних мереж.

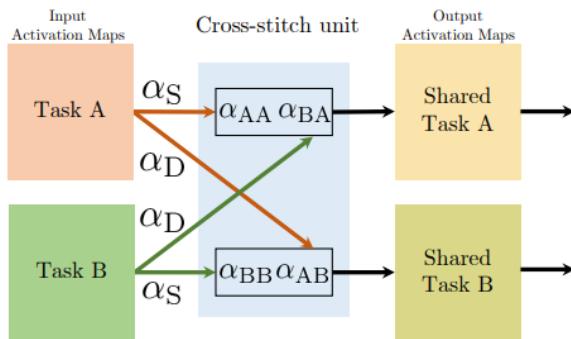


Рис. 1.17: Архітектура cross-stitch блоку [76]

Узагальненням ідеї переносу інформації між послідовними шарами де-

кількох специфічних до задачі нейронних мереж є шлюзові мережі (англ. *Sluice networks*). Вхід до кожного з шарів також є лінійної комбінацією виходів попередніх шарів дляожної з задач. На відміну від перехресних мереж, кожний шар нейронної мережі розділений на окремі підпростори: для конкретної задачі та спільний між всіма задачами. Також, додається обмеження на ортогональність підпросторів через додавання функції втрат, що мінімізує квадратовану норму Фробеніуса (матричну L^2 норму) між підпросторами в кожному з шарів.

Подальший розвиток шлюзових мереж було запропоновано в роботі «Нейромережеве дискримітивне зменшення розмірності» (англ. *Neural Discriminative Dimensionality Reduction, NDDR*) [77]. Тут, замість використання лінійної комбінації шарів між різними задачами, виконується згортка конкатенованих виходів шарів всіх задач з ядром розміру 1×1 з виконанням подальшої нормалізації. Архітектуру NDDR зображену на рисунку 1.18.

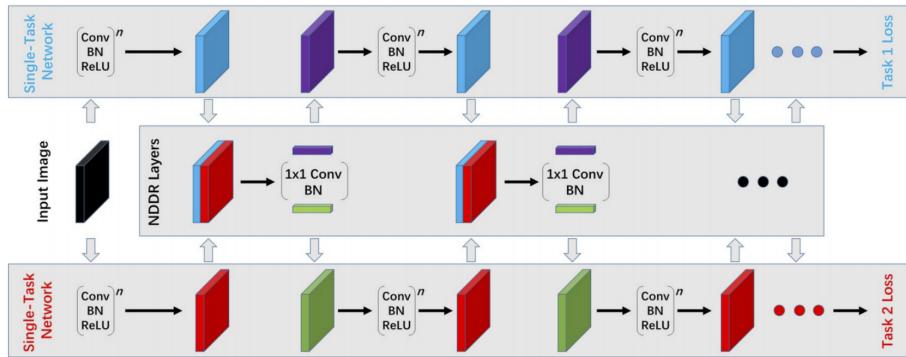


Рис. 1.18: Архітектура згорткової нейронної мережі NDDR [77]

Суттєвим недоліком методів, що використовують м'який розподіл параметрів є постійне збільшення кількості параметрів нейронної мережі, що, в свою чергу, збільшує необхідну кількість ресурсів для роботи такої нейронної мережі зі збереженням продуктивності.

Дистиляція прогнозів моделі. Дистиляція прогнозів замість вивчення ознак, спільних до декількох задач, базується на використанні ре-

зультатів для одних задач, як засобу вивчення ознак для інших. Наприклад, при одночасному вивченні задач класифікації та сегментації, карти сегментації можуть бути використані для подальшої класифікації зображення.

Першою роботою, що використала цей принцип була PAD-Net (Prediction-and-Distillation Network) [78]. В ній використовувалися результати задач семантичної сегментації, регресії карти глибини, регресії карти поверхневих нормалей та детекції контурів для отримання уточнених результатів в задачах регресії карти глибини та семантичної сегментації. Для цього автори запропонували три варіанти спеціальних модулів дистиляції:

- Проста конкатенація результатів;
- Передача повідомлень між задачами
- Передача повідомлень з вагами

Подальшим розвитком цього методу стала робота PAR-Net, в якій було представлено модуль, що окрім вивчає попарні взаємозв'язки між задачами, та об'єднує вивчені карти ознак відповідно до попарних відношень.

1.3.2. Багатозадачне навчання в задачах класифікації та сегментації. Використання багатозадачного навчання в задачах класифікації та сегментації не є поширеним в літературі. Зазвичай, в роботах, що пропонують вирішення задачі сегментації, задача класифікації вирішується на етапі попередньої обробки зображень, щоб відсіяти хибнопозитивні результати сегментації та зменшити час роботи системи в цілому через використання легких моделей класифікації. Недоліком такого підходу є можливість поширення помилки від класифікатора, що в задачах скрінінгу вимагає підвищення повноти класифікатора, з подальшою роботою сегментаційної мережі. Також, в задачах, що мають маленькі об'єкти на зображеннях, чи незбалансовані набори даних, такий підхід може зменшити влучність роботи системи порівняно з її частинами.

На відміну від послідовної обробки, в роботі SegTHOR одночасно розв'язуються задачі класифікації та сегментації, доповнюючи одна одну. Так, автори пропонують нову архітектуру нейронної мережі, що заснована на архітектурі UNet, але має додатковий вихід класифікації з останнього шару мережі сегментації. Отримання глобальних класів з виходу сегментації досягається через середнє арифметичне всіх пікселів для кожного з класів. Архітектура нейронної мережі SegTHOR зображена на рисунку 1.19.

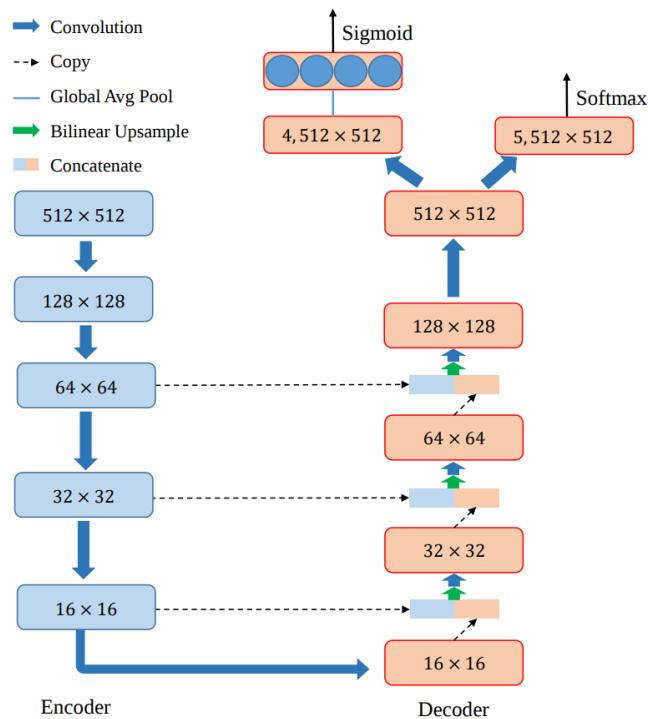


Рис. 1.19: Архітектура методу SegTHOR [79]

В нейронній мережі SegTHOR вихід класифікації використовується лише при тренуванні в якості додаткової задачі, та не використовується під час прогнозування масок сегментації.

В роботі [80] автори запропонували застосувати підхід багатозадачного машинного навчання до класифікації та сегментації новоутворень шкіри. На відміну від звичайних архітектур для сегментації зображень, в роботі запропоновано сегментацію будь-якого новоутворення, з паралельною класифікацією меланоми та себорейного кератозу як окремих задач. Автори

показують, що така архітектура нейронної мережі незначно покращує достовірність як в задачах класифікації, так і сегментації. В даній архітектурі під час прогнозування результати різних задач використовуються окремо.

Автори [81] запропонували одночасно вивчати задачі класифікації та сегментації для локалізації ракових пухлин в мамографії. Як і [80], в даній роботі використовується нейронна мережа, що складається з одного енкодера і двох декодерів - для класифікації та сегментації відповідно. Архітектуру нейронної мережі зображену на рисунку ??.

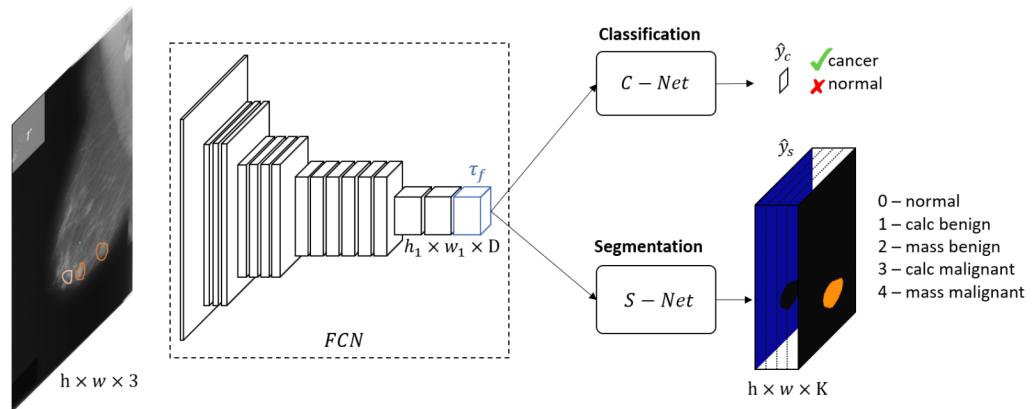


Рис. 1.20: Архітектура нейронної мережі класифікації та сегментації новоутворень на мамографії [81]

В роботі демонструється, що, на відміну від двох нейронних мереж, що вивчають задачі незалежно, запропонована архітектура має більшу точність як класифікації, так і сегментації. В даній архітектурі під час прогнозування результати різних задач використовуються окремо.

Окрім підходів з жорстким розподілом параметрів, в задачі сегментації може використовуватися каскадна схема. В роботі [82] запропоновано архітектуру каскадної нейронної мережі для розв'язання задачі сегментації будівель на аерофотознімках. Для покращення контурів сегментації, в роботі вводиться нова задача регресії піксельної відстані до найближчого контуру. Результат виконання задачі регресії використовується як частина вхідних даних для задачі сегментації. Архітектуру нейронної мережі

зображені на рисунку 1.21. Автори показують, що таке використання декількох задач дозволяє підвищити достовірність сегментації в цілому та якість контурів окремо.

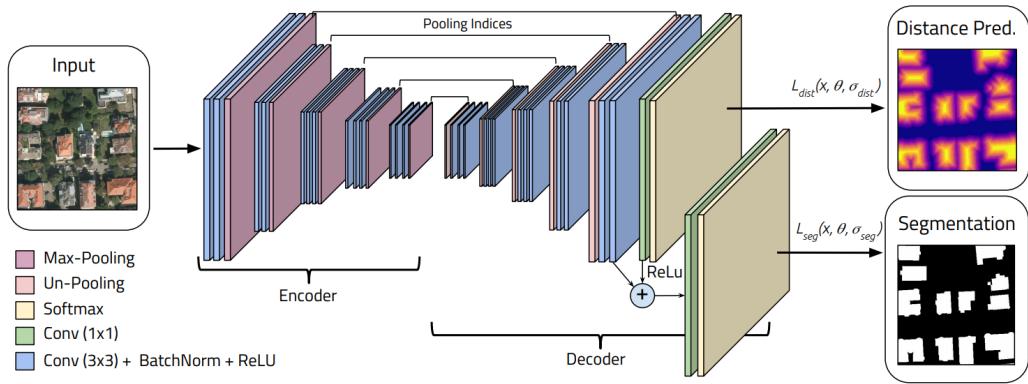


Рис. 1.21: Архітектура нейронної мережі для сегментації будівель [82]

Деякі методи машинного навчання з частковим залученням вчителя використовують архітектури, натхнені багатозадачним машинним навчанням. Так, популярною задачею машинного навчання з частковим залученням вчителя є семантична сегментація зображень з використанням лише міток рівня зображення, точок, або обмежувальних коробок навколо об'єктів. Наприклад, в роботі [83] запропоновано ЕМ-алгоритм сумісно з глибокими нейронними мережами для розв'язання задачі сегментації через тренування нейронної мережі з мітками рівня зображення, або обмежувальних коробок. Задача класифікації в даному випадку використовується на етапі оцінювання очікування в ЕМ алгоритмі як априорна інформація.

Альтернативний метод сегментації з частковим залученням вчителя базується на функції втрат, що дозволяє одночасно оптимізувати наступні задачі:

- регресію локалізації об'єктів
- розширення локацій об'єктів відповідно до класів, присутніх на зображеннях
- обмеження границь масок сегментації об'єктів до границь об'єктів

Архітектура поєднання задач зображення на рисунку 1.22.

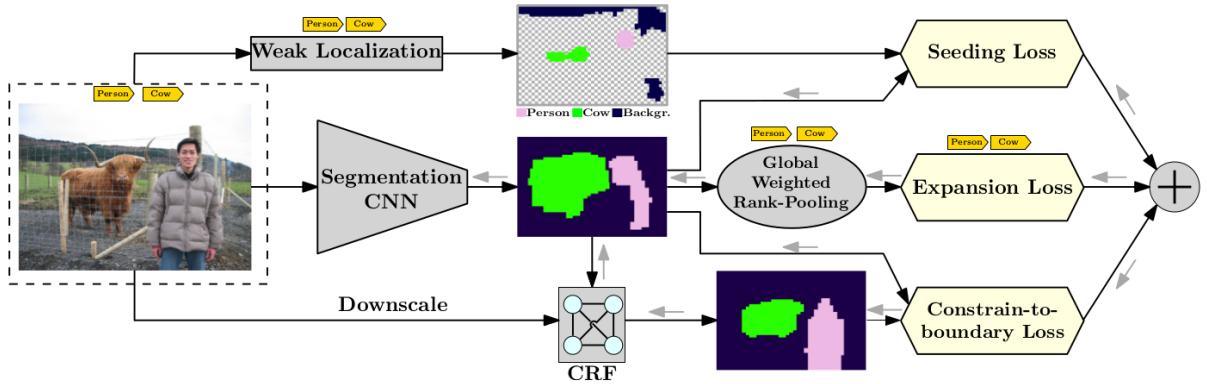


Рис. 1.22: Архітектура методу SEC-CNN [?]

1.4. Висновки до первого розділу

В **першому розділі** дисертаційної роботи проведено аналіз проблем аналізу планарних зображень (які не мають виміру глибини, та в яких можна знехтувати масштабом об'єктів) в системах автоматизованого скринінгу на основі нейромережевих технологій.

Проаналізовано існуючі системи автоматизованого скринінгу в предметних областях медицини, метеорології та віддаленого зондування та особливості їх функціонування шляхом аналізу планарних зображень. Показано, що ефективність роботи таких систем безпосередньо залежить від достовірності класифікації та сегментації зображень, оскільки хибно-позитивні результати можуть привести до передчасного реагування, а хибно-негативні результати - до пропуску наявних проблем та пізнього реагування.

Крім того, не менш важливим фактором, що впливає на ефективність скринінгових систем є витрати часу на реалізацію процесу навчання, що в свою чергу пов'язано з якістю попередньо створених навчальних вибірок даних. Але, в задачах скринінгу, отримання достатньої кількості даних та проведення якісного аnotування професіоналами є, як правило, затратним,

а іноді неможливим через малу кількість та різний вигляд навчальних прикладів, суб'єктивність під час формування анотацій в наборах даних. Ці фактори негативно впливають на достовірність класифікації та сегментації при навчанні глибинних нейронних мереж стандартними методами. Для усунення такого впливу запропоновано при побудові моделей класифікації та сегментації вирішити задачу навчання з урахуванням частково-помилкових анотацій.

Показано, що в таких умовах доцільне використання глибинних нейронних мереж, через можливість автоматизації процесу навчання та застосування методів багатозадачного навчання, які підвищують достовірність класифікації та сегментації.

Проведений аналіз наявних моделей наборів даних та методів їх генерації показав обмеженість моделей наборів даних, які б дозволяли формувати частково-помилкові анотації, які характерні для реальних задач автоматизованого скринінгу.

Таким чином, вирішення важливої науково-практичної задачі підвищення достовірності класифікації та сегментації в задачах аналізу планарних зображень для систем автоматизованого скринінгу шляхом удосконалення моделей нейронних мереж та методів їх багатозадачного навчання, особливо за умови частково-помилково анотованих даних навчальної вибірки.

Метою дослідження є підвищення достовірності класифікації та сегментації планарних зображень в системах автоматизованого скринінгу шляхом розробки нейромережевих моделей і методів.

Об'єктом дослідження є процеси аналізу планарних зображень в системах автоматизованого скринінгу.

Предмет дослідження – нейромережеві моделі та методи класифікації і сегментації планарних зображень в задачах автоматизованого скринінгу.

РОЗДІЛ 2

ПАРАМЕТРИЧНА МОДЕЛЬ НАБОРІВ ДАНИХ ІЗ ЧАСТКОВО-ПОМИЛКОВИМИ АНОТАЦІЯМИ ТА МЕТОД ЇХ ГЕНЕРАЦІЇ

Для систематизованого вивчення впливу помилок в анотаціях, та оцінки відповідного впливу запропонованих рішень, розроблено модель набору даних із частково-помилковими анотаціями семантичної сегментації, що відповідає оцінкам моделей шуму в різних задачах автоматизованого скрінінгу.

2.1. Проблема частково-помилкових анотацій в наборах даних

В задачах машинного навчання, поняття частково-помилкових анотацій (*англ. label noise*) є досить розмитим і його нелегко визначити точно. В літературі цей термін використовується для позначення невідповідності анотацій істинним даним, її пошкодження, або неідеальність.

Треба відрізняти частково-помилкові анотації від шуму у вихідних даних. Так, для множини зображень X та відповідної їй множини міток Y в наборі даних утворюються пари (x_i, y_i) . Зазвичай, в задачах класифікації та сегментації, множина Y містить дискретні мітки y_i рівня зображення, або рівня окремих пікселей відповідно. Ці мітки часто відповідають справжньому класу елемента x_i , але вони можуть бути піддані впливу шумового процесу перед тим, як будуть представлені алгоритму навчання.

2.1.1. Природні причини помилок в анотаціях в наборах даних.

Відповідно до [84], ідентифікація джерел помилок в анотаціях не

обов'язково важлива, коли фокус аналізу зосереджений на наслідках цих помилок. Однак, коли моделі розподілу помилок мають бути вбудовані безпосередньо в алгоритми навчання, або оптимізації, яка точно пояснює джерела та вид шуму.

Зазвичай, природні помилки в анотаціях відбувається, коли до процесу анотування залучені люди-анотатори. Тоді, причинами некоректних анотацій можуть бути візуально-схожі структури, недостатній досвід роботи анотаторів, випадкові помилки через необережність, а також людські упередженості.

Автори [85] пропонують наступну класифікацію видів зашумлення розмітки:

- Рівномірні помилки - ймовірність зміни мітки класу на мітку будь-якого іншого класу розподілена рівномірно.
- Залежні від класу помилки - ймовірність зміни мітки класу на мітку іншого класу залежить від цих класів.
- Залежні від ознак помилки - ймовірність зміни мітки класу на мітку іншого класу залежить від ознак об'єкта. На відміну від інших двох видів, цей майже не зустрічається в роботах через складність виконання.

Наприклад, набір даних Imagenet, що використовується як основа для трансферного навчання в інженерній практиці [42], має значний відсоток помилкових анотацій, через те, що в ньому на одному зображені може знаходитися декілька об'єктів різних класів, але мітка є тільки для одного. Автори [86] оцінюють рівень помилок та пропонують варіант для його автоматичної корекції через розбиття зображень з декількома об'єктами на декілька зображень з одним об'єктом.

Також, в задачах сегментації, помилки в анотаціях можуть бути упередженими [87] відповідно до людей-анотаторів, кожний з яких розмічає зображення певною мірою суб'єктивно. Особливо частою ця проблема є в

задачах медичної сегментації - в деяких прикладних областях навіть один і той самий анотатор може давати різну розмітку одних і тих самих зображень, в залежності від власного стану [88].

Для того, щоб оцінити приблизний рівень помилок в складних умовах для людей-анотаторів, автори [89] створили спеціальний набір даних, що складається з пар класів об'єкти яких мають схожі візуальні ознаки, наприклад, кіт та рись, або вовк та койот. Після розмітки прикладів людьми, автори використали перехресну перевірку для оцінки ймовірності зміни мітки класу в парах, яка вийшла рівною 6.44%. Одночасно з набором даних, автори також запропонували метод відновлення правильних анотацій SELFIE.

2.1.2. Введення помилок в анотації наборів даних. В процесі тестування різних методів машинного навчання в умовах частково-помилкових наборів даних виникає потреба в тестових наборах даних, що мають достовірні анотації, в той час, як наявні набори даних мають частково-помилкові анотації. Оскільки отримання таких наборів даних є практично неможливим, основним варіантом тестування методів машинного навчання є перехресна перевірка моделей. Однак, використання перехресної перевірки з нейромережевими методами ускладнюється високими витратами часу та обчислювальних ресурсів, особливо для великих моделей нейронних мереж.

Тому, важливим методом використання наборів даних з достовірними анотаціями є штучне введення помилок до оригінальних, якісних анотацій тільки для тренувального набору даних.

Наприклад, в роботі [85] запропоновані методи додавання помилок на основі зазначененої вище класифікації. Для всіх класів в наборі даних складається матриця невідповідностей N , так що:

Для рівномірних помилок:

$$N_{ij} = \begin{cases} p & \text{якщо } i = j \\ \frac{1-p}{M-1} & \text{якщо } i \neq j \end{cases}$$

де M - кількість класів.

Для помилок, що залежать від класу автори пропонують використовувати матрицю невідповідностей, що створюється попередньо навченою нейронною мережею.

Для помилок, що залежать від ознак, в роботі запропоновано використання претренованої нейронної мережі в якості екстрактора ознак, з подальшою іх кластеризацією та побудовою матриці невідповідностей.

Так само, можна використовувати зазначені методи і для задачі сегментації, змінюючи мітки класу для всього об'єкту в цілому, однак, це не відповідає більшості реальних помилок в задачі сегментації.

Окрім зміни міток класів для об'єктів в цілому, маски семантичної сегментації можуть не повністю покривати об'єкти, або частково не відповідати їх контурам. Для моделювання цих недоліків анотацій, в роботі [87] запропоновано введення помилок як неупереджено так і упереджено відповідно:

- еластична трансформація масок для кожного з об'єктів;
- константний зсув маски відносно об'єкта для всіх зображень.

Такі модифікації можна розглядати як окремий випадок помилок, що, залежного від ознак для задачі класифікації, якщо розглядати сегментацію як задачу піксельної класифікації.

Також, окремим випадком рівномірних помилок в анотаціях є відсутність масок сегментації для певних об'єктів, або їх присутність в місцях, де немає об'єктів. В такому разі помилки виникають між класами об'єктів та класом фону.

2.2. Параметрична модель набору даних із частково-помилковими анотаціями

На основі проведеного аналізу виявлено, що основними проблемами в анотації для задачі сегментації планарних зображень в задачах скринінгу є:

- Маски сегментації, що захоплюють сусідні з об'єктом пікселі;
- Маски сегментації, що покривають об'єкт не повністю;
- Відсутні маски сегментації для деяких об'єктів;
- Присутні зайні маски на місцях, де немає об'єктів.

Для відображення означених помилок в анотаціях, запропоновано штурчне введення помилок за допомогою випадкового застосування морфологічних операцій ерозії та дилатації з квадратним ядром до масок сегментації окремих об'єктів перед додаванням їх до загальної маски.

Операції ерозії та дилатації відповідають неупередженим помилкам, оскільки за їхнім результатом не можна відновити вихідну конфігурацію маски [87]. Також, означені операції реалізують помилки, що залежать від ознак, оскільки нерівномірно модифікують тонкі та більш товсті лінії, а також порожнини та опуклості.

На відміну від геометричних перетворень маски, включення, або невключення пікселів з граничних регіонів маски є частішим випадком помилок в анотаціях задачі сегментації.

При створенні параметричної моделі наборів даних враховується наступне:

- В задачах скринінгу об'єкти часто не мають чітких контурів, а також їх текстурне забарвлення може зливатися з текстурою фону, або об'єктів інших класів, тому модель набору даних має мати можливість відтворення об'єктів з нечіткими контурами та з текстурами, що частково співпадають з фоном.

- Методи локалізації та класифікації в задачах автоматизованого скрінінгу об'єктів використовують як текстурну інформацію, так і інформацію про границі, тому модель набору даних має мати можливість представлення на зображеннях різних об'єктів з різними характеристиками як границь, так і текстур.
- Об'єкти в задачах автоматизованого скрінінгу можуть мати різноманітні форми навіть в межах одного класу, тому модель набору даних має мати можливість моделювання об'єктів одного класу, що мають різноманітні зовнішні характеристики.

Запропонована модель набору даних \mathcal{M} має наступне представлення:

$$\mathcal{M} \in \{\mathcal{X}_b, \mathcal{X}_{tex}, N, S_{img}, S_{obj}, \Delta_{max}, N_{obj}, P_e, P_d, S_e, S_d\} \quad (2.1)$$

де \mathcal{X}_b - набір зображень фону,

\mathcal{X}_f - набір об'єктів,

\mathcal{X}_{tex} - набір зображень текстур об'єктів,

N - кількість зображень в генерованому наборі даних,

S_{img} - розмір генерованих зображень в пікселях,

S_{obj} - середній розмір об'єкта в пікселях,

Δ_{max} - максимальне відхилення розміру об'єкта в відсотках,

N_{obj} - максимальна кількість об'єктів на зображені,

P_e та P_d - ймовірності зменшення та збільшення маски кожного з об'єктів,

S_e та S_d - допустимі масштаби збільшення та зменшення масок всіх об'єктів.

Останні чотири параметри введено для контролюваного створення помилок в анотаціях.

В якості наборів даних об'єктів можуть бути використані зображення з наборів даних MNIST [90], або FashionMNIST [91].

Використання зазначених наборів даних зумовлено наступними фактами:

- Можливість простого відділення об'єкта від вихідного фону для створення анотації сегментації;
- Наявність схожих елементів з різних класів (наприклад, цифри 1 та 7, або класи *T-shirt* та *Dress*), що дозволяє використати ці об'єкти як схожі представники різних класів, особливо при використанні схожих текстур для їх забарвлення;
- Наявність різноманітних за формою об'єктів в межах одного класу (наприклад, різні варіанти написання цифр, або різний зовнішній вигляд предметів одягу одного класу)
- Висока точність роботи сучасних нейронних мереж на цих наборах даних, що дозволяє сконцентруватися на впливі помилок в розмітці, замість розпізнавання безпосередньо об'єктів

Для створення текстур об'єктів може бути використаний будь-який набір натуральних зображень, наприклад, Imagenet [42], або його підвибірка Imagenette [92]. Використання набору реальних зображень дозволяє забезпечити різноманіття текстур об'єктів та фону. Для забезпечення схожості текстур об'єктів та фону, можливе застосування одного і того ж набору зображень для генерації текстур як фону, так і об'єктів. Для того, щоб уникнути повного збігу текстур об'єктів та фону, для них використовуються різні зображення з одного набору даних що забезпечується методом генерації наборів даних.

2.3. Метод генерації наборів даних

Запропоновано метод контролюваної генерації як зображень, так і масок сегментації що модифікуються відповідно до випадково обраних помилок.

На основі параметричної моделі генеруються навчальний та тестовий набори даних, всі зображення в наборах генеруються незалежно.

Контрольована генерація зображень виконується в два етапи: генерація фону та розташування на ньому довільної кількості об'єктів. Для генерації фону можуть бути використані як звичайні натуруальні зображення, так і синтетичні текстури, чи заливка константним кольором.

На основі запропонованої параметричної моделі (2.1) розроблено метод генерації наборів даних із частково-помилковими анотаціями, який містить наступні кроки:

1. Вибрати випадкове зображення фону: $x_{bg} \sim \mathcal{X}_b$
2. Вибрати кількість об'єктів на зображенні: $N_{obj} \sim \mathcal{U}(1, N_{obj})$
3. Провести ініціалізацію маски сегментації: $M = 0$ так що $M \in \mathcal{R}^{C \times S_{img} \times S_{img}}$
4. Виконати наступні кроки n_{obj} разів:
 - 4.1 Вибрати розміри об'єкта: $s \sim \mathcal{U}(S_{obj} - \Delta_{max}, S_{obj} + \Delta_{max})$
 - 4.2 Вибрати координати розміщення об'єкта:

$$i_f \sim \mathcal{U}(0, S_{img} - s)$$

$$j_f \sim \mathcal{U}(0, S_{img} - s)$$
 - 4.3 Вибрати зображення об'єкта $x_{fg} \sim \mathcal{X}_f$ та відповідний клас об'єкта $c_{fg} \sim \mathcal{Y}_f$
 - 4.4 Змінити розмір зображення об'єкта за допомогою білінійної інтерполяції:

$$\hat{x}_{fg} = R_{bilinear}(x_{fg})$$
 - 4.5 Вибрати зображення текстири $x_{tex} \sim \mathcal{X}_{tex}$
 - 4.6 Модифікувати зображення об'єкта за допомогою текстири:

$$\hat{x}_{fg} = x_{fg} \circ x_{tex}[i_f : i_f + s, j_f : j_f + s]$$

4.7 Розмістити зображення об'єкта на зображені фону:

$$x_{bg}[i_f : i_f + s, j_f : j_f + s] = (1 - x_{fg}) \circ x_{bg} + \hat{x}_{fg}$$

4.8 Сформувати маску сегментації об'єкта:

$$M_{seg} = x_{fg} > \theta_{seg}$$

де θ_{seg} - поріг бінаризації вихідного зображення об'єкта. Для набору даних MNIST $\theta_{seg} = 0.2$, для набору даних FashionMNIST $\theta_{seg} = 0.1$.

4.9 Модифікувати маску сегментації відповідно до необхідного рівня помилок:

$$M_{seg} = \begin{cases} M_{seg} \oplus K^{S_d \times S_d} & \text{якщо } p_d \sim \mathcal{U}(0, 1) < P_d \\ M_{seg} \ominus K^{S_e \times S_e} & \text{якщо } p_e \sim \mathcal{U}(0, 1) < P_e \end{cases}$$

де $K^{S_d \times S_d}$ - матриця ядра $K^{S_e \times S_e}$ - матриця ядра ерозії.

4.10 Розмістити модифіковану маску сегментації об'єкта на загальному зображенні маски сегментації:

$$M[c_{fg}, i_f : i_f + s, j_f : j_f + s] = \max \{M[c_{fg}, i_f : i_f + s, j_f : j_f + s], M_{seg}\}$$

4.11 Зберегти зображення x_{bg} та маску M

5. Завершити генерацію

Завдяки такому варіанту генерації, можливо отримання безлічі наборів даних із схожими характеристиками, що дозволяє використовувати непараметричні статистичні методи, такі як бутстрепінг для оцінки моделей. На практиці, для генерації різних наборів даних, достатньо зміні ініціалізації генератора випадкових чисел.

Проведено аналіз впливу розмірів та форми ядер ерозії на достовірність генерованої розмітки. Достовірність визначено як міру Дайса-Соренсена між вихідною генерованою розміткою та результатом операцій ерозії та дилатації над цією розміткою. Графіки залежності зображені на рисунку 2.1.

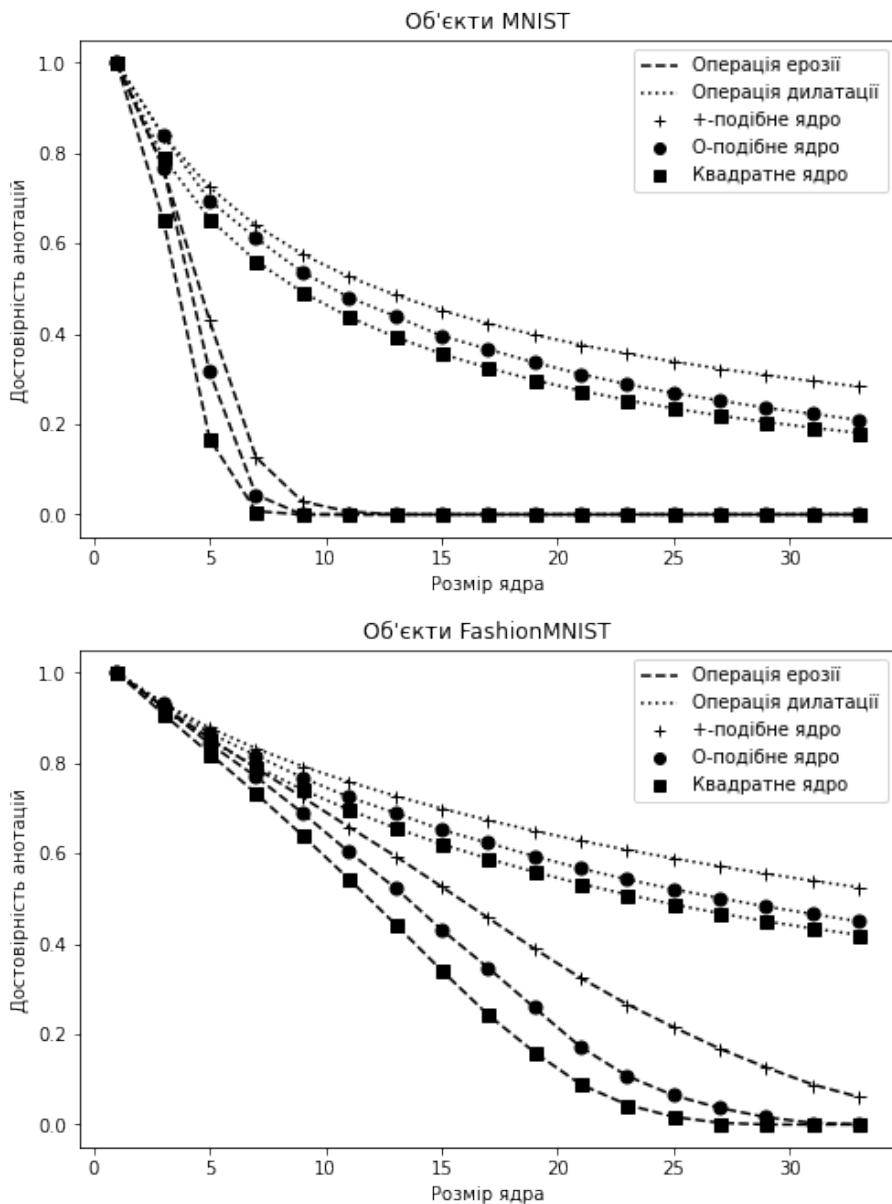


Рис. 2.1: Залежність достовірності анотацій від розміру та форми ядер ерозії та дилатації

З зазначеного графіку видно, що достовірність нелінійно залежить від кількості ненульових елементів ядра морфологічних операцій, а також від розміру ядра. Також, зазначено, що набір даних об'єктів MNIST є більш схильним до зниження достовірності через введення помилок в анотації через більшу кулькість тонких елементів, що зникають під впливом операції ерозії.

Означена залежність дозволяє оцінити рівень помилок анотацій та вра-

ховувати його при генерації наборів даних для тестування моделей машинного навчання.

2.4. Види згенерованих зображень

В даному розділі продемонстровано згенеровані зображення та відповідні маски сегментації для різних параметрів моделі.

За допомогою моделі можна генерувати набори даних з досить різноманітними характеристиками. На рисунках 2.2 та 2.3 зображено найпростіший випадок для генерації за допомогою моделі: заповнення фону та об'єктів константними значеннями (0 і 1) відповідно.

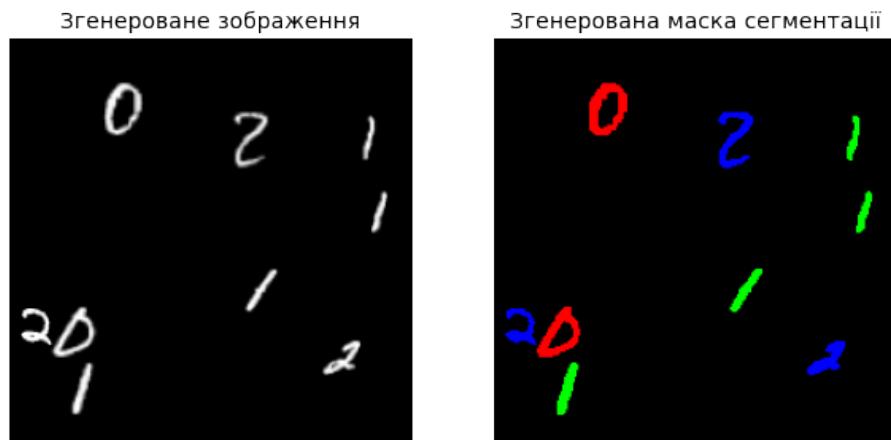


Рис. 2.2: Об'єкти MNIST, фон: константний, колір об'єктів: білий

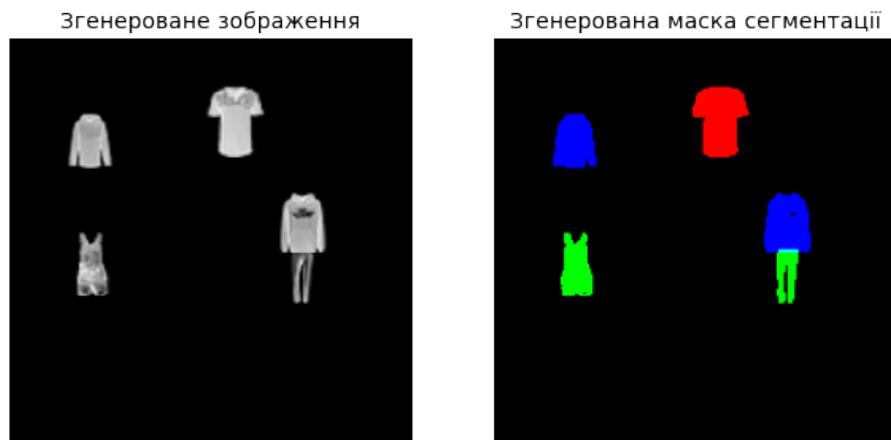


Рис. 2.3: Об'єкти FashionMNIST, фон: константний, колір об'єктів: білий

Такі генеровані зображення можуть бути використані при тестуванні базової достовірності прогнозів моделей на синтетичних даних для отримання показового максимального рівня достовірності, який може досягти тестована модель та метод її навчання в ідеальних умовах. Насамперед, використання простих зображень корисне для валідації моделей класичного комп'ютерного зору для спрощення відбору ознак зображення для подальшого використання в задачах класифікації.

На рисунках 2.4 та 2.5 зображено фон з використанням зображення з набору даних Imagenette та заповнення об'єктів константним значенням 1.



Рис. 2.4: Об'єкти MNIST, фон: Imagenette, колір об'єктів: білий

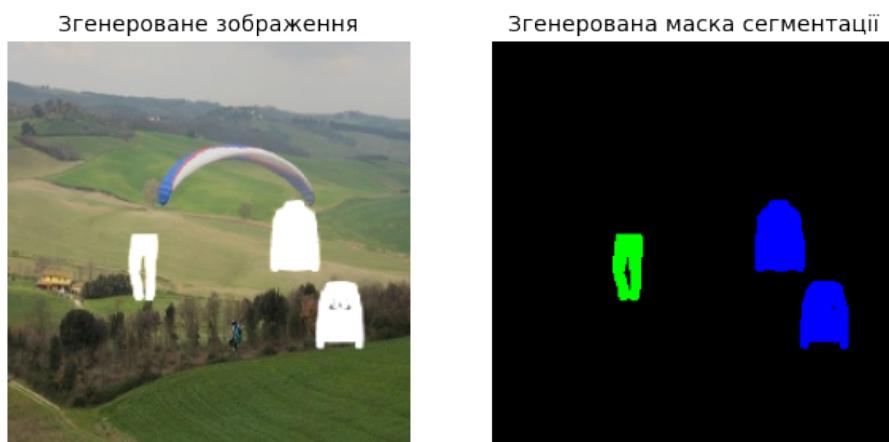


Рис. 2.5: Об'єкти FashionMNIST, фон: Imagenette, колір об'єктів: білий

Такий варіант генерованих зображень доцільно використовувати при

тестуванні моделей та методів машинного навчання у випадках, коли необхідна локалізація нетекстурованих об'єктів на текстурному фоні, або таких об'єктів, в яких головними ознаками для сегментації мають бути контури об'єктів. Також, даний варіант генерації зображень може містити фонові контури, що схожі на контури об'єктів, що є характерним для задач автоматизованого скринінгу в рентгенології (наприклад, флюорограми).

На рисунках 2.6 та 2.7 зображено фон з використанням зображення з набору даних Imagenette та заповнення об'єктів константним значенням 0.

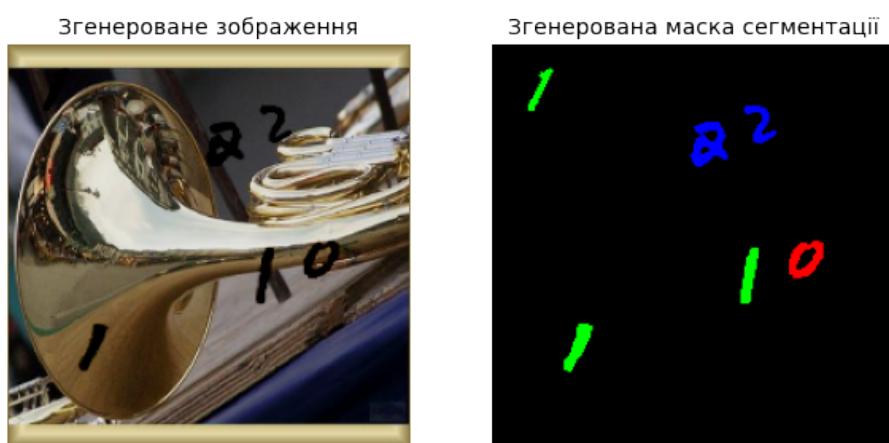


Рис. 2.6: Об'єкти MNIST, фон: Imagenette, колір об'єктів: чорний

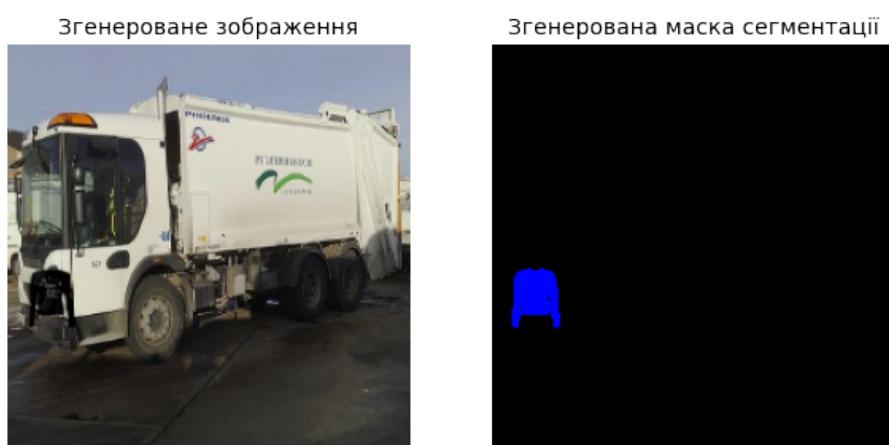


Рис. 2.7: Об'єкти FashionMNIST, фон: Imagenette, колір об'єктів: чорний

Як і попередній варіант генерованих зображень, цей варіант доцільно використовувати при тестуванні моделей, які мають працювати з нетексту-

рорваними об'єктами. На відміну від попереднього варіанту, в даному випадку колір об'єктів перетинається з кольором тіней на фоні, що відповідає таким задачам, як скринінг на основі зрізів комп'ютерних томограм. Це дозволяє використовувати запропоновану модель для тестування достовірності методів сегментації об'єктів, схожих з елементами фону.

На рисунках 2.8 зображені найскладніший варіант набору даних: як фон, так і текстури об'єктів взято з набору даних Imagenette.



Рис. 2.8: Об'єкти MNIST, фон: Imagenette, текстури об'єктів: Imagenette



Рис. 2.9: Об'єкти FashionMNIST, фон: Imagenette, текстури об'єктів: Imagenette

Такий варіант генерованих зображень відповідає більшості задач автоматизованого скринінгу, оскільки об'єкти відділяються від фону як за

допомогою текстур, так і за допомогою контурів. При такій генерації можливі випадки об'єктів одного класу з повністю різними текстурами, а також різні класи об'єктів з однаковими текстурами, що відповідає різномініттю даних в задачах автоматизованого скринінгу. Також, на рисунку 2.9 внизу справа продемонстровано об'єкт, текстура якого лише незначно вирізняється від текстури фону

Для наочності, **модифікації масок сегментації** показано на простих зображеннях з константним кольором фону та об'єктів.

На рисунку 2.10 зображено вихідне зображення, маску сегментації з еrozією ядром 3×3 та їх комбінацію для наочності.



Рис. 2.10: Об'єкти MNIST, еrozія маски ядром 3×3

На рисунку 2.11 зображено вихідне зображення, маску сегментації з дилатацією ядром 3×3 та їх комбінацію для наочності.



Рис. 2.11: Об'єкти MNIST, дилатація маски ядром 3×3

На рисунку 2.12 зображене вихідне зображення, маску сегментації з дилатацією та ерозією ядром 3×3 та їх комбінацію для наочності.



Рис. 2.12: Об'єкти MNIST, ерозія та дилатація маски ядром 3×3

Також, при виборі розмірів ядер ерозії потрібно бути уважним до вихідних масок - дуже великі ядра ерозії можуть знищити маску сегментації для об'єкта. На рисунку 2.13 зображене вихідне зображення, маску сегментації з ерозією ядром 7×7 та їх комбінацію для наочності.

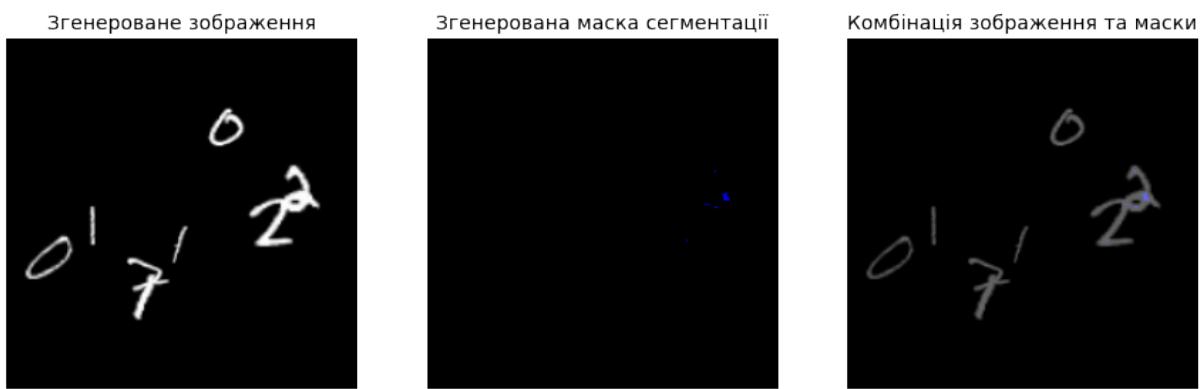


Рис. 2.13: Об'єкти MNIST, ерозія маски ядром 7×7

Таким чином, виконується генерація частково-помилкових анотацій для зображень в повністю контролюваних умовах, що дозволяє використовувати запропоновані модель та метод для тестування моделей та методів машинного навчання.

2.5. Висновки до другого розділу

У другому розділі представлено параметричну модель наборів даних із частково-помилковою розміткою для задач сегментації та класифікації, що відповідає оцінкам моделей помилок в різних задачах автоматизованого скринінгу, а також метод контролльованої генерації як зображень, так і масок сегментації та міток класів що модифікуються відповідно до випадково обраних недоліків.

Продемонстровано різні варіанти генерації зображень, показано аналогічність до даних в задачах автоматизованого скринінгу. Показано варіанти зашумлення розмітки для простих даних для наочності.

Запропоновано використання модельних наборів даних тестування достовірності моделей та методів навчання глибоких нейронних мереж в задачах класифікації та семантичної сегментації. Тестування виконується в контролльованих умовах за допомогою введення помилок в розмітку лише тренувального набору даних, в той час як тестувальних набірів даних залишається з точною розміткою.

РОЗДІЛ 3

НЕЙРОМЕРЕЖЕВА МОДЕЛЬ АНАЛІЗУ ПЛАНАРНИХ ЗОБРАЖЕНЬ ТА МЕТОДИ ЇЇ НАВЧАННЯ

Запропоновано модель нейронної мережі та метод багатозадачного навчання для одночасного підвищення достовірності класифікації та сегментації без зниження оперативності, а також метод її навчання, що заснований на методах багатозадачного навчання.

На відміну від однозадачного машинного навчання, багатозадачне дозволяє використати спільноті та відмінності між різними задачами для підвищення узагальнювальної здатності на кожній з окремих задач. Використання комбінації декількох задач машинного навчання може сприяти вивчення кращих наборів ознак, та підвищити точність роботи нейронних мереж. Також, сумісне використання прогнозів декількох задач для уточнення кожного з окремих прогнозів, так і побудови нових.

Основним припущенням, на яке спирається розроблений метод є можливість створення достовірної розмітки тренувального набору даних для задачі класифікації на основі розмітки семантичної сегментації для того ж набору вхідних зображень. В даному контексті, задача семантичної сегментації називається основною задачею, а класифікації - додатковою.

3.1. Структурна модель багатозадачної штучної нейронної мережі

Для того, щоб основна і додаткова задачі оновлювали параметри нейронної мережі, що відповідають за виділення ознак, використано схему багатозадачного навчання з жорстким розподілом параметрів.

3.1.1. Загальна структура моделі нейронної мережі. Нейронна мережа виконана з використанням архітектури енкодер-декодер, заснована на архітектурі нейронних мереж LinkNet [55]. На відміну від оригінальної структури LinkNet, запропонована модель нейронної мережі складається з одного енкодера та двох декодерів: для задач сегментації та класифікації відповідно.

Модель представлено аналітично: нехай нейронну мережу енкодера визначено як $F_{encoder}$. Тоді для вхідного зображення $x \in \mathcal{R}^{3 \times H \times W}$, v_1, v_2, \dots, v_n - набір карт ознак, так що $v_i \in \mathcal{R}^{c_i \times \frac{H}{i} \times \frac{W}{i}}$, де c_i - кількість каналів для кожної з карт ознак, а n - кількість стадій енкодера (залежить від архітектури):

$$v_1, v_2, \dots, v_n = F_{encoder}(x, \theta_{enc}) \quad (3.1)$$

де θ_{enc} - набір параметрів енкодера.

Тоді, нейронні мережі декодера сегментації та класифікації можна визначити як F_{seg} та F_{cls} відповідно. Для набору карт ознак, що генерує енкодер, маємо:

$$M_{seg} = F_{seg}((v_1, v_2, \dots, v_n), \theta_{seg}) \quad (3.2)$$

$$C_{cls} = F_{cls}((v_n), \theta_{cls}) \quad (3.3)$$

де θ_{seg} та θ_{cls} - набори параметрів декодерів сегментації та класифікації відповідно.

В ролі енкодера можуть бути використані наявні багатостадійні архітектури, такі як VGGNet, ResNet, EfficientNet та ін. Карти ознак після кожного етапу просторового зменшення використовуються як входи для декодера сегментації, для декодера класифікації використовується карта ознак з найглибшого шару енкодера. Узагальнену структуру моделі зображенено на рисунку 3.1

Таким чином, виконується жорсткий розподіл параметрів, оскільки параметри енкодера відповідають одночасно за обидві задачі. Така структура

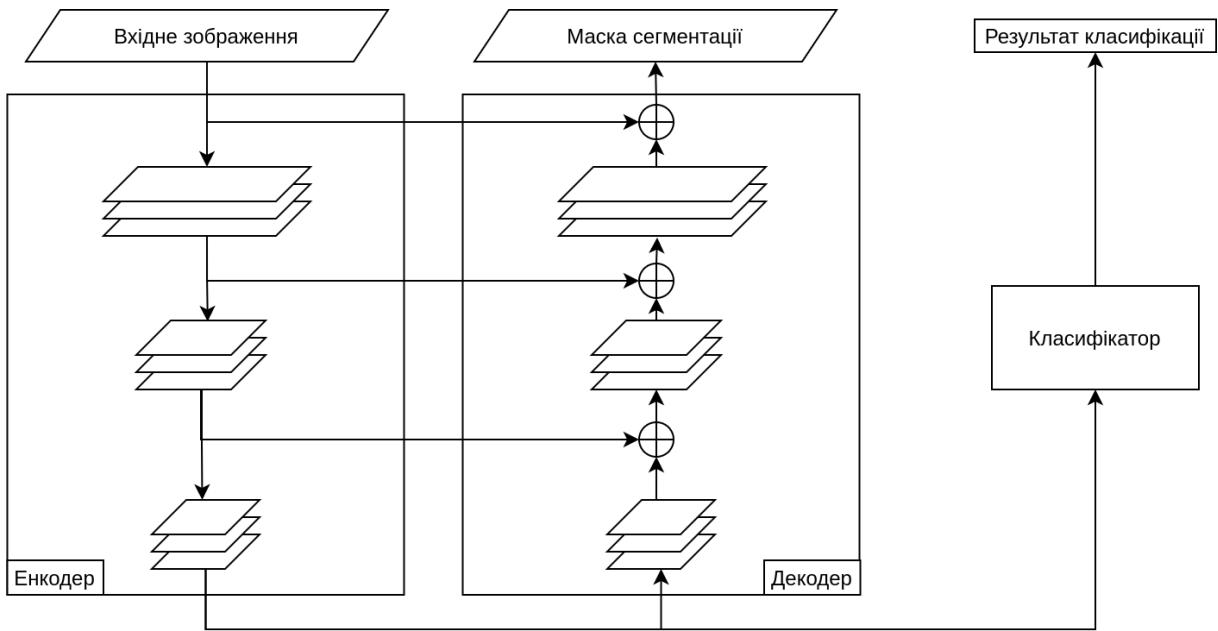


Рис. 3.1: Загальна архітектура нейронної мережі з декодером та класифікатором

дозволяє використати методи трансферного навчання (індуктивного переносу) для підвищення роздільності внутрішніх представлень і пришвидшення процесу навчання: набір параметрів енкодера θ_{enc} ініціалізується з використанням параметрів, отриманих після навчання енкодера на наборі даних Imagenet [42]. Декодери ініціалізуються з використанням методу Хе: $\theta \in \mathcal{U}(-b, b)$, де b - константа залежна від типу шару [93]

Окрім загальної архітектури моделі, критичними для коректної роботи запропонованих методів є структури декодерів дляожної з задач, зокрема, нормалізація їх внутрішніх представлень.

3.1.2. Структура декодера семантичної сегментації. Декодер складається з декількох стадій, кожна з яких має розміри карт ознак відповідно до карт ознак енкодера.

Структура декодера сегментації повторює декодер архітектури LinkNet. На відміну від оригінальної структури, в запропонованому декодері застосовується пакетна нормалізація після операцій конкатенації з шарами

енкодера та перед згортковими шарами. Ця нормалізація необхідна для забезпечення незалежності значень інтенсивності карт ознак на різних шарах декодера сегментації та енкодера, які також спільно використовуються декодером класифікації.

Також, замість зворотних згорток, в декодері використовується білінійна інтерполяція для підвищення просторового розширення карт ознак, що дозволяє уникнути "шахового патерну" в прогнозованих масках [94].

Оскільки в задачі сегментації з частково-помилковою розміткою одному пікселю може бути призначено декілька класів одночасно, необхідно це враховувати при використанні функцій втрат. Для формування фінальної карти сегментації використовується спеціальна версія блоку декодера, що має сигмоподібну функцію активації. Це дозволяє нейронній мережі прогнозувати одночасно декілька класів для одного пікселя, замість спроб виділення одного конкретного класу при використанні нормованої експоненційної функції (Softmax).

Структуру окремих стадій показано на рисунку 3.2.

В ролі вхідної карти ознак для найпершого шару декодера використовується останній шар енкодера v_n .

3.1.3. Структура декодера класифікації. Оскільки, в даному випадку, на зображенні може бути декілька об'єктів, а навчання в задачі класифікації відбувається набором зразків, декодер класифікації має враховувати як глобальний просторовий контекст, так і локальні ознаки, притаманні об'єктам. Для цього, першим шаром класифікатора є конкатенація результатів двох операцій глобальної підвибірки: з операцією усереднення (GlobalAvgPooling), та вибору максимуму (GlobalMaxPooling) для кожного з каналів карти ознак енкодера.

Для забезпечення незалежності інтенсивності останнього шару енкодера при навчанні двох декодерів одночасно, першим шаром є шар пакетної

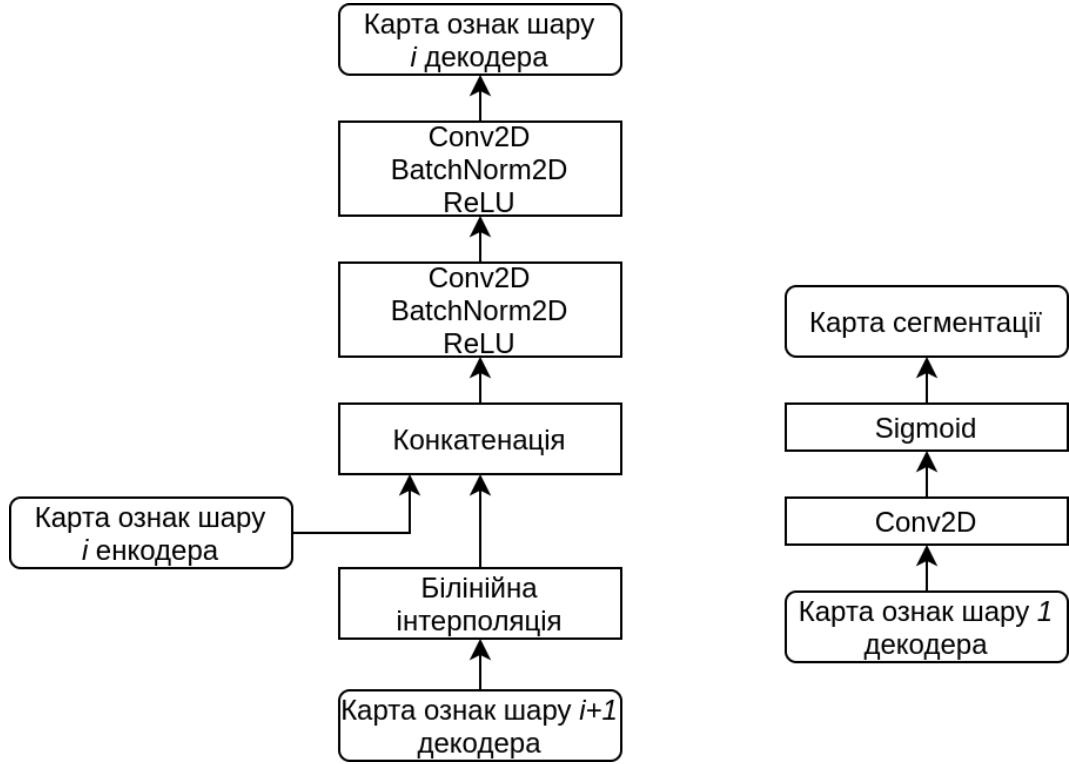


Рис. 3.2: Структура стадій декодеру сегментації

нормалізації.

Для підвищення роздільності ознак останнього шару енкодера, класифікатор являє собою лінійну функцію, що спонукає енкодер до формування лінійно-роздільних представлень v_n [47], які також використовуються декодером сегментації. Застосування такої структури декодера класифікації дозволяє знизити необхідну кількість нелінійних перетворень, потрібних декодеру сегментації.

Структура декодера класифікації є важливою складовою успішного навчання при використанні обмеженої функції втрат в задачі класифікації. Основними задачами декодеру класифікації є провадження градієнтів до енкодера у випадку фільтрації неправильної розмітки в задачі сегментації, а також покращення роздільності ознак найглибшого шару енкодера.

Структура декодера класифікації зображена на рисунку 3.3.

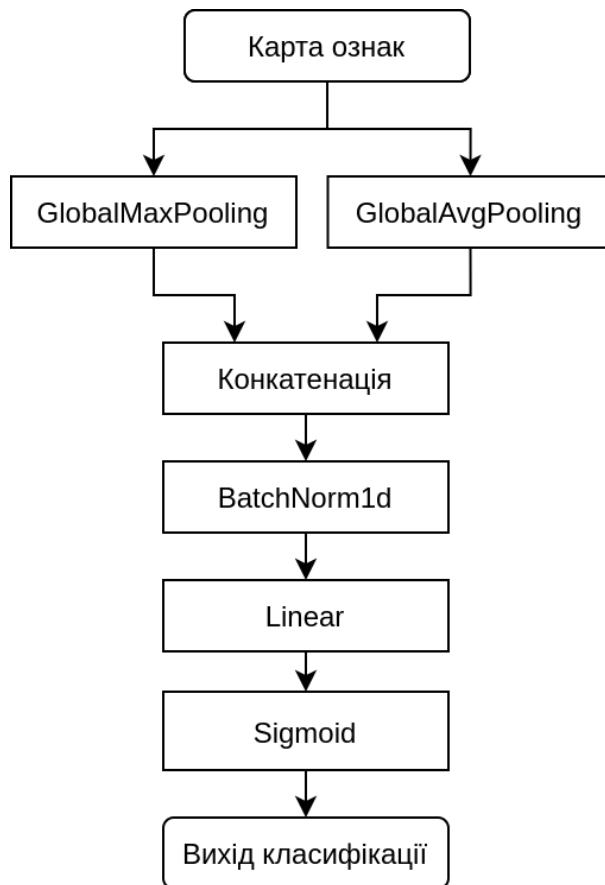


Рис. 3.3: Структура декодера класифікації

3.2. Методи навчання та прогнозування багатозадачних ШНМ в умовах частково помилкової розмітки сегментації

На основі запропонованої моделі багатозадачної нейронної мережі розроблено метод багатозадачного навчання, що в умовах частково-помилкової розмітки дозволяє зменшити вплив неправильно розмічених прикладів на процес навчання за рахунок їх ефективної фільтрації безпосередньо під час навчання. Так, часто на одному зображені можуть бути розташовані як коректно-розмічені, так і некоректно-розмічені об'єкти. Для підвищення достовірності нейронних мереж в таких умовах, запропоновано використати метод багатозадачного навчання з використанням похідної більш загальної задачі.

Запропонований метод складається з двох етапів:

3.2.1. Генерація похідної задачі до задачі семантичної сегмен-тації. Для того, щоб забезпечити успішне навчання нейронних мереж на ранніх етапах, та врахувати занадто складні приклади, необхідно ввести інформацію про ці об'єкти в обхід відфільтрованої розмітки.

На основі розмітки для задачі семантичної сегментації може бути побудовано розмітку для інших більш загальних задач, таких як задача детекції (наприклад, обмежувальні прямокутники навколо об'єктів), регресії (наприклад, координати центрів об'єктів), або класифікації (класи об'єктів). Такі задачі є більш зашальними, оскільки мають менше незалежних змінних в виходах нейронної мережі. Наприклад, замість класифікації кожного з пікселів зображення в задачі семантичної сегментації, в задачі детекції потрібно лише прогнозування класів та координат обмежувальних прямокутників.

Також, з точки зору багатозадачного машинного навчання, ці задачі є семантично-близькими: такі задачі, що мають одинакові вхідні дані, та пов'язані вихідні дані [65].

На відміну від [95], що спирається на вивчення більш детальних семантично-близьких задач, в розробленому методі використання більш точних даних, для загальніших задач дозволяє покращити роздільність внутрішніх представлень нейронної мережі, що, в свою чергу, покращує результати на вихідній задачі. Також, оскільки задачі є семантично близькими, не відбувається конфлікту градієнтів, що є типовим при навчанні семантично-різномірних задач [73].

Оскільки задача семантичної сегментації може бути розглянута як задача піксельної класифікації, можна розглядати задачу класифікації всього зображення як більш загальну до неї. В такому контексті, класифікація буде окремим випадком навчання за набором зразків [96]: замість розмітки

кожного з об'єктів для всіх класів на зображенні, зображення являє собою мішок з одним, чи декількома об'єктами та відповідним маркуванням, чи є об'єкти заданих класів на ньому.

Попередній аналіз показав, що неточна розмітка в задачах сегментації полягає в наявності зайвих, або у відсутності деяких розмічених пікселів, або об'єктів. Наявність такої неточної розмітки дозволяє створити на її основі точну розмітку для класифікації.

Нехай для зображення $x \in \mathcal{R}^{3 \times H \times W}$ існує маска сегментації $y_s \in \mathcal{R}^{C \times H \times W}$ де C - кількість класів, а H та W висота та ширина зображення відповідно. Якщо в масці сегментації є хоча б один розмічений об'єкт класу, встановлюється мітка відповідного класу $y_c \in (0, 1)^C$ в розмітці задачі класифікації:

$$y_c = t < \sum_i^H \sum_j^W y_{s_{ij}} \quad (3.4)$$

де t - поріг мінімального розміру об'єкта в пікселях

Згенерована таким чином задача класифікації має меншу кількість помилкових анотацій.

В такому випадку, обмеження функції втрат використовується лише для оригінальної задачі, а більш загальна додаткова задача використовує оригінальну функцію втрат без обмеження. Таким чином, завжди існують градієнти від більш загальної розмітки, що спонукають навчання на прикладах, для яких немає градієнтів через обмеження функції втрат.

3.2.2. Автоматична фільтрація помилкової розмітки. Маючи велику кількість параметрів, сучасні нейронні мережі здатні до запам'ятовування тренувального набору даних замість вивчення корисних ознак для узагальнення на інші набори даних (перенавчання). В умовах наявності помилкової розмітки в тренувальному наборі даних, нейронні мережі схильні до

вивчення і помилок розмітки на протязі навчання.

Традиційні методи запобігання перенавчанню, такі як аугментація даних [97], регуляризація ваг мережі [98], та завчасна зупинка навчання [99] хоча і показують покращені результати, але можуть виявитися недостатніми в умовах великої ймовірності помилок в навчальних даних. Також, вони можуть погіршити якість навчання через зменшення ємності нейронної мережі, або, при надмірному використанні, можуть спонукати нейронну мережу акцентувати увагу на текстурах [100], що часто є недопустимим в задачах семантичної сегментації.

Для того, щоб ефективно зменшити вплив помилкової розмітки, необхідно вилучити її з процесу навчання нейронної мережі. Якщо неможливо вилучення неправильно розмічених прикладів з набору даних до початку навчання, це можна зробити ітеративно в процесі навчання, для чого потрібно визначити критерії вилучення.

Такі функції втрат, як перехресна ентропія, або фокальна функція втрат призначають експоненційно високе значення для неправильних прогнозів нейронних мереж, що має шкідливий ефект при наявності помилкової розмітки в тренувальному наборі даних - правильні прогнози на неправильнорозмічених даних створюють конфліктні градієнти та знижують впевненість прогнозів нейронної мережі.

Для виключення неправильної розмітки в процесі навчання запропоновано введення обмеження другого роду (зверху) для функції втрат, з відповідним визначенням градієнтів для отриманої нової функції:

$$\mathcal{L} = \min(L, \theta) \quad (3.5)$$

де θ - поріг обмеження функції втрат.

Графіки обмежених зверху функцій втрат зображені на рисунку 3.4.

Оскільки, для функції обмеження зверху градієнт визначено лише на

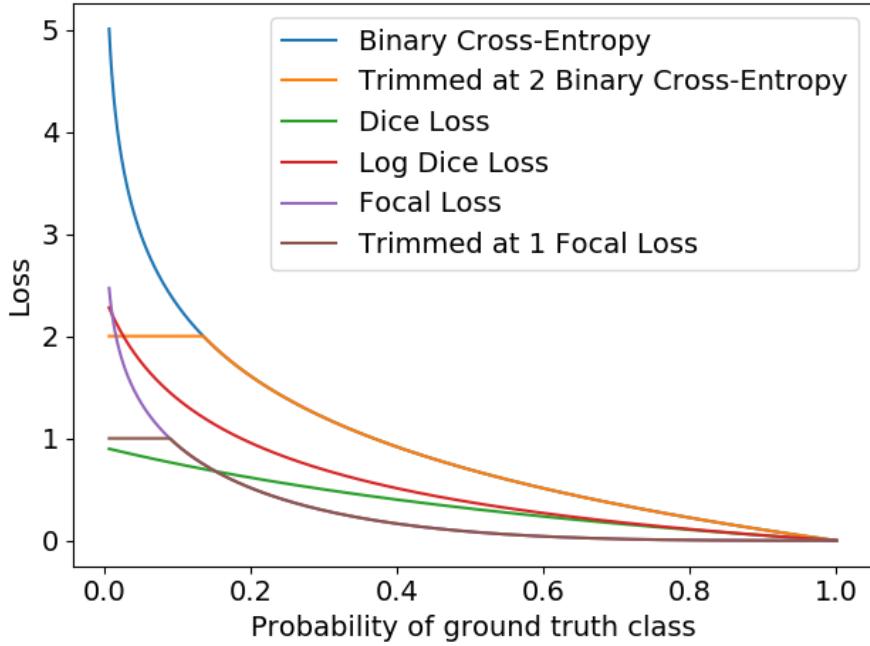


Рис. 3.4: Приклади обмежених зверху функцій втрат

проміжку $(-\inf, \theta]$, тому для проміжку (θ, \inf) , в контексті даної задачі, градієнт запропоновано визначити рівним нулю:

$$\nabla \min(L, \theta) = \begin{cases} 1 & L \in (-\inf, \theta] \\ 0 & L \in (\theta, \inf) \end{cases} \quad (3.6)$$

Всі загальновживані методи оптимізації параметрів нейронних мереж засновані на методі оновлення параметрів першого порядку, що, в загальному випадку, є модифікацією стохастичного градієнтного спуску:

$$\theta \leftarrow \theta - \eta \nabla_{\theta} L \quad (3.7)$$

де θ - параметри нейронної мережі, η - темп навчання, а $\nabla_{\theta} L$ - градієнт функції втрат відносно параметрів, таким чином, приклади з занадто великою похибкою (переважно з помилковою розміткою) ефективно виключаються з навчання, оскільки мають нульовий градієнт відносно параметрів.

Недоліком цього методу фільтрації помилкової розмітки є можливість виключення прикладів, що занадто складні для вивчення нейронною мережею на ранніх етапах навчання. Це може сповільнити процес навчання, або

взагалі зупинити його. Щоб запобігти сповільненню процесу, запропоновано використання багатозадачного навчання з використанням додаткової більш загальної задачі.

Агрегація функцій втрат під час навчання. Оскільки модель багатозадачної нейронної мережі має два виходи дляожної з задач, для одночасного оновлення всіх параметрів необхідне одночасне використання двох функцій втрат. Дляожної з задач окремо обчислюється функція втрат. Для навчання декодера сегментації використовується обмежена зверху, в той час як для декодера класифікації - звичайна. Кожна з функцій втрат нормалізується відносно кількості прикладів в одному пакеті навчання для зниження впливу цього гіперпараметру.

Загальне значення функції втрат визначається як арифметичне середнє між індивідуальними значеннями:

$$L_{total} = \frac{L_{seg}] + L_{cls}}{2} \quad (3.8)$$

Відповідно, загальний градієнт функції втрат буде сумаю градієнтів складових частин:

$$\nabla L_{total} = \frac{\nabla L_{seg}] + \nabla L_{cls}}{2} \quad (3.9)$$

Під час навчання нейронної мережі використовується стандартний алгоритм зворотного поширення помилки з оптимізатором Adam [101], що відрізняється від методу стохастичного градієнтного спуску наявністю адаптивної нормалізації моментів градієнтів.

Поріг обмеження функції втрат для задачі сегментації є параметром алгоритму навчання та має обиратися емпірично в залежності від рівня помилок в розмітці. Таким чином, забезпечується "прохід" градієнтів для оновлення параметрів від хоча б однієї функції втрат для кожного вхідного прикладу.

3.2.3. Багатозадачне прогнозування результатів семантичної сегментації. Оскільки запропонована модель нейронної мережі може одночасно виконувати як задачу сегментації, так і задачу класифікації, стає можливою реалізація фільтрування хибно-позитивних ознак без зниження оперативності на етапі прогнозування результатів.

Для фільтрації хибно-позитивних результатів запропоновано метод комбінації задач класифікації та сегментації, що дозволяє використовувати задачу класифікації, що попередньо була навчена на менш зашумленій розмітці як фільтр для задачі семантичної сегментації. В такому випадку можливі два варіанти комбінації двох задач: послідовний і паралельний.

Нехай $C_{cls} \in \mathcal{R}^C$ та $M_{seg} \in \mathcal{R}^{C \times H \times W}$ - результати декодерів класифікації та сегментації відповідно, значення яких знаходяться на проміжку $(-\infty, +\infty)$ (логіти).

Для отримання результатів на проміжку $[0, 1]$ використовується логістична сигмоїдна функція активації:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (3.10)$$

Послідовний варіант полягає у відкиданні результатів сегментації, якщо результат класифікації менший, за деякий поріг t_{cls} :

$$M_{refined} = \begin{cases} \sigma(M_{seg}) & \text{якщо } \sigma(C_{cls}) > t_{cls} \\ 0^{C \times H \times W} & \text{якщо } \sigma(C_{cls}) \leq t_{cls} \end{cases} \quad (3.11)$$

Такий варіант має декілька суттєвих недоліків:

- Необхідність вибору додаткового параметра θ_{cls} вносить додаткову можливість помилки; неправильний підбір цього параметра може привести до значного збільшення хибно-негативних результатів.
- Можливість розповсюдження помилки класифікатора виникає через значно меншу кількість параметрів порівняно з декодером се-

гментації.

- Зниження оперативності через послідовне виконання двох нейронних мереж

Паралельний варіант полягає у зважуванні карти сегментації за допомогою нормованих логітів класифікатора. Першим кроком є трансформація логітів сегментації та класифікації в некалібровані оцінки на проміжку $[0, 1]$:

$$\hat{M}_{seg} = \sigma(M_{seg}) \quad (3.12)$$

$$\hat{C}_{cls} = \sigma(C_{cls}) \quad (3.13)$$

Ці оцінки мають ті самі розмірності, що й оригінальні маска та класи, для зручності репрезентації операцій додано додаткові розмірності до вектору класів: $\hat{M}_{seg} \in \mathcal{R}^{C \times H \times W}$ та $\hat{C}_{cls} \in \mathcal{R}^{C \times 1 \times 1}$

Зважування карти сегментації відбувається за допомогою добутку Адамара між матрицями \hat{M}_{seg} та \hat{C}_{cls}

$$M_{refined} = \hat{M}_{seg} \circ \hat{C}_{cls} \quad (3.14)$$

Графічну репрезентацію структури зображено на рисунку 3.5.

Паралельний варіант має перевагу над послідовним у відсутності додаткового параметру навчання. Також, зважені карти сегментації допомагають збереженню більшої кількості інформації для етапу пост-обробки масок сегментації.

Однак, в разі використання добутку двох сигмоїдних функцій, потрібно зважати, що змінюється центральна точка, і, відповідно поріг бінаризації в подальшій обробці. Навіть у випадку співпадіння значень класифікації та сегментації, фінальний прогноз буде відрізнятися від послідовного підходу.

Графік сигмоїдної функції, та добутку двох ідентичних сигмоїдних функцій зображено на рисунку 3.6.

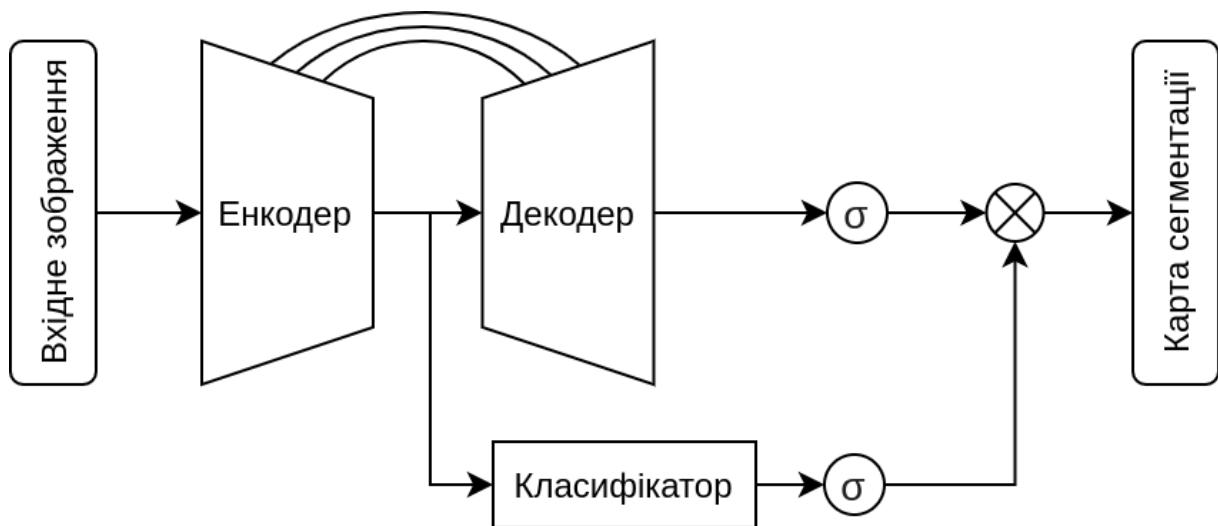


Рис. 3.5: Структура об'єднання прогнозів класифікації та сегментації

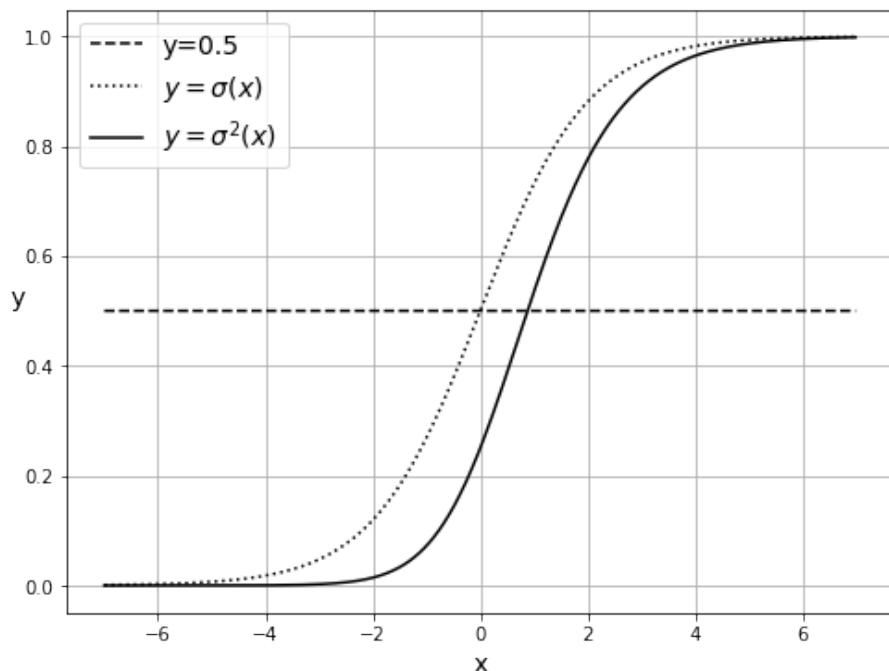


Рис. 3.6: Структура декодера класифікації

При використанні того самого порогу бінаризації для карт сегментації, зменшиться кількість позитивних пікселів в результаті, тому при прямому переході між однозадачним методом прогнозування, потрібно використовувати квадрат t_{cls} :

$$\hat{t}_{cls} = t_{cls}^2 \quad (3.15)$$

Таким чином, кількість позитивних пікселів не заміниться за рахунок зміни в нормалізації сигмоїд.

3.2.4. Локалізація важливих для класифікації ознак в умовах відсутності розмітки для сегментації в навчальному наборі даних. На основі двох представлених методів, розроблено метод навчання з частковим залученням вчителя та метод пост-обробки, що дозволяє виконувати локалізацію важливих для класифікації ознак на вихідному зображені, що є корисним під час інтерпретації прогнозів моделей. За рахунок використання методів багатозадачного навчання досягається можливість застосування методу в умовах відсутності розмітки семантичної сегментації в навчальному наборі даних.

Даний метод застосовується для навчання тієї самої моделі, але за умови наявності лише розмітки для задачі класифікації. Так само, як і в розділі 3.1 використані наступні позначення:

x - вхідне зображення енкодера розмірами $3 \times H \times W$

$F_{encoder}(x, \theta_{enc})$ - функція енкодера зображень,

θ_{enc} - набір параметрів нейронної мережі енкодера,

$v_1, v_2, \dots, v_i, \dots, v_n$ - набір карт ознак енкодера, відповідно до рівняння 3.1,

F_{seg} та F_{cls} - нейронні мережі декодера сегментації та класифікації,

M_{seg} та C_{cls} - результати сегментації та класифікації відповідно до рівнянь 3.2 та 3.3

\hat{M}_{seg} та \hat{C}_{cls} - нормовані результати сегментації та класифікації відповідно до рівнянь 3.12 та 3.13

В основі запропонованого методу лежить ітеративне уточнення карти ознак сегментації за допомогою направлена градієнтів від задачі класифікації. В даному випадку, під час навчання декодер локалізації використовується в ролі механізму уваги [102], що навчається автоматично за рахунок градієнтів до задачі класифікації. Хоча, при використанні даного методу, якість сегментації є низькою в районі контурів об'єктів, отриманих результатів достатньо для локалізації цих об'єктів на зображеннях.

Запропонований метод складається з двох етапів: етапу навчання та етапу прогнозування.

Напівавтоматичне навчання ШНМ в задачі локалізації. Першим етапом, обчислюється уточнена ознак класифікації. Для цього, обчислюється добуток Адамара між нормованим за допомогою сигмоїдної функції виходом декодера сегментації та логітами класифікації:

$$M_{unsup} = \hat{M}_{seg} \circ C_{cls} \quad (3.16)$$

Далі, для отримання результату класифікації виконується сумація елементів M_{unsup} з нормалізацією за сумою елементів оригінальної ненормалізованої карти сегментації:

$$C_{unsup} = \frac{\sum_{h=0}^H \sum_{w=0}^W M_{unsup(h,w)}}{\sum_{h=0}^H \sum_{w=0}^W M_{seg(h,w)} + c} \quad (3.17)$$

Для чисельної стабільності, до знаменника додано малу константу $c \approx 10^{-5}$

Оскільки масштаб нормованого виходу декодера сегментації знаходиться на проміжку $[0, 1]$, використання добутку Адамара дозволяє розглядати \hat{M}_{seg} як карту важливості регіонів для задачі класифікації. В процесі

навчання нейронної мережі, така структура поєднання задач спонукає нейронну мережу до призначення високих значень ($(\hat{M})_{seg} \rightarrow 1$) для важливих ознак, що є спільними на зображеннях з навчального набору даних.

Для запобігання вивченю нейронною мережею карт сегментації, що складаються виключно з високих значень, необхідна така ініціалізація параметру зсуву останнього шару декодера сегментації F_{seg} , щоб активації, за замовчуванням, були близькими до нуля. Для сигмоїдної функції активації таке значення дорівнює -4.6 .

Прогнозування результатів сегментації. На етапі прогнозування використовується як декодер класифікації, так і декодер, що відповідає за задачу локалізації. Тільки у тому випадку, коли вихід декодера класифікації перевищує заданий поріг T_c , виконується процедура декодування сегментації.

Оскільки виходи декодера сегментації M_{unsup} є неперервними, а їхній масштаб визначається процесом навчання, поріг бінаризації сегментації T_{seg} може бути різним для різних зображень. Для вибору оптимального порогу T_{seg} на кожному з вхідних зображень, в процесі пост-обробки запропоновано використати адаптивну бінарізацію за методом Оцу, щоб уникнути необхідності калібрації прогнозів нейронної мережі.

Для цього, вихід декодера сегментації M_{unsup} квантизується до 256 значень M_q , після чого ітеративно знаходиться поріг T_{seg} , що мінімізує дисперсію всередині каналів маски для кожного з класів, яка визначається як зважена сума дисперсій класів переднього та заднього плану:

$$\sigma_w^2(T_{seg}) = \min : \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \quad (3.18)$$

Тут ω_0 , ω_1 - ймовірності класів при розділенні порогом t , а σ_0^2 та σ_1^2 - дисперсії класів.

Після виконання бінаризації, над отриманою маскою $M_t = M_q > T_{seg}$

виконується морфологічна операція ерозії [?] з квадратним ядром розміру 1% від розміру зображення для того, щоб позбутися малих можливо хибно-позитивних регіонів маски:

$$M_e = M_t \ominus k \quad (3.19)$$

де k - ядро ерозії.

Результатом такого перетворення є бінаризована карта сегментації, в якій розмічено найбільш вірогідні ознаки, що вплинули на результат класифікації.

3.3. Висновки до третього розділу

1. Запропоновано модель багатозадачної нейронної мережі, що дозволяє вирішувати задачі семантичної сегментації та класифікації для аналізу планарних зображень.
2. Запропоновано нормалізацію внутрішніх представлень декодерів, що дозволяє використовувати розроблені методи багатозадачного навчання.
3. На основі запропонованої моделі багатозадачної нейронної мережі розроблено метод багатозадачного навчання, що в умовах частково-помилкової розмітки дозволяє зменшити вплив неправильно розмічених прикладів на процес навчання за рахунок їх ефективної фільтрації безпосередньо під час навчання.
4. Запропоновано метод автоматичної фільтрації помилкової розмітки, що спирається на відсіювання градієнтів в точках, що відповідають занадто високим значенням функції втрат.
5. Запропоновано використання задачі класифікації як більш загальної до задачі семантичної сегментації при багатозадачному навчанні для оновлення параметрів енкодера в точках, що були помилково

відфільтровані як зашумлені.

6. Удосконалено метод об'єднання задач сегментації та класифікації на етапі прогнозування для зменшення кількості хибно-позитивних результатів.
7. На основі двох представлених методів, розроблено метод навчання з частковим залученням вчителя та метод пост-обробки, що дозволяє виконувати локалізацію важливих для класифікації ознак на вихідному зображення, що є корисним під час інтерпретації прогнозів моделей.

РОЗДІЛ 4

ПОБУДОВА ТА ВИПРОБУВАННЯ СИСТЕМИ

АВТОМАТИЗОВАНОГО СКРИНІНГУ

В даному розділі розглянуто експерименти із розробленими моделями нейронних мереж та методами їх навчання (розділ 3.1) на синтетичних даних, що були згенеровані за допомогою запропонованої в розділі 2 моделі, а також експерименти, поставлені на наборах даних з реальних задач скринінгу.

4.1. Інструментальні засоби

Застосування розроблених моделей та методів значно ускладнюється через відсутність інструментальних програмних засобів, що забезпечували б їх роботу. Розробка інструментальних засобів, що реалізують моделі та методи є актуальною задачею.

В даний час швидко розвивається складність програмного забезпечення, що виконує наукові та інженерні задачі. Сучасні системи автоматизованого скринінгу являють собою складні багатокомпонентні системи програмних модулів, які можуть бути фізично розподілені один від одного. Так, наприклад, обладнання для скринінгу може знаходитися безпосередньо в лікарні, в той час як сервіси для обробки даних можуть бути пов'язані з обладнанням, або знаходитися на сторонніх сервісах. Також, все частіше нагальною є потреба зміни модулів в процесі роботи системи, наприклад, при оновленні нейронних мереж, що є їхніми складовими частинами.

Задача навчання глибоких нейронних мереж є обчислюально-складною

задачею, тому для прискорення процесів навчання та прогнозування необхідно використовувати хмарні технології обчислень для задач машинного навчання.

На основі запропонованих у дисертації моделей, розроблених методів навчання нейронних мереж та прогнозування результатів, реалізовано інструментальні засоби у вигляді програмних модулів для навчання та прогнозування на основі хмарних сервісів.

Всі інструментальні засоби реалізовано мовою програмування Python. Провайдером хмарних сервісів є Amazon Web Services [103]. Моделі глибоких нейронних мереж та методи іх навчання реалізовано на базі фреймворку автоматичного диференціювання Pytorch [104]. Відтворюваність результатів експериментів забезпечується використанням програмної високорівневої обгортки Catalyst [105], яка дозволяє зберігання конфігурації експериментів у широко анотованих конфігураційних файлах.

Інструментальні засоби навчання моделей розгорнуті за допомогою сервісу AWS EC2 [106], в той час як прогнозування результатів розгорнуто за допомогою AWS Sagemaker [107].

Зберігання даних та артефактів навчання моделей (файли конфігурації та бінарні файли) зберігаються за допомогою сервісу AWS S3 [108].

Сучасні нейронні мережі в цілому, та запропоновані їх моделі потребують високої кількості обчислювальних ресурсів в процесі навчання. Необхідно складовою обладнання для їх навчання є відеоприскорювачі загального призначення (*англ.* General-Purpose Graphic Processing Unit - GPGPU). Економічно-вигідним є використання арендних відеоприскорювачів в «хмарних сервісах», що потребує відповідної зміни парадигм проєктування систем навчання.

Протягом останніх років виник інтерес до запуску нейронних мереж на пограничних пристроях як до альтернативи роботи на віддалених серверах, наприклад, безпосередньо на камерах відеоспостереження, для зниження

витрат на підтримку інфраструктури.

Таким чином, створення такої архітектури, яка б дозволяла мати можливість зміни місця розташування сервісів навчання та прогнозування є актуальним.

Загальна структура інструментальних засобів зазначена на рисунку 4.1.

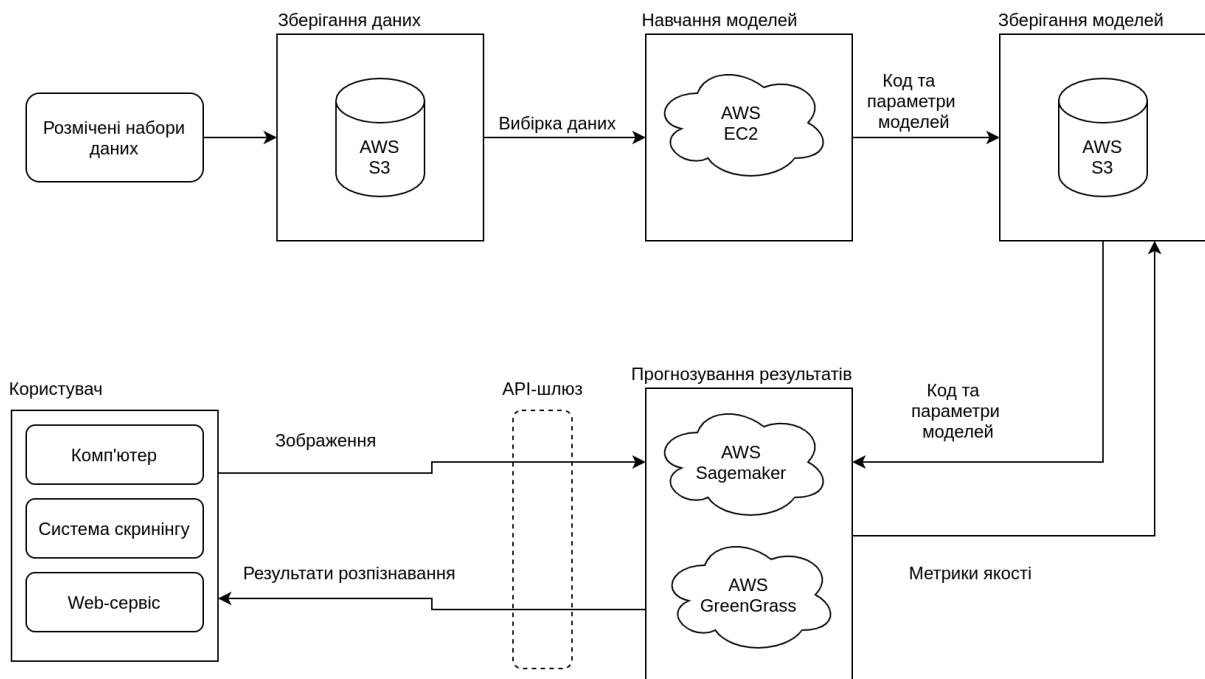


Рис. 4.1: Архітектура інструментальних засобів

Така композиція архітектури має можливість інтеграції в більшість систем автоматизованого скринінгу, а також спрощує написання клієнтських модулів.

4.2. Експерименти на синтетичних даних

Для оцінки запропонованих методів в контролюваних умовах помилкових анотацій, проведено експерименти на згенерованих наборах даних, що були синтезовані за допомогою розробленої моделі наборів даних із частково-помилковими анотаціями.

4.2.1. Оцінка прогнозів нейронної мережі.

Основні метрики для оцінки задач. В наведених нижче експериментах використовуються наступні метрики:

- Метрика F1 для оцінки задачі класифікації
- Коефіцієнт Дайса для оцінки задачі сегментації
- Коефіцієнт роздільноті внутрішніх представлень K_{sep}

В експериментах поставлена задача багатокласової класифікації та семантичної сегментації, для отримання фінального значення метрик, вони усереднені по класах.

Кількісна оцінка роздільноті внутрішніх представлень нейронної мережі. Для оцінки роздільноті внутрішніх представлень, використовується метод K-середніх. Оскільки в задачах класифікації та семантичної сегментації заздалегідь відома кількість класів, можна визначити необхідну кількість кластерів ознак у внутрішньому представленні нейронної мережі.

Для набору даних над ознаками з внутрішнього представлення обчислюється кількість кластерів, що відповідає кількості класів в основній задачі. Після чого, обчислюється середня Евклідова відстань між центрами кластерів. Таким чином, більшій відстані між кластерами відповідає краща роздільність внутрішніх представлень.

Також, для наочної демонстрації складу внутрішніх представлень, ознаки проектиуються за допомогою алгоритму нелінійного відображення Uniform Manifold Approximation and Projection (UMAP) [109] на двомірну площину.

4.2.2. Структура нейронних мереж. У всіх експериментах використана однакова структура нейронних мереж якщо не зазначено інакше.

Структура базової моделі. Базова модель являє собою екземпляр архітектури UNet. В ролі енкодера використана архітектура нейронної мережі ResNet34, що ініціалізована вагами претренованої на наборі даних

Imagenet нейронної мережі.

Декодер виконано відповідно до класичної архітектури UNet, що складається з п'яти стадій (рисунок 3.2). Кількість каналів в згортках стадій від найглибшої: 256, 128, 64, 32, 16. Функція активації декодера - ReLU. На кожній стадії використовується пакетна нормалізація ознак. Додаткові з'єднання від енкодера передаються за допомогою операції конкатенації. Функція активації останнього шару декодера - логістична сигмоїда.

Структура запропонованої моделі. Енкодер та декодер сегментації запропонованої моделі виконані відповідно до структури, описаної в розділі 3.1. При навчанні запропонованої моделі використовується обмежена функція втрат для задачі сегментації, а при прогнозуванні використовується метод обєднання задач класифікації та сегментації.

4.2.3. Параметри експериментів.

Загальна процедура навчання. Складається з одного циклу навчання та валідації, якщо не вказано інакше.

Використано наступні параметри навчання:

- Розмір зображення: 224×224 пікселя
- Розмір пакету навчання: 32
- Кількість тренувальних зображень: 15000
- Кількість тестових зображень: 5000
- Кількість епох навчання: 20
- Оптимізатор: RAdam [110]
- Темп навчання: 10^{-3}
- Коефіцієнт L2 регуляризації параметрів мережі: 10^{-4}
- Закон зміни темпу навчання: косинус [111]

Експеримент 1. Порівняння базового та запропонованого методів на простому наборі даних із достовірними анотаціями. Параметри генерації

набору даних зазначено в таблиці 4.1.

Таблиця 4.1: Параметри набору даних експеримента 1

Об'єкти	MNIST
Текстура фону	константна
Текстура об'єктів	константна
P_e	0
P_d	0
N_{obj}	10
S_{obj}	36
Δ_{max}	0.2

Результати експерименту наведено в таблиці 4.2.

Таблиця 4.2: Результати експеримента 1

Модель	Міра Дайса	F1-міра	K_{sep}
Базова модель	0.86	-	14.33
Запропонований метод	0.87	0.98	17.51

На рисунку 4.2 зображені приклади прогнозів нейронної мережі для тестових зображень.

На рисунках 4.3 та 4.4 зображені УМАР-проекцію ознак останнього шару енкодера для набору тестових зображень.

Експеримент 2. Порівняння базового та запропонованого методів на простому наборі даних із частково-помилковими анотаціями. Параметри генерації набору даних зазначено в таблиці 4.3.

Таблиця 4.3: Параметри набору даних експеримента

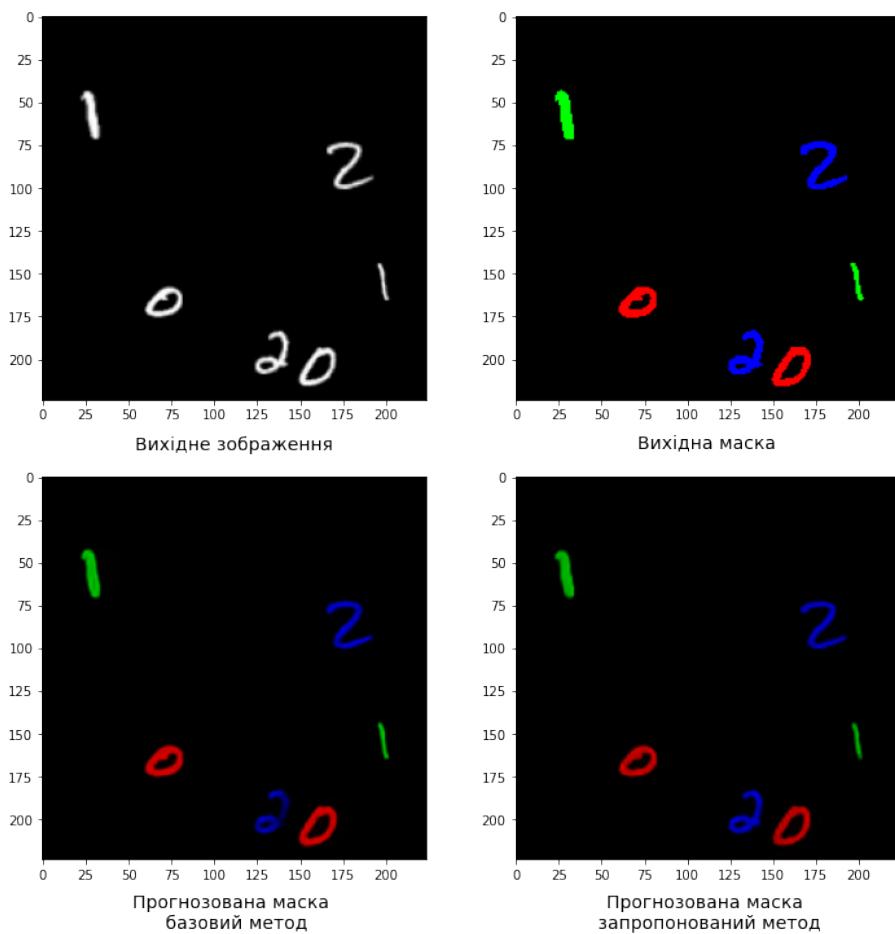


Рис. 4.2: Прогнози нейронної мережі для тестового зображення

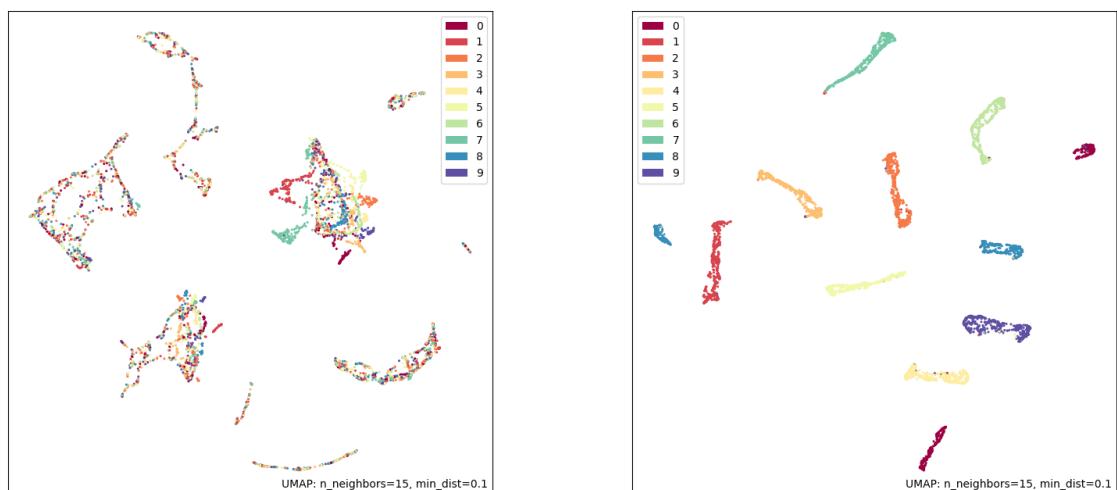


Рис. 4.3: UMAP ознак
базової моделі

Рис. 4.4: UMAP ознак
запропонованої моделі

Об'єкти	MNIST
Текстура фону	константна
Текстура об'єктів	константна
P_e	0.1
P_d	0.25
N_{obj}	10
S_{obj}	36
Δ_{max}	0.2

Результати експерименту наведено в таблиці 4.4.

Таблиця 4.4: Результати експеримента

Модель	Mира Дайса	F1-міра	K_{sep}
Базова модель	0.81	-	12.10
Запропонований метод	0.91	0.93	15.73

На рисунку 4.5 зображено приклади прогнозів нейронної мережі для тестових зображень.

На рисунках 4.6 та 4.7 зображено УМАР-проекцію ознак останнього шару енкодера для набору тестових зображень.

В умовах навчання за частково-помилковими анотаціями, запропонований метод дає більш впевнені прогнози, також зберігаючи тонкі лінії, що порушуються при виконанні операції ерозії над розміткою.

Експеримент 3. Порівняння базового та запропонованого методів на простому наборі даних з ускладненими об'єктами та частково-помилковими анотаціями. Параметри генерації набору даних зазначено в таблиці 4.5.

Таблиця 4.5: Параметри набору даних експеримента

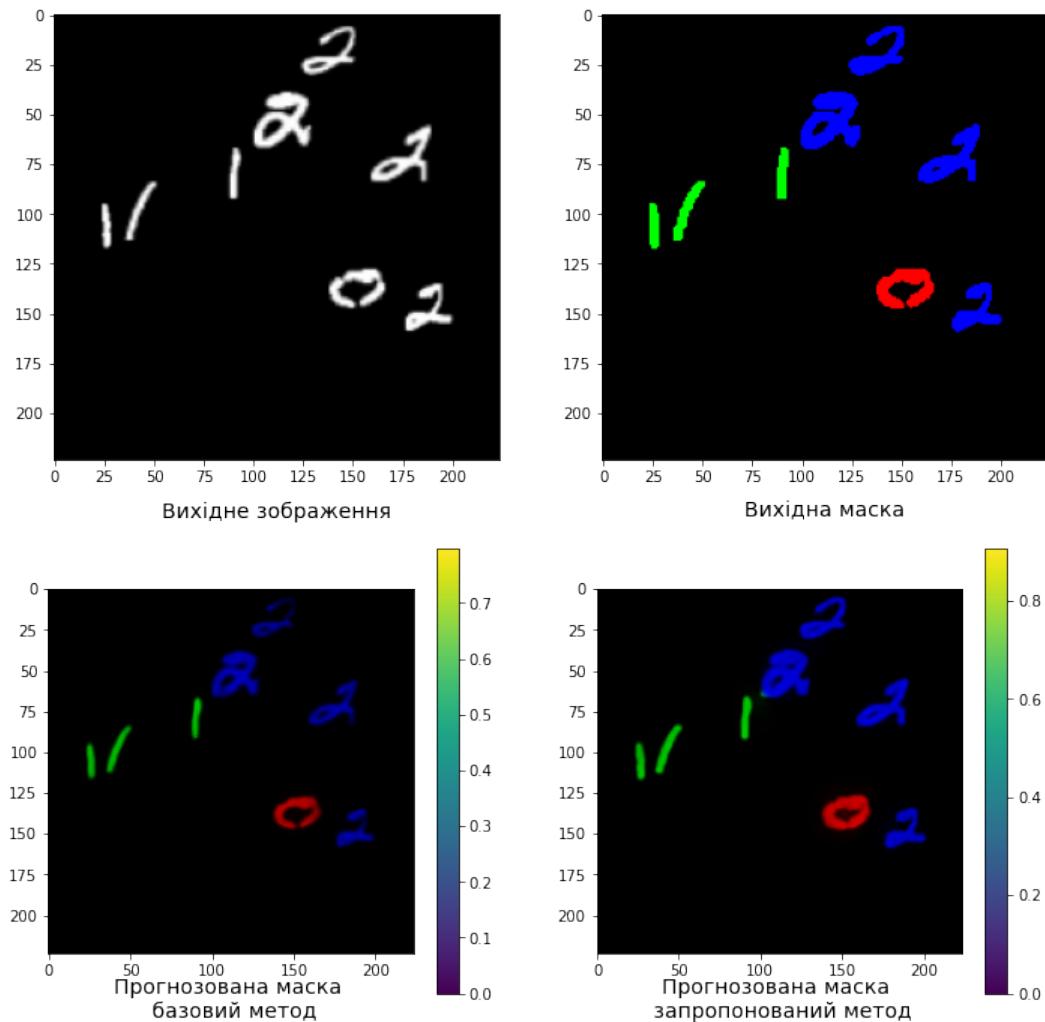


Рис. 4.5: Прогнози нейронної мережі для тестового зображення

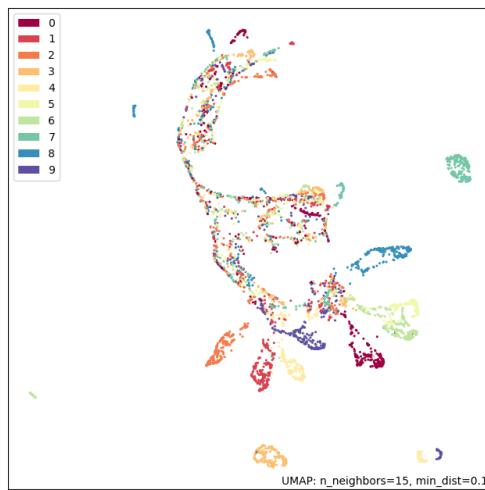


Рис. 4.6: УМАР ознак
базової моделі

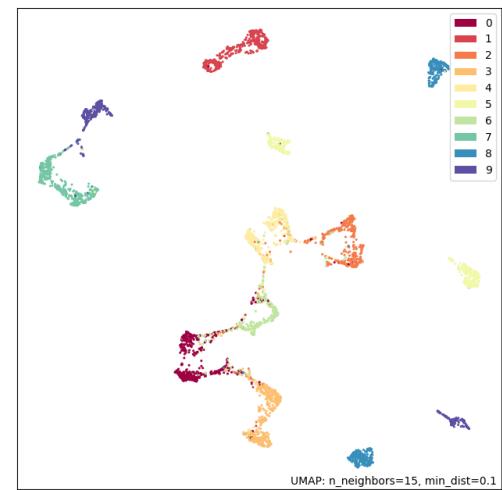


Рис. 4.7: УМАР ознак
запропонованої моделі

Об'єкти	FashionMNIST
Текстура фону	константна
Текстура об'єктів	константна
P_e	0.1
P_d	0.25
N_{obj}	10
S_{obj}	36
Δ_{max}	0.2

Результати експерименту наведено в таблиці 4.6.

Таблиця 4.6: Результати експеримента

Модель	Mира Дайса	F1-міра	K_{sep}
Базова модель	0.83	-	27.49
Запропонований метод	0.87	0.88	32.12

На рисунку 4.8 зображені приклади прогнозів нейронної мережі для

тестових зображень.

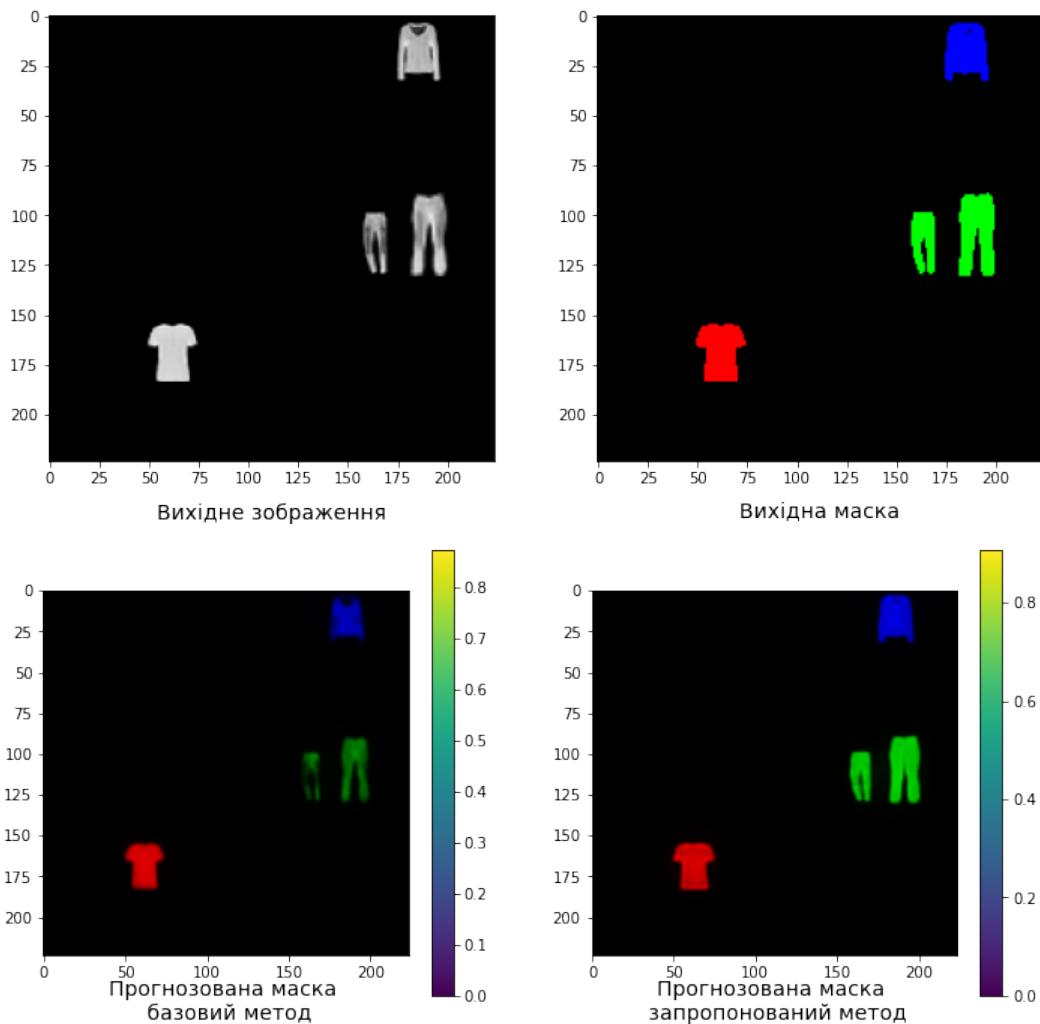


Рис. 4.8: Прогнози нейронної мережі для тестового зображення

На рисунках 4.9 та 4.10 зображене УМАР-проекцію ознак останнього шару енкодера для набору тестових зображень.

Як і в попередньому експерименті, в умовах частково-помилкових анотацій, запропонований метод дає більш впевнені прогнози, також зберігаючи більше малих деталей.

Експеримент 4. Порівняння базового та запропонованого методів на складному наборі даних з простими об'єктами та частково-помилковими анотаціями. Параметри генерації набору даних зазначено в таблиці 4.7.

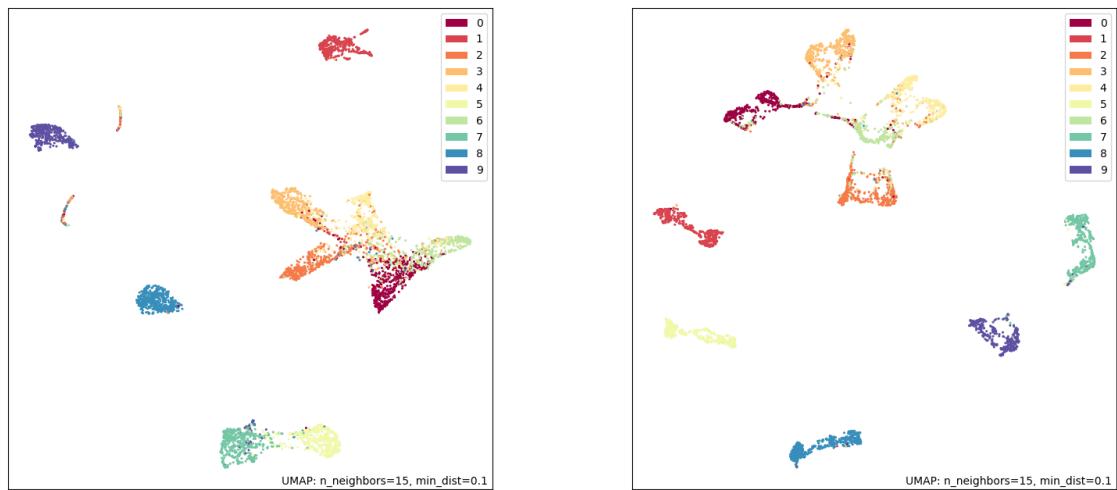


Рис. 4.9: UMAP ознак
базової моделі

Рис. 4.10: UMAP ознак
запропонованої моделі

Таблиця 4.7: Параметри набору даних експеримента

Об'єкти	MNIST
Текстура фону	Imagenette
Текстура об'єктів	Imagenette
P_e	0.1
P_d	0.25
N_{obj}	10
S_{obj}	36
Δ_{max}	0.2

Результати експерименту наведено в таблиці 4.8.

Таблиця 4.8: Результати експеримента

Модель	Міра Дайса	F1-міра	K_{sep}
Базова модель	0.69	-	15.26
Запропонований метод	0.72	0.75	18.90

На рисунку 4.11 зображені приклади прогнозів нейронної мережі для тестових зображень.

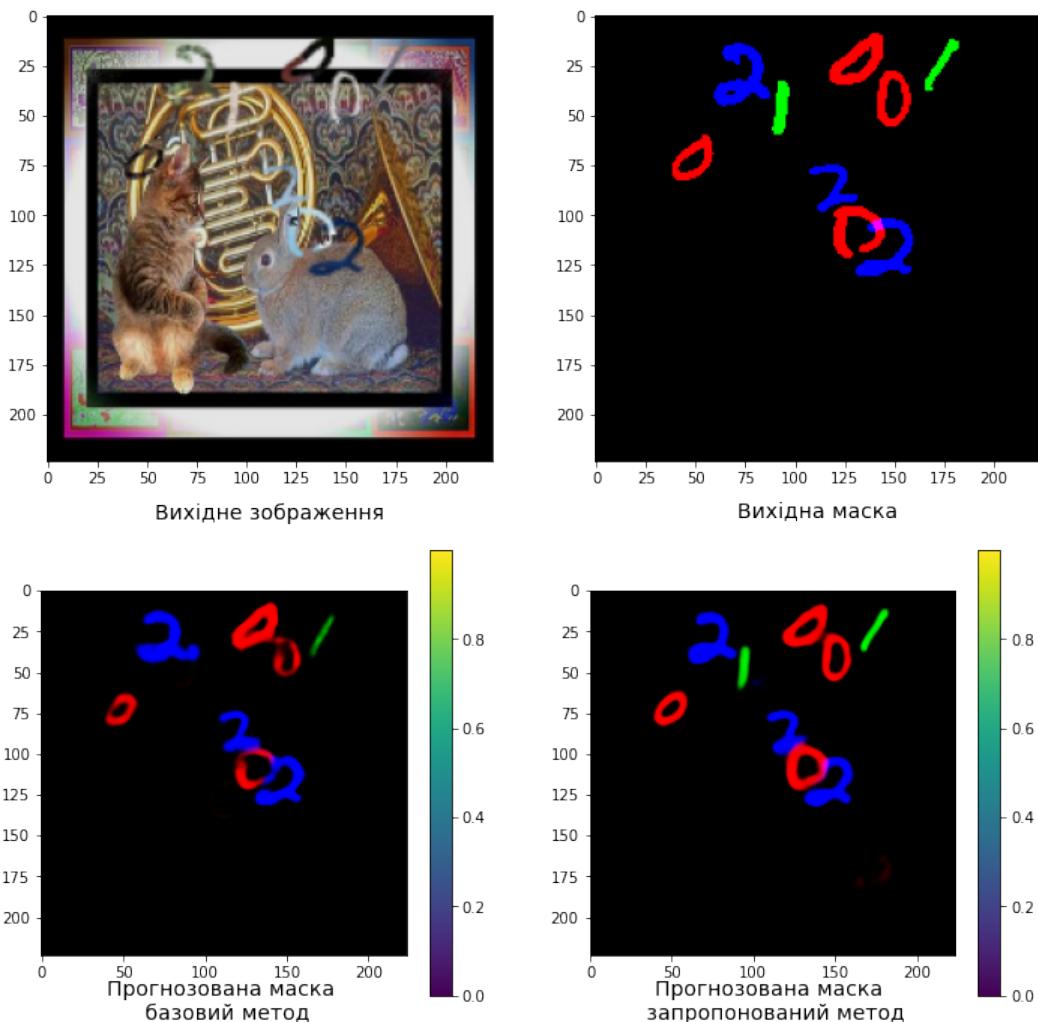


Рис. 4.11: Прогнози нейронної мережі для тестового зображення

На рисунках 4.12 та 4.13 зображені UMAP-проекцію ознак останнього шару енкодера для набору тестових зображень.

Як і в попередньому експерименті, в умовах навчання з частково-помилковими анотаціями, запропонований метод дає більш впевнені прогнози, також зберігаючи більше малих деталей. Однак, внутрішні представлення мають гір-

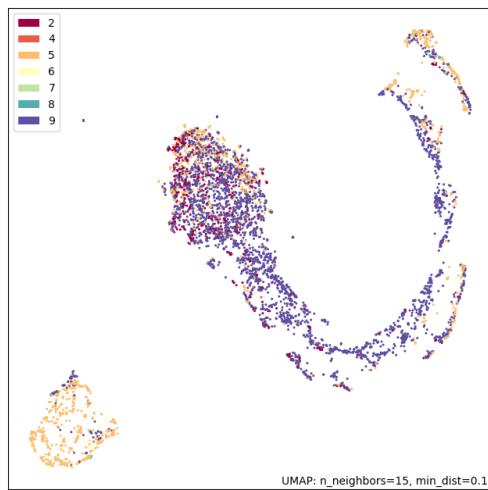


Рис. 4.12: УМАР ознак
базової моделі

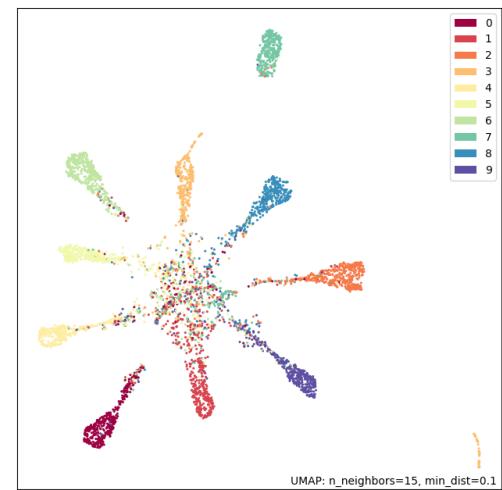


Рис. 4.13: УМАР ознак
запропонованої моделі

шу роздільність через більш складний тип вхідних даних.

Експеримент 5. Порівняння базового та запропонованого методів на складному наборі даних з ускладненими об'єктами та частково-помилковими анотаціями. Параметри генерації набору даних зазначено в таблиці 4.9.

Таблиця 4.9: Параметри набору даних експеримента

Об'єкти	FashionMNIST
Текстура фону	Imagenette
Текстура об'єктів	Imagenette
P_e	0.1
P_d	0.25
N_{obj}	10
S_{obj}	36
Δ_{max}	0.2

Результати експерименту наведено в таблиці 4.10.

Таблиця 4.10: Результати експеримента

Модель	Міра Дайса	F1-міра	K_{sep}
Базова модель	0.70	-	7.02
Запропонований метод	0.75	0.78	10.6

На рисунку 4.14 зображені приклади прогнозів нейронної мережі для тестових зображень.

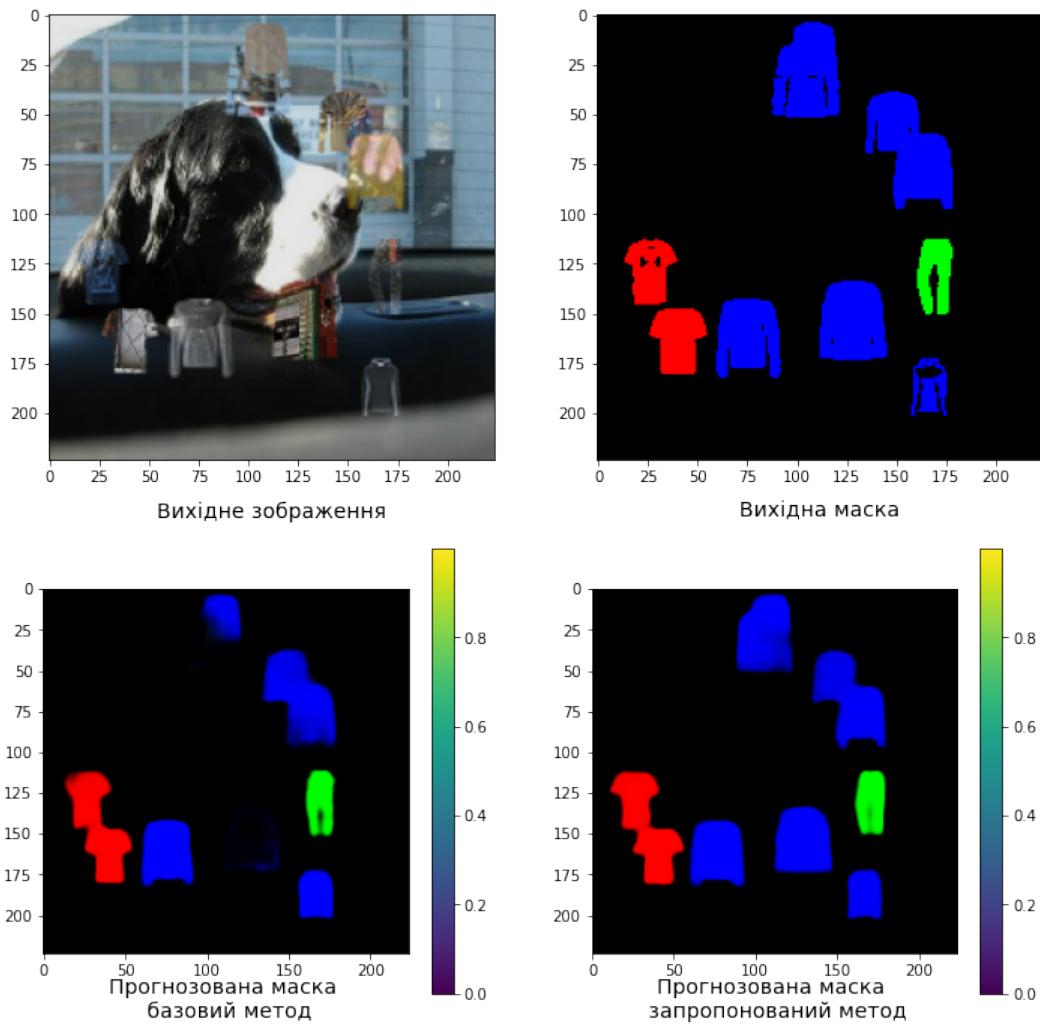


Рис. 4.14: Прогнози нейронної мережі для тестового зображення

На рисунках 4.15 та 4.16 зображені UMAP-проекцію ознак останнього шару енкодера для набору тестових зображень.

Як і в попередньому експерименті, в умовах помилок в анотаціях, за-

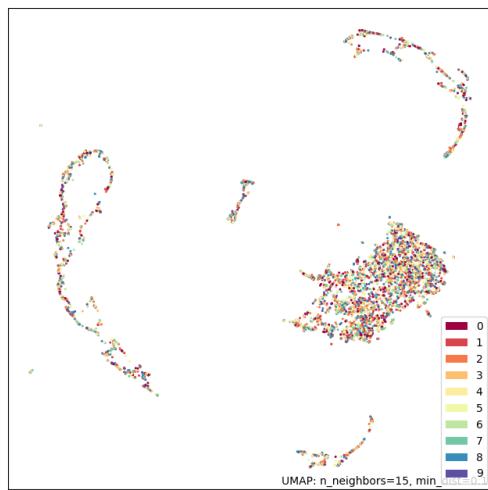


Рис. 4.15: УМАР ознак
базової моделі

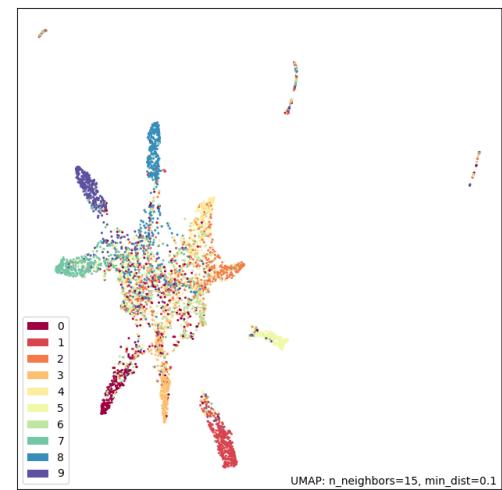


Рис. 4.16: УМАР ознак
запропонованої моделі

пропонований метод дає більш впевнені прогнози, також зберігаючи більше малих деталей. Однак, внутрішні представлення мають гіршу роздільність через більш складний тип вхідних даних.

Експеримент 6. Стійкість методів до різних рівнів помилок в анотаціях на простих даних.

Для оцінки стійкості базового та запропонованого методів до відсотку помилок в анотаціях, проведено серію експериментів зі зміною вірогідності помилки для кожного з об'єктів в наборі даних.

Результати експерименту наведено в таблиці 4.11

Таблиця 4.11: Порівняння стійкості моделей до помилкових анотацій

P_e	P_d	Міра Дайса Базовий метод	Міра Дайса Запропонований метод
0.1	0.1	0.84	0.96
0.1	0.25	0.81	0.91
0.25	0.1	0.65	0.83
0.5	0.25	0.51	0.62
0.25	0.5	0.57	0.69
0.5	0.5	0.3	0.47
0.75	0.75	0.12	0.23

З таблиці видно, що запропонований метод є більш стійким до більшого відсотку помилкових анотацій. Також, як запропонований, так і базовий методи менш стійкі до підвищеної ймовірності ерозії.

Експеримент 7. Стійкість методів до різних рівнів помилок в анотаціях на складних даних.

Для перевірки стійкості до помилок в анотаціях в умовах наблизених до реальних, попередній експеримент проведено з використанням фону та текстур об'єктів з набору даних Imagenette. Результати експерименту наведено в таблиці 4.12.

Таблиця 4.12: Порівняння стійкості моделей до помилок в анотаціях

P_e	P_d	Міра Дайса Базовий метод	Міра Дайса Запропонований метод
0.1	0.1	0.72	0.78
0.1	0.25	0.70	0.75
0.25	0.1	0.61	0.69
0.5	0.25	0.48	0.53
0.25	0.5	0.55	0.60
0.5	0.5	0.21	0.41
0.75	0.75	0.08	0.15

Результати експерименту збігаються з результатами попереднього експерименту, хоча мають тенденцію до зменшення середнього значення метрики через підвищенну складність даних.

Дослідження внеску окремих компонентів. Параметри генерації набору даних зазначено в таблиці 4.13.

Таблиця 4.13: Параметри набору даних для проведення дослідження внеску окремих компонентів

Текстура фону	Imagenette
Текстура об'єктів	Imagenette
P_e	0.2
P_d	0.2
N_{obj}	10
S_{obj}	36
Δ_{max}	0.2

В таблиці 4.14 наведено значення міри Дайса в задачі семантичної сегментації при послідовному додаванні складових частин запропонованого метода до базової моделі. Використані скорочення:

- MN - MNIST
- FM - FashionMNIST
- IN - ImageNette
- С - константний колір

Використано наступний порядок кодів в шифрі:

1. Набір даних, з якого взято об'єкти
2. Набір даних, з якого взято текстури фону
3. Набір даних, з якого взято текстури об'єктів

Таблиця 4.14: Дослідження внеску окремих компонентів

Змінна	MN/C/C	FM/C/C	MN/IN/IN	FM/IN/IN
Базова модель	0.81	0.83	0.69	0.70
+ декодер класифікації	0.83	0.84	0.69	0.71
+ обмеження функції втрат	0.88	0.86	0.71	0.74
+ комбінація прогнозів	0.91	0.87	0.72	0.75

З зазначеної таблиці видно, що найбільший приріст дають обмеження функції втрат та комбінація прогнозів, хоча для отримання повного ефекту необхідні всі складові частини запропонованого методу. Також, більш складні набори даних зменшують відносний приріст метрик, що є очікуваним через потребу збільшення кількості параметрів в нейронній мережі для можливості вивчення ускладнених представлень.

4.3. Експерименти на реальних даних

4.4. Класифікація та сегментація формаций хмар

4.4.1. Опис набору даних Understanding Clouds from Satellite Images. Набір даних Understanding Clouds from Satellite Images (UCSID) складається з 10000 RGB зображень, зроблених з двох супутників TERRA та AQUA, що знаходяться на полярній орбіті [6]. Кожен з цих супутників проходить над певну областью один раз на день. Через обмежене поле зору камер, встановлених на цих супутниках, кожне зображення зшито з двох супутників, що знаходяться над однією й тою самою областью одночасно, але на різних орбітах. Решта зображень, для якої не було знято даних під час прольоту супутників (між орбітами) заповнена чорним кольором. Оскільки зображення підлягають компресії, є артефакти стиснення, та чорний колір має деякі незначні аномалії.

На знімках є регіони, які містять певні хмарні утворення, та помічені

дослідниками відповідно: Риба, Квітка, Гравій, Цукор (*англ.* Fish, Flower, Gravel, Sugar). Кожне зображення має принаймні одне хмарне утворення і може містити до всіх чотирьох одночасно.

Розмітка для цих регіонів була створена під час краудсорсингу в Інституті метеорології імені Макса Планка в Гамбурзі, Німеччина, та Laboratoire de météorologie Dynamique у Парижі, Франція. Команда з 68 вчених виявила ділянки з хмарами на кожному зображенні, і кожне зображення було розмічене, в середньому, трьома різними вченими. Кожен вчений мав виділити хмари за допомогою прямокутних областей на власний розсуд. Основна розмітка була створена об'єднанням областей, що були розмічені всіма вченими для цього зображення, після видалення чорної смуги з цих областей [7].

Через те, як був зібраний UCSID, він має значну кількість помилково-анотованих пікселів масках для хмар. Оскільки маски складаються з прямокутників, які повністю перекривають хмару, існує багато пікселів, які позначені як хмари, але, насправді, відповідають фону. Також, не всі хмари позначені масками. Через об'єднання масок від різних анотаторів, класи можуть суттєво перетинатися - один і той самий піксель може бути помічений декількома класами, включаючи випадки, коли всі чотири класи присвоюються одним і тим самим пікселям.

Усі зображення мають вихідну роздільну здатність 2100x1400 пікселів. Більшість зображень містять більше ніж один клас хмар. Немає зображень без хмар. Розподіл класів хмар на зображеннях показано на рисунку 4.17.

Різні типи хмар зазвичай зустрічаються разом, розподіл комбінацій різних типів хмар зазначено на рисунку. UCSID розділено на 5546 тренувальних, та 3698 валідаційних зображень. Крім того, немає доступу до міток валідаційного набору даних. Значення метрики можливо отримати через систему валідації.

До розподілу міток класів в наборі даних не було виконано змін (недо-

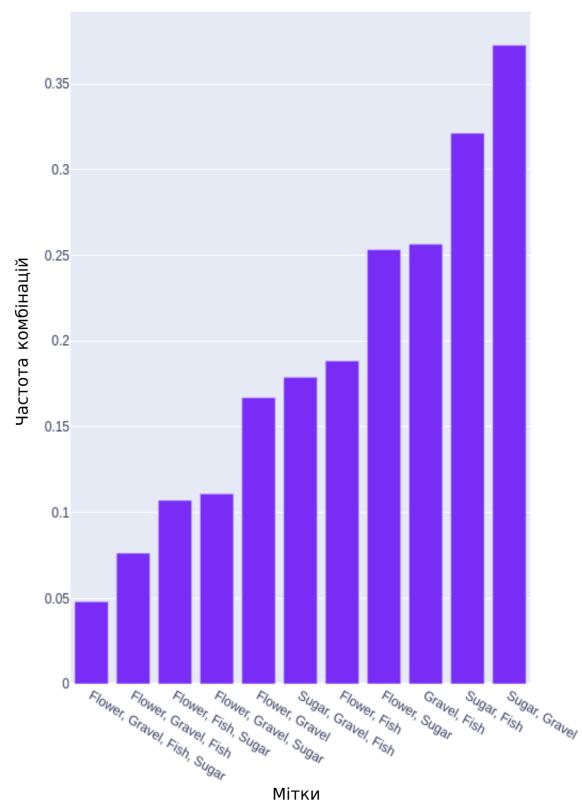


Рис. 4.17: Частота зустрічі декількох типів хмар на одному зображені

дискретизація, передискретизація тощо). Для отримання більш стабільного значення метрик в умовах помилок в тестових анотаціях, оцінювання якості моделей виконано за допомогою k -кратної перехресної перевірки [112].

4.4.2. Процедура навчання. Тренування та прогнозування проводиться на зменшених версіях оригінальних зображень. Загальні параметри навчання:

- Розмір зображення: 350×525 пікселя
- Розмір пакету навчання: 64
- Кількість тренувальних зображень: 5546
- Кількість тестових зображень: 3698
- Кількість епох навчання: 50
- Оптимізатор: Adam
- Темп навчання: 10^{-4}
- Коефіцієнт L2 регуляризації параметрів мережі: 10^{-4}
- Закон зміни темпу навчання: косинус
- $k : 5$

Структура базової моделі. Базова модель являє собою екземпляр архітектури UNet. В ролі енкодера використана архітектура нейронної мережі ResNet50, що ініціалізована вагами претренованої на наборі даних Imagenet нейронної мережі.

Декодер виконано відповідно до класичної архітектури UNet, що складається з п'яти стадій. Кількість каналів в згортках стадій від найглибшої: 256, 128, 64, 32, 16. Функція активації декодера - ReLU. На кожній стадії використовується пакетна нормалізація ознак. Додаткові з'єднання від енкодера передаються за допомогою операції конкатенації. Функція активації останнього шару декодера - логістична сигмоїда.

Структура запропонованої моделі. Енкодер та декодер сегментації запропонованої моделі виконані відповідно до структури, описаної в роз-

ділі 3.1. При навчанні запропонованої моделі використовується обмежена функція втрат для задачі сегментації, а при прогнозуванні використовується метод об'єднання задач класифікації та сегментації.

4.4.3. Результати експерименту. Чисельні результати експерименту зазначено в таблиці ??.

Таблиця 4.15: Результати експеримента

Модель	Міра Дайса	F1-міра
Базова модель	0.56 ± 0.03	-
Запропонований метод	0.63 ± 0.02	0.70 ± 0.01

Приклад зображення з тестового набору даних з вихідною розміткою та прогнозом нейронної мережі зображене на рисунку 4.18.

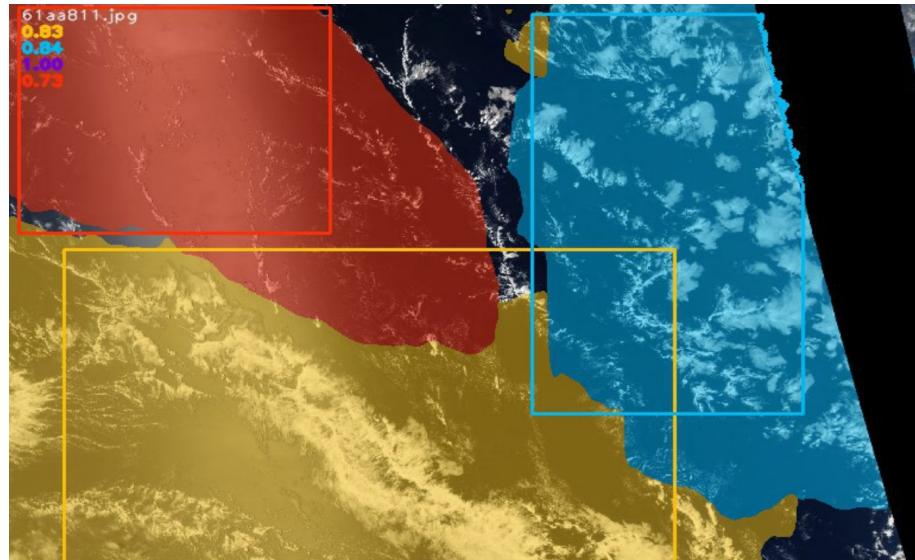


Рис. 4.18: Оригінальна розмітка (контури) та прогнози нейронної мережі (заливка)

4.4.4. Класифікація стадії осередків діабетичної ретинопатії.

В даному експерименті розглянуто можливість переносу підходу із задачі сегментації на задачу класифікації. Так, в ролі похідних задач до зада-

чі класифікації використані задачі лінійної регресії мітки класу, а також порядкової регресії, що відповідає специфіці задачі [113].

4.4.4.1. Опис наборів даних. Використані в цьому експерименті зображення, були взяті з декількох наборів даних. Для претренування нейронних мереж було використано відкритий набір даних від Kaggle: “Diabetic Retinopathy Detection Challenge 2015” dr2015.

Цей набір даних є найбільшим із загальнодоступних. Він складається з 35126 фотографій очного дна для лівого та правого ока американських громадян, з розміченими стадіями діабетичної ретинопатії за стандартним протоколом:

- Відсутність діабетичної ретинопатії (мітка 0)
- Легка діабетична ретинопатія (мітка 1)
- Помірна діабетична ретинопатія (мітка 2)
- Важка діабетична ретинопатія (мітка 3)
- Проліферативна діабетична ретинопатія (мітка 4)

Крім того, були використані менші набори даних: набір зображень індійської діабетичної ретинопатії (IDRiD) [114], з якого використано 413 фотографій очного дна, та MESSIDOR (Методи оцінки методів сегментації та індексації в області офтальмології сітківки) [115], з якого використано 1200 фотографій очного дна. Оскільки розмітка оригінального набору даних MESSIDOR відрізняється від інших наборів даних, ми використали версію, яку група офтальмологів [116] помаркувала відповідно до стандартного протоколу.

Усі зазначені набори даних мають одинаковий розподіл міток класів, що є фундаментальною властивістю для цієї задачі.

Оцінка проводиться на наборі даних Kaggle APTOS2019 [117], дослідники мають доступ лише до тренувальної та валідаційної частин. Повний набір даних складається з 18590 фотографій очного дна, які розділені на

набори з 3662 тренувальних, 1928 валідаційних та 13000 тестових зображень, що були розділені організаторами змагань Kaggle.

4.4.5. Процедура навчання. Тренування та прогнозування проводиться на зменшених версіях оригінальних зображень. Загальні параметри навчання:

- Розмір зображення: 380×380 пікселя
- Розмір пакету навчання: 32
- Кількість тренувальних зображень: 5275
- Кількість тестових зображень: 1928
- Кількість епох навчання: 75
- Оптимізатор: RAdam
- Темп навчання: 10^{-4}
- Коефіцієнт L2 регуляризації параметрів мережі: 10^{-4}
- Закон зміни темпу навчання: косинус
- $k : 5$

Структура базової моделі. Базова модель являє собою екземпляр архітектури ResNet. В ролі енкодера використана архітектура нейронної мережі ResNeXt50, що ініціалізована вагами претренованої на наборі даних Imagenet нейронної мережі.

Декодер класифікації являє собою два лінійних шари, функція активації - ReLU. Функція активації останнього шару декодера - логістична сигмоїда.

Структура запропонованої моделі. Енкодер та декодери запропонованої моделі виконані відповідно до структури, описаної в розділі 3.1. В означеній задачі використано ідентичні декодери для різних задач, структуру яких зазначено на рисунку ???. При навчанні запропонованої моделі використовується обмежена функція втрат для задачі класифікації, а при прогнозуванні використовується запропонований метод об'єднання задач.

Комбінування результатів декількох задач. На етапі після основного навчання, для кожної з моделей, навчається модель лінійної регресії з виходів декодерів в єдине значення.

Лінійна регресія навчається після основного навчання, оскільки в іншому випадку, сходиться до неоптимальних локальних мінімумів з вагами двох декодерів, близьких до нуля. Ці нульові ваги запобігають оновленню відповідних ваг декодерів і, відповідно, запобігають навчанню. Початкові ваги для кожного з виходів декодерів були встановлені рівними $1/3$, а потім тренувались протягом п'яти епох, щоб мінімізувати середньоквадратичну функцію помилки.

Ініціалізація нейронної мережі. Відпочатку, енкодер ініціалізується параметрами нейронної мережі, що була натренована на наборі даних ImageNet. Параметри декодерів ініціалізуються випадково (ініціалізація Xe).

Природні особливості діабетичної ретинопатії узгоджуються між різними людьми і не залежать від набору даних. Крім того, різні набори даних збираються на різному обладнанні. Включення цих знань у модель підвищує її здатність до узагальнення та підвищує важливість природних ознак за рахунок зменшення чутливості до особливостей обладнання.

Для попереднього навчання, ініціалізована нейронна мережа навчається на протязі 20 епох на наборі даних DRDC2015 за допомогою стохастичного градієнтного спуску. Основна мета попереднього навчання - створити ініціалізацію параметрів на розподілі даних, що є близьким до цільового. Після попереднього навчання, параметри нейронної мережі використовуються як ініціалізація для основного навчання.

Під час попереднього навчання, кожен декодер мінімізує свою функцію втрат: перехресну ентропію для класифікаційного декодера, бінарну перехресну ентропію для декодера порядкової регресії та середню абсолютну

похибку для декодера регресії.

Процедура навчання. Основне тренування проводиться на наборах даних APTOS2019, IDRID та MESSIDOR разом. Починаючи з ваг, отриманих на етапі попереднього тренування, виконується 5-кратна перехресна перевірка та оцінка моделі на відкладеному наборі даних. На цьому етапі функції втрат для декодерів змінено: фокальна функція втрат [118] для класифікаційного декодера, бінарна фокальна функція втрат [118] для декодера порядкової регресії та середньоквадратична помилка для декодера регресії.

4.4.6. Результати експерименту. Чисельні результати експерименту зазначено в таблиці 4.16.

Таблиця 4.16: Результати експеримента

Модель	F1-міра
Базова модель	0.68 ± 0.05
Запропонований метод	0.79 ± 0.06

4.5. Класифікація раку шкіри та локалізація родимок

Метою даного експерименту є тестування запропонованого методу локалізації важливих для класифікації ознак в умовах відсутності розмітки сегментації.

4.5.1. Опис набору даних SIIM-ISIC Melanoma Classification.

Дані зображень, використані в цьому дослідженні, були взяті з декількох наборів даних з однаковою структурою: SIIM&ISIC з 2017, 2018, 2019 та 2020 років. Ці набори даних були створені Міжнародною співпрацею з обробки зображень шкіри (англ. International Skin Imaging Collaboration - ISIC), а зображення отримані з наступних джерел:

- лікарня Клінік де Барселона
- Віденський медичний університет
- Центр раку Меморіал Слоун Кеттерінг
- Австралійський інститут меланоми
- Квінслендський університет
- Афінська медична школа

Загалом, ці набори даних складаються із приблизно 50000 RGB-зображень, з яких близько 3000 мають зображення злоякісних уражень. Набір даних містить 434 повторюваних зображення. Okрім даних про зображення, були надані метадані про пацієнтів. Один пацієнт має декілька зображень різних родинок.

Зображення та метадані надані у форматі DICOM, який є загальновживаним форматом даних медичних зображень. Крім того, набір даних доступний у форматі JPEG із розмірами зображень, змінених до 1024x1024. Метадані також надаються за межами формату DICOM, у файлах CSV [119].

Метадані містять наступну інформацію:

- image_name - унікальний ідентифікатор, вказує на ім'я файлу відповідного зображення DICOM;
- patient_id - унікальний ідентифікатор пацієнта;
- sex - стать пацієнта;
- age_approx - приблизний вік пацієнта на момент фотографії (ціле число);
- anatom_site_general_challenge - розташування родинки на тілі;
- diagnosis - детальна інформація про діагностику (рядок);
- benign_malignant - показник злоякісності зображеного ураження (cnhjrf, одна з "benign" і "malignant");
- target - бінарізована версія цільової змінної.

Значення для `anatom_site_general_challenge` та `diagnosis` беруться із за-
здалегідь визначеного кінцевого набору.

Метадані доступні для кожного пацієнта, тому різні зображення мо-
жуть мати одинаковий набір функцій рівня пацієнта. Ми використовуємо всі
доступні метадані, крім ідентифікатора пацієнта та діагностики, оскільки
вони доступні лише в навчальних наборах даних.

Набір даних має дисбаланс високого класу. Розподіл діагнозів показано
на рисунку 4.19.

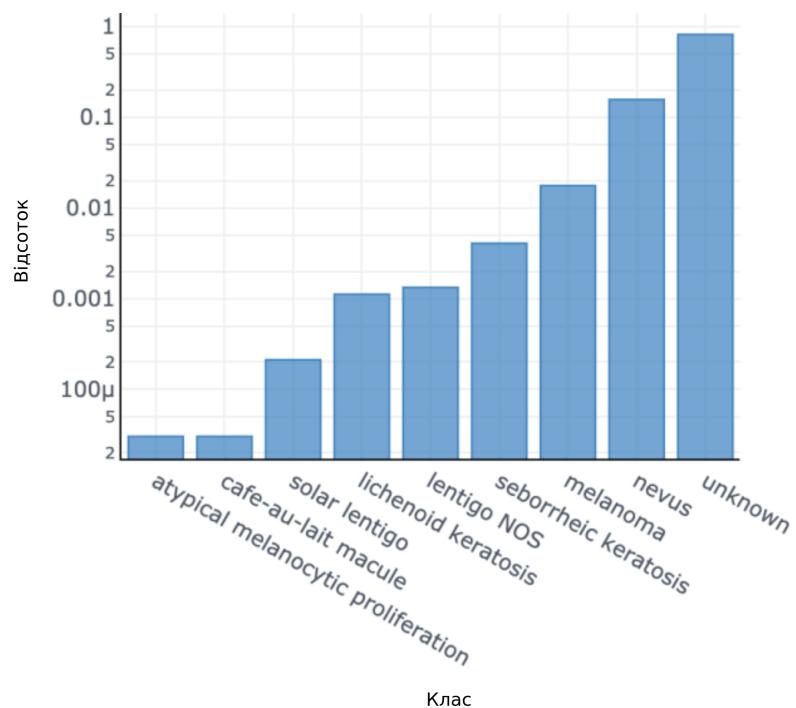


Рис. 4.19: Розподіл класів в наборі даних SIIM-ISIC Melanoma Classification

Для значень `unknown` в полі `diagnosis`, автори набору даних гарантують,
що новоутворення не є злоякісним [120].

4.5.2. Процедура навчання. Тренування та прогнозування прово-
диться на зменшених версіях оригінальних зображень. Загальні параметри
навчання:

- Розмір зображення: 512×512 пікселів

- Розмір пакету навчання: 64
- Кількість тренувальних зображень: 25000
- Кількість тестових зображень: 5000
- Кількість епох навчання: 50
- Оптимізатор: Adam
- Темп навчання: 10^{-4}
- Коефіцієнт L2 регуляризації параметрів мережі: 10^{-4}
- Закон зміни темпу навчання: косинус
- $k : 5$

Структура базової моделі. Базова модель являє собою екземпляр архітектури ResNet. В ролі енкодера використана архітектура нейронної мережі ResNet50, що ініціалізована вагами претренованої на наборі даних Imagenet нейронної мережі.

Декодер класифікації являє собою два лінійних шари, функція активації - ReLU. Функція активації останнього шару декодера - логістична сигмоїда.

Структура запропонованої моделі. Енкодер та декодери запропонованої моделі виконані відповідно до структури, описаної в розділі 3.1. Для спрощення задачі локалізації, декодер сегментації має менше стадій та виводить карту уваги в меншій роздільній здатності, ніж вхідне зображення.

Процедура навчання. Процедури навчання і прогнозування відповідають запропонованому методу напівавтоматичного навчання задачі локалізації, що описаний в розділі 3.2.4.

4.5.3. Результати експерименту. Чисельні результати експерименту зазначено в таблиці 4.17.

Таблиця 4.17: Результати експеримента

Модель	F1-міра
Базова модель	0.83 ± 0.01
Запропонований метод	0.86 ± 0.01

Приклад локалізації нейронною мережею зображенено на рисунку 4.20.

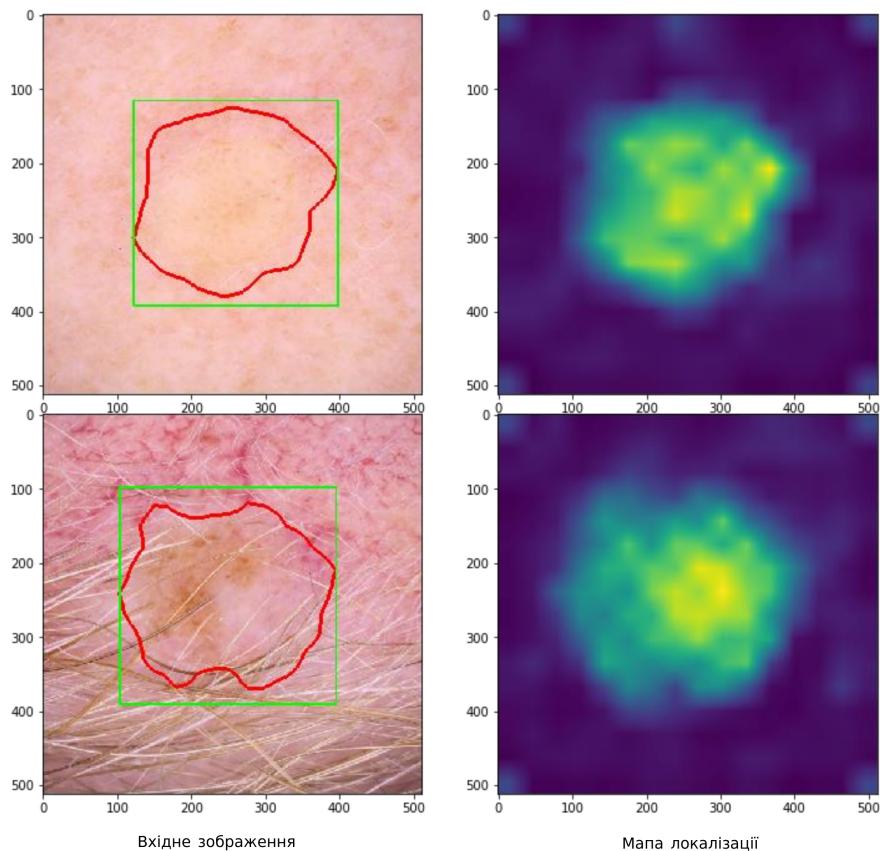


Рис. 4.20: Приклад напівавтоматичної локалізації родинок

4.6. Висновки до четвертого розділу

В четвертому розділі проведено експериментальні дослідження на синтетичних даних, що були згенеровані за допомогою моделі, а також на реальних даних з різних задач автоматизованого скринінгу.

Представлено архітектуру інструментальних засобів для реалізації запропонованих моделей, а також методів її навчання та прогнозування результатів.

В результаті експериментів показано переваги запропонованих багатозадачних нейронних мереж та методів їх навчання над базовими версіями. Проведено дослідження впливу відсотку помилкових анотацій на результати навчання, виявлено підвищення достовірності при використанні запропонованих методів для різних відсотків помилок в анотаціях.

Проведено аналіз внеску окремих компонентів запропонованих моделей нейронних і методів їх навчання, показано їх позитивний вплив на достовірність прогнозів як окремо, так і в цілому.

В процесі порівняльного аналізу результатів запропонованих та базових моделей і методів встановлено перевагу над базовими моделями на 4-8%

ВИСНОВКИ

Сукупність отриманих в дисертації результатів вирішує актуальну науково-технічну задачу підвищення достовірності семантичної сегментації та класифікації планарних зображень в системах автоматизованого скринінгу без підвищення витрат часу за рахунок розробки моделей багатозадачних нейронних мереж та методів їх навчання.

На основі проведених досліджень отримано наступні наукові результати, які складають істотний внесок у подальший розвиток теорії моделювання нейронних мереж, методів оптимізації, а також методів багатозадачного навчання.

1. Виконано огляд досліджень, що стосуються задачі автоматизованого скринінгу в різних областях; проведено аналіз моделей та методів комп’ютероного зору, насамперед, моделей глибоких нейронних мереж та методів їх навчання. Проведено аналіз методів розпізнавання, класифікації та семантичної сегментації планарних зображень, використання для цього моделей та методів багатозадачного навчання. Додатково проаналізовано метрики, що використовуються для оцінки достовірності в задачах класифікації та сегментації зображень. Виявлено, що достовірність прогнозів може бути визначена як міра Дайса-Соренсена, або F1-міра. Наведено підходи використання багатозадачного навчання для класифікації та сегментації зображень в задачах автоматизованого скринінгу.

2. Представлено параметричну модель наборів даних із зашумленою розміткою для задач сегментації та класифікації, що відповідає оцінкам моделей шуму в різних задачах автоматизованого скринінгу, а також метод контролюваної генерації як зображень, так і масок сегментації та міток класів що модифікуються відповідно до випадково обраних недоліків.

Запропоновано використання модельних наборів даних тестування достовірності моделей та методів навчання глибоких нейронних мереж в задачах класифікації та семантичної сегментації. Тестування виконується в контролюваних умовах за допомогою введення зашумлення в розмітку лише тренувального набору даних, в той час як тестувальних набір даних залишається з точною розміткою.

3. Запропоновано модель багатозадачної нейронної мережі, що дозволяє вирішувати задачі семантичної сегментації та класифікації для аналізу планарних зображень. Виявлено, що окрім загальної архітектури моделі, критичними для коректної роботи запропонованих методів є структури декодерів для кожної з задач, зокрема, нормалізація їх внутрішніх представлень.

На основі запропонованої моделі багатозадачної нейронної мережі розроблено метод багатозадачного навчання, що в умовах частково-помилкової розмітки дозволяє зменшити вплив неправильно розмічених прикладів на процес навчання за рахунок їх ефективної фільтрації безпосередньо під час навчання. Запропоновано метод автоматичної фільтрації помилкової розмітки, що спирається на відсіювання градієнтів в точках, що відповідають занадто високим значенням функції втрат.

Запропоновано використання задачі класифікації як більш загальної до задачі семантичної сегментації при багатозадачному навчанні для оновлення параметрів енкодера в точках, що були помилково відфільтровані як зашумлені. Удосконалено метод об'єднання задач сегментації та класифікації на етапі прогнозування для зменшення кількості хибно-позитивних результатів.

На основі двох представлених методів, розроблено метод навчання з частковим залученням вчителя та метод пост-обробки, що дозволяє виконувати локалізацію важливих для класифікації ознак на вихідному зображені, що є корисним під час інтерпретації прогнозів моделей.

4. В четвертому розділі проведено експериментальні дослідження на синтетичних даних, що були згенеровані за допомогою моделі, а також на реальних даних з різних задач автоматизованого скринінгу.

Представлено архітектуру інструментальних засобів для реалізації запропонованих моделей, а також методів її навчання та прогнозування результатів.

В результаті експериментів показано переваги запропонованих багатозадачних нейронних мереж та методів їх навчання над базовими версіями. Проведено дослідження впливу рівня зашумлення розмітки на результати навчання, виявлено підвищення достовірності при використанні запропонованих методів для різних рівнів зашумлення. Проведено аналіз внеску окремих компонентів запропонованих моделей нейронних і методів їх навчання, показано їх позитивний вплив на достовірність прогнозів як окремо, так і в цілому.

В процесі порівняльного аналізу результатів запропонованих та базових моделей і методів встановлено перевагу над базовими моделями на 4-8%

Розроблені методи отримали **впровадження** в програмний продукт SafetyRadar компанії *VI Tech Lab*, програмні продукти компанії «ПЛАНЕТА ЮГ» та в навчальний процес кафедри Інформаційних систем Інституту Комп’ютерних систем Одеського Національного Політехнічного Університету.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- [1] Wood C. What is screening? The background, definitions and criteria. — 2017. — 12. — Vol. 3.
- [2] Segmentation for Classification of Screening Pancreatic Neuroendocrine Tumors. — 2020. — 2004.02021.
- [3] Hartley R., Zisserman A. Multiple View Geometry in Computer Vision. — 2 ed. — USA : Cambridge University Press, 2003. — ISBN: 0521540518.
- [4] Melanoma - Statistics. — 2012. — June. — online; accessed: <https://www.cancer.net/cancer-types/melanoma/statistics> (online; accessed: 2021-01-17).
- [5] Mustafa S., Kimura A. A SVM-based diagnosis of melanoma using only useful image features // 2018 International Workshop on Advanced Image Technology (IWAIT). — 2018. — P. 1–4.
- [6] Deep neural networks are superior to dermatologists in melanoma image classification / Brinker T. J., Hekler A., Enk A. H., Berking C., Haferkamp S., Hauschild A., Weichenthal M., Klode J., Schadendorf D., Holland-Letz T., von Kalle C., Fröhling S., Schilling B., and Utikal J. S. // European Journal of Cancer. — 2019. — Vol. 119. — P. 11–17. — Access mode: <https://www.sciencedirect.com/science/article/pii/S0959804919303491>.
- [7] Deep Residual Learning for Image Recognition. — 2015. — 1512.03385.
- [8] Deep Learning Ensembles for Melanoma Recognition in Dermoscopy Images. — 2016. — 1610.04662.
- [9] DePicT Melanoma Deep-CLASS: a deep convolutional neural networks approach to classify skin lesion images / Nasiri S., Helsper J., Jung M.,

- and Fathi M. // BMC Bioinformatics. — 2020. — Mar. — Vol. 21, no. 2. — P. 84. — Access mode: <https://doi.org/10.1186/s12859-020-3351-y>.
- [10] An End-to-End Multi-Task Deep Learning Framework for Skin Lesion Analysis / Song L., Lin J., Wang Z. J., and Wang H. // IEEE Journal of Biomedical and Health Informatics. — 2020. — Vol. 24, no. 10. — P. 2912–2921.
- [11] Lesion Attributes Segmentation for Melanoma Detection with Multi-Task U-Net / Chen E. Z., Dong X., Li X., Jiang H., Rong R., and Wu J. // 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019). — 2019. — P. 485–488.
- [12] A Novel Multi-task Deep Learning Model for Skin Lesion Segmentation and Classification. — 2017. — 1703.01025.
- [13] Priya R., Aruna P. SVM and Neural Network based Diagnosis of Diabetic Retinopathy. — 2012. — Access mode: https://pdfs.semanticscholar.org/78d6/a826f49f65d715ef6533303b62adc83d7a1b.pdf?_ga=2.3810382985.1571867892.
- [14] Application of Machine Learning to Classify Diabetic Retinopathy / Conde P., de la Calleja J., Medina M., and Benitez Ruiz A. B. — 2012. — 7.
- [15] Harry Pratt Frans Coenen D. M. B. S. P. H. Y. Z. Convolutional Neural Networks for Diabetic Retinopathy. — 2016. — Access mode: <https://doi.org/10.1016/j.procs.2016.07.014>.
- [16] Carson Lam Darvin Yi M. G., Lindsey T. Automated Detection of Diabetic Retinopathy using Deep Learning. — 2018. — Access mode: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5961805/>.
- [17] Yung-Hui Li Nai-Ning Yeh S.-J. C., Chung Y.-C. Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network. — 2019. — Access mode: <https://doi.org/10.1155/2019/6142839>.

- [18] Asiri N., Hussain M., Aboalsamh H. A. Deep Learning based Computer-Aided Diagnosis Systems for Diabetic Retinopathy: A Survey // CoRR. — 2018. — Vol. abs/1811.01238. — 1811.01238.
- [19] Hagos M. T., Kant S. Transfer Learning based Detection of Diabetic Retinopathy from Small Dataset // CoRR. — 2019. — Vol. abs/1905.07203. — 1905.07203.
- [20] Rubina Sarki Sandra Michalska K. A. H. W. Y. Z. Convolutional neural networks for mild diabetic retinopathy detection: an experimental study // bioRxiv. — 2019. — Access mode: <https://www.biorxiv.org/content/10.1101/763136v1>.
- [21] Cloud and Cloud Shadow Detection for Landsat Images: The Fundamental Basis for Analyzing Landsat Time Series / Zhu Z., Qiu S., He B., and Deng C. — 2018. — 05. — ISBN: 9781315166636.
- [22] Harb M., Gamba P., Dell'Acqua F. Automatic Delineation of Clouds and Their Shadows in Landsat and CBERS (HRCC) Data // IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. — 2016. — Vol. 9, no. 4. — P. 1532–1542.
- [23] Hu X., Wang Y., Shan J. Automatic Recognition of Cloud Images by Using Visual Saliency Features // IEEE Geoscience and Remote Sensing Letters. — 2015. — Vol. 12, no. 8. — P. 1760–1764.
- [24] Ozkan S., Efendio?lu M., Demirpolat C. Cloud Detection from RGB Color Remote Sensing Images with Deep Pyramid Networks. — 2018. — 07. — P. 6939–6942.
- [25] Li F.-F., Karpathy A., Johnson J. CS231n: Convolutional Neural Networks for Visual Recognition 2016. — Access mode: <http://cs231n.stanford.edu/>.
- [26] Datta L. A Survey on Activation Functions and their relation with Xavier and He Normal Initialization. — 2020. — 2004.06632.

- [27] Bio-inspired Neurocomputing. — 2021. — Access mode: <http://dx.doi.org/10.1007/978-981-15-5495-7>.
- [28] Efficient hardware implementation of the hyperbolic tangent sigmoid function / Namin A., Leboeuf K., Muscedere R., Wu H., and Ahmadi M. — 2009. — 06. — P. 2117 – 2120.
- [29] Agarap A. F. Deep Learning using Rectified Linear Units (ReLU). — 2019. — 1803.08375.
- [30] Empirical Evaluation of Rectified Activations in Convolutional Network. — 2015. — 1505.00853.
- [31] Improving deep neural networks using softplus units / Zheng H., Yang Z., Liu W.-J., Liang J., and Li Y. — 2015. — 07. — P. 1–4.
- [32] Chen J. Combinatorially Generated Piecewise Activation Functions. — 2016. — 1605.05216.
- [33] Ramachandran P., Zoph B., Le Q. V. Searching for Activation Functions. — 2017. — 1710.05941.
- [34] Misra D. Mish: A Self Regularized Non-Monotonic Activation Function. — 2020. — 1908.08681.
- [35] Learning Pooling for Convolutional Neural Network / Sun M., Song Z., Jiang X., Pan J., and Pang Y. // Neurocomputing. — 2017. — Vol. 224. — P. 96–104. — Access mode: <https://www.sciencedirect.com/science/article/pii/S0925231216312905>.
- [36] Attention pooling-based convolutional neural network for sentence modelling / Er M. J., Zhang Y., Wang N., and Pratama M. // Information Sciences. — 2016. — Vol. 373. — P. 388–403. — Access mode: <https://www.sciencedirect.com/science/article/pii/S0020025516306673>.
- [37] Mixed Pooling for Convolutional Neural Networks / Yu D., Wang H., Chen P., and Wei Z. // Rough Sets and Knowledge Technology. — Springer International Publishing. — 2014. — P. 364–375.

- [38] Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. — 2015. — 1502.03167.
- [39] Ba J. L., Kiros J. R., Hinton G. E. Layer Normalization. — 2016. — 1607.06450.
- [40] Ulyanov D., Vedaldi A., Lempitsky V. Instance Normalization: The Missing Ingredient for Fast Stylization. — 2017. — 1607.08022.
- [41] Wu Y., He K. Group Normalization. — 2018. — 1803.08494.
- [42] Imagenet: A large-scale hierarchical image database / Deng J., Dong W., Socher R., Li L.-J., Li K., and Fei-Fei L. // 2009 IEEE conference on computer vision and pattern recognition / Ieee. — 2009. — P. 248–255.
- [43] Gradient-based learning applied to document recognition / Lecun Y., Bottou L., Bengio Y., and Haffner P. // Proceedings of the IEEE. — 1998. — Vol. 86, no. 11. — P. 2278–2324.
- [44] Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks // Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1. — Red Hook, NY, USA : Curran Associates Inc. — 2012. — NIPS’12. — P. 1097–1105.
- [45] Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. — 2015. — 1409.1556.
- [46] Going Deeper with Convolutions. — 2014. — 1409.4842.
- [47] Rethinking the Inception Architecture for Computer Vision. — 2015. — 1512.00567.
- [48] Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. — 2016. — 1602.07261.
- [49] Densely Connected Convolutional Networks. — 2018. — 1608.06993.
- [50] Memory-Efficient Implementation of DenseNets. — 2017. — 1707.06990.

- [51] Aggregated Residual Transformations for Deep Neural Networks. — 2017. — 1611.05431.
- [52] Long J., Shelhamer E., Darrell T. Fully Convolutional Networks for Semantic Segmentation. — 2015. — 1411.4038.
- [53] Badrinarayanan V., Kendall A., Cipolla R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. — 2016. — 1511.00561.
- [54] Ronneberger O., Fischer P., Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. — 2015. — 1505.04597.
- [55] Chaurasia A., Culurciello E. LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation // CoRR. — 2017. — Vol. abs/1707.03718. — arXiv : 1707.03718.
- [56] UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation. — 2020. — 1912.05074.
- [57] Ibtehaz N., Rahman M. S. MultiResUNet?: Rethinking the U-Net architecture for multimodal biomedical image segmentation // Neural Networks. — 2020. — Jan. — Vol. 121. — P. 74–87. — Access mode: <http://dx.doi.org/10.1016/j.neunet.2019.08.025>.
- [58] Lou A., Guan S., Loew M. DC-UNet: Rethinking the U-Net Architecture with Dual Channel Efficient CNN for Medical Images Segmentation. — 2020. — 2006.00414.
- [59] Feature Pyramid Networks for Object Detection. — 2017. — 1612.03144.
- [60] Feature Pyramid Network for Multi-Class Land Segmentation. — 2018. — 1806.03510.
- [61] Ripley B. D. Pattern Recognition and Neural Networks. — Cambridge University Press, 1996.
- [62] Srivastava N., Mansimov E., Salakhutdinov R. Unsupervised Learning of Video Representations using LSTMs. — 2016. — 1502.04681.

- [63] segmentation-moving-MNIST. — Режим доступу: <https://github.com/RoshanRane/segmentation-moving-MNIST> (дата звернення: 2021-01-17).
- [64] seg-mnist-dataset: Generates segmentation examples from the MNIST and similar datasets. — Режим доступу: <https://github.com/williford/seg-mnist-dataset> (дата звернення: 2021-01-17).
- [65] Caruana R. Multitask Learning // Learning to Learn. — Boston, MA : Springer US, 1998. — P. 95–133. — ISBN: 978-1-4615-5529-2.
- [66] Wu S., Zhang H. R., R? C. Understanding and Improving Information Transfer in Multi-Task Learning. — 2020. — 2005.00944.
- [67] Pilault J., Elhattami A., Pal C. Conditionally Adaptive Multi-Task Learning: Improving Transfer Learning in NLP Using Fewer Parameters and Less Data. — 2021. — 2009.09139.
- [68] Lee G.-H., Lee S.-W. Uncertainty-Aware Mesh Decoder for High Fidelity 3D Face Reconstruction // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). — 2020. — June.
- [69] Rich feature hierarchies for accurate object detection and semantic segmentation. — 2014. — 1311.2524.
- [70] Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. — 2016. — 1506.01497.
- [71] Mask R-CNN. — 2018. — 1703.06870.
- [72] Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks / Zhang K., Zhang Z., Li Z., and Qiao Y. // IEEE Signal Processing Letters. — 2016. — Oct. — Vol. 23, no. 10. — P. 1499–1503. — Access mode: <http://dx.doi.org/10.1109/LSP.2016.2603342>.
- [73] Gradient Surgery for Multi-Task Learning. — 2020. — 2001.06782.
- [74] Just Pick a Sign: Optimizing Deep Multitask Models with Gradient Sign Dropout. — 2020. — 2010.06808.

- [75] Mahapatra D., Rajan V. Multi-Task Learning with User Preferences: Gradient Descent with Controlled Ascent in Pareto Optimization // Proceedings of the 37th International Conference on Machine Learning / ed. by III H. D., Singh A. — PMLR. — 2020. — 13–18 Jul. — Vol. 119 of Proceedings of Machine Learning Research. — P. 6597–6607. — Access mode: <https://proceedings.mlr.press/v119/mahapatra20a.html>.
- [76] Cross-stitch Networks for Multi-task Learning. — 2016. — 1604.03539.
- [77] NDDR-CNN: Layerwise Feature Fusing in Multi-Task CNNs by Neural Discriminative Dimensionality Reduction. — 2019. — 1801.08297.
- [78] PAD-Net: Multi-Tasks Guided Prediction-and-Distillation Network for Simultaneous Depth Estimation and Scene Parsing. — 2018. — 1805.04409.
- [79] SegTHOR: Segmentation of Thoracic Organs at Risk in CT images. — 2019. — 1912.05950.
- [80] A Novel Multi-task Deep Learning Model for Skin Lesion Segmentation and Classification. — 2017. — 1703.01025.
- [81] Multitask Classification and Segmentation for Cancer Diagnosis in Mammography. — 2019. — 1909.05397.
- [82] Multi-Task Learning for Segmentation of Building Footprints with Deep Neural Networks. — 2017. — 1709.05932.
- [83] Weakly- and Semi-Supervised Object Detection with Expectation-Maximization Algorithm. — 2017. — 1702.08740.
- [84] Frenay B., Verleysen M. Classification in the Presence of Label Noise: A Survey // IEEE Transactions on Neural Networks and Learning Systems. — 2014. — Vol. 25, no. 5. — P. 845–869.
- [85] Algan G., ?lkay Ulusoy. Label Noise Types and Their Effects on Deep Learning. — 2020. — 2003.10471.
- [86] Re-labeling ImageNet: from Single to Multi-Labels, from Global to Localized Labels. — 2021. — 2101.05022.

- [87] Towards annotation-efficient segmentation via image-to-image translation. — 2021. — 1904.01636.
- [88] Intra- and inter-rater agreement between an ophthalmologist and mid-level ophthalmic personnel to diagnose retinal diseases based on fundus photographs at a primary eye center in Nepal: the Bhaktapur Retina Study / Thapa R., Bajimaya S., Bouman R., Paudyal G., Khanal S., Tan S., Thapa S. S. and van Rens G. // BMC ophthalmology. — 2016. — Jul. — Vol. 16. — P. 112–112. — online; accessed: <https://pubmed.ncbi.nlm.nih.gov/27430579>.
- [89] Song H., Kim M., Lee J.-G. SELFIE: Refurbishing Unclean Samples for Robust Deep Learning // ICML. — 2019.
- [90] LeCun Y., Cortes C. MNIST handwritten digit database. — 2010. — Access mode: <http://yann.lecun.com/exdb/mnist/>.
- [91] Xiao H., Rasul K., Vollgraf R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. — 2017. — 1708.07747.
- [92] Howard J. imagenette. — Access mode: <https://github.com/fastai/imagenette/>.
- [93] Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. — 2015. — 1502.01852.
- [94] Odena A., Dumoulin V., Olah C. Deconvolution and Checkerboard Artifacts // Distill. — 2016. — Access mode: <http://distill.pub/2016/deconv-checkerboard>.
- [95] Pedestrian Detection aided by Deep Learning Semantic Tasks. — 2014. — 1412.0069.
- [96] Maron O., Lozano-Pérez T. A Framework for Multiple-Instance Learning // Advances in Neural Information Processing Systems / ed. by Jordan M., Kearns M., Solla S. — MIT Press. — 1998. — Vol. 10. — Access mode:

- <https://proceedings.neurips.cc/paper/1997/file/82965d4ed8150294d4330ace00821d77-Paper.pdf>.
- [97] Shorten C., Khoshgoftaar T. M. A survey on Image Data Augmentation for Deep Learning // Journal of Big Data. — 2019. — Jul. — Vol. 6, no. 1. — P. 60. — Access mode: <https://doi.org/10.1186/s40537-019-0197-0>.
- [98] Ng A. Y. Feature Selection, L_1 vs. L_2 Regularization, and Rotational Invariance // Proceedings of the Twenty-First International Conference on Machine Learning. — New York, NY, USA : Association for Computing Machinery. — 2004. — ICML '04. — P. 78. — Access mode: <https://doi.org/10.1145/1015330.1015435>.
- [99] Caruana R., Lawrence S., Giles L. Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping // Proceedings of the 13th International Conference on Neural Information Processing Systems. — Cambridge, MA, USA : MIT Press. — 2000. — NIPS'00. — P. 381–387.
- [100] ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. — 2019. — 1811.12231.
- [101] Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization. — 2017. — 1412.6980.
- [102] Yang X. An Overview of the Attention Mechanisms in Computer Vision // Journal of Physics: Conference Series. — 2020. — dec. — Vol. 1693. — P. 012173. — Access mode: <https://doi.org/10.1088/1742-6596/1693/1/012173>.
- [103] Облачные сервисы – Amazon Web Services (AWS). — Режим доступу: <https://aws.amazon.com/ru/> (дата звернення: 2021-01-17).
- [104] PyTorch. — Режим доступу: <https://www.pytorch.org> (дата звернення: 2021-01-17).
- [105] Kolesnikov S. Catalyst. — Режим доступу: <https://catalyst-team.com/> (дата звернення: 2021-01-17).

- [106] Amazon EC2. — Режим доступу: <https://aws.amazon.com/ru/ec2/> (дата звернення: 2021-01-17).
- [107] Amazon SageMaker – Machine Learning – Amazon Web Services. — Режим доступу: <https://aws.amazon.com/ru/sagemaker/> (дата звернення: 2021-01-17).
- [108] Облачное объектное хранилище | Сохранение и извлечение данных из любого места | Amazon Simple Storage Service (S3). — Режим доступу: <https://aws.amazon.com/ru/s3/> (дата звернення: 2021-01-17).
- [109] McInnes L., Healy J., Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. — 2020. — 1802.03426.
- [110] On the Variance of the Adaptive Learning Rate and Beyond. — 2020. — 1908.03265.
- [111] Loshchilov I., Hutter F. SGDR: Stochastic Gradient Descent with Warm Restarts. — 2017. — 1608.03983.
- [112] Berrar D. Cross-Validation. — 2018. — 01. — ISBN: 9780128096338.
- [113] Ordinal Regression as Structured Classification. — 2019. — 1905.13658.
- [114] Indian Diabetic Retinopathy Image Dataset (IDRiD). — 2018. — Access mode: <https://dx.doi.org/10.21227/H25W98>.
- [115] Feedback on a publicly distributed database: the Messidor database / Decenci?re E., Zhang X., Cazuguel G., Lay B., Cochener B., Trone C., Gain P., Ordonez R., Massin P., Erginay A., Charlton B. and Klein J.-C. // Image Analysis & Stereology. — 2014. — Aug. — Vol. 33, no. 3. — P. 231–234. — online; accessed: <http://www.ias-iss.org/ojs/IAS/article/view/1155>.
- [116] MESSIDOR-2 DR Grades. — online; accessed: <https://kaggle.com/googlebrain/messidor2-dr-grades> (online; accessed: 2021-01-17).
- [117] APTOS 2019 Blindness Detection. — Режим доступу: <https://kaggle.com/c/aptos2019-blindness-detection> (дата звернення: 2021-01-17).

- [118] Focal Loss for Dense Object Detection. — 2018. — 1708.02002.
- [119] SIIM-ISIC Melanoma Classification. — Режим доступу: <https://kaggle.com/c/siim-isic-melanoma-classification> (дата звернення: 2021-01-17).
- [120] SIIM-ISIC Melanoma Classification. — Режим доступу: <https://kaggle.com/c/siim-isic-melanoma-classification> (дата звернення: 2021-01-17).

Додаток А. Список публікацій здобувача за темою дисертації

- controller for vision-based car driving // XVIII International Conference on Data Science and Intelligent Analysis of Information / Springer. 2018. C. 20–29. (Index Copernicus)
- https://www.springerprofessional.de/en/race-from-pixels-evolving-neural-network-controller-for-vision-b/16003024*
7. Tymchenko B, Hramatik A., Tulchyi H., Antoshchuk S. Making money: Evolving neural network for stock prediction // VI українсько-німецька конференція «Інформатика. Культура. Техніка». 2018. C. 34-35
 8. Tymchenko B., Antoshchuk S. Evolution strategy for policy search in robotics // Сучасні Інформаційні Технології / Одеський Національний Політехнічний Університет. 2018.
- http://dspace.opu.ua/jspui/handle/123456789/8056*
9. Tymchenko B., Halchonkov O. Online lane detection algorithm for line scan camera // Сучасні Інформаційні Технології / Одеський Національний Політехнічний Університет. 2017. C. 86-89
- http://dspace.opu.ua/jspui/handle/123456789/3351*
10. Tymchenko B. Global position system sensor model for robotics simulator // Праці Одеського політехнічного університету. 2017. № 3. C. 88–93.
- http://dspace.opu.ua/jspui/handle/123456789/7939*
11. Tymchenko B., Samodelok V., Putilina D., Galchonkov O. The robust control system for skid elimination in dynamic road environments // Електротехнічні та комп’ютерні системи. 2016. № 23. C. 107–112.
- http://dspace.opu.ua/jspui/handle/123456789/1176*
12. Komleva N., Cherneha K., Tymchenko B., Komlevoy O. Intellectual approach application for pulmonary diagnosis 2016 IEEE First International Conference on Data Stream Mining Processing (DSMP). 2016. C. 48–52.
- https://ieeexplore.ieee.org/document/7583505/*

13. Cherneha K., Tymchenko B., Komleva N. Decision support system for automated medical diagnostics // Електротехнічні та комп'ютерні системи. 2016. № 23. С. 65–72. (Index Copernicus)
<http://dspace.opu.ua/xmlui/handle/123456789/1140>

Додаток Б. Акт впровадження результатів дисертації в компанії VITech Lab



Товариство з Обмеженою Відповідальністю "Ві.Ай.Тек-Плюс"
ЕДРПОУ № 35664839 р/р 2600000132728 в АТ "ОТП Банк" м. Київ
МФО 300528

вул. Жуковського, 11/6
Львів, 79044
Україна
+38(032) 237 02 56
office@vitech.com.ua
www.vitech.com.ua

Вих. № 22/09/01/01
від "13" вересня 2021 р.

АКТ

впровадження практичних результатів дисертаційної роботи
аспіранта PhD кафедри інформаційних систем
Державного Університету "Одеська Політехніка"
Тимченко Бориса Ігоровича

Комісія у складі:

голови комісії директора Сарапін Віктор

та членів комісії:

керівник R&D відділу Сподарець Дмитро

склала цей акт про те, що наукові результати аспіранта PhD кафедри інформаційних систем Державного Університету "Одеська Політехніка" Тимченко Б.І., а саме метод багатозадачного навчання штучних нейронних мереж для роботи в умовах частково помилкової розмітки навчального набору даних та метод об'єднання семантично-близьких задач на етапі прогнозування у вигляді технічного опису і програмного забезпечення впроваджено при побудові системи автоматичного скрінінгу порушень порядку носіння засобів індивідуального захисту SafetyRadar у 2020-2021 роках.

В результаті, аналіз, моделювання та практичне застосування наданих матеріалів дало можливість на 17% підвищити достовірність розпізнавання людей та засобів індивідуального захисту на зображеннях без зниження оперативності.

Даний акт не є підставою для фінансових розрахунків.

Голова комісії
Директор

Члени комісії
керівник R&D відділу



Сарапін В.Я.

Сподарець Д.В.

**Додаток В. Акт впровадження результатів дисертації в
навчальний процес ОНПУ**

**Додаток Г. Акт впровадження результатів компанії
ПЛАНЕТА ЮГ**



Україна, 65020, г.Одеса, ул. 10 Апреля,16
Почтовий адрес: 65020, г. Одеса, ул. 10 Апреля, 16
Тел.: (048) 705-74-96, E-mail: office@ps.od.ua

Р/с UA083287040000026002060900287 в АО КБ "ПриватБанк"
г. Одеса МФО 328704, код ОКПО 30747868, індивідуальний
налоговий номер плательщика НДС 307478615538,
свідчительство № 23235396

Вих. № 60/1
від " 15 " 09 2021р.

АКТ
впровадження практичних результатів дисертаційної роботи
аспіранта PhD кафедри інформаційних систем
Державного Університету "Одеська Політехніка"
Тимченко Бориса Ігоровича

Комісія у складі:
голова комісії, директор — Коробко Вікторія Олександрівна

та членів комісії:
Засновник, кандидат технічних наук — Сиропятов Олександр Арсенійович
Технічний директор — Каліновський Роман Юрійович

склала цей акт про те, що наукові результати аспіранта PhD кафедри інформаційних систем Державного Університету "Одеська Політехніка" Тимченко Б.І., а саме методи:

1. метод багатозадачного навчання штучних нейронних мереж для роботи в умовах частково помилкової розмітки навчального набору даних,
 2. метод об'єднання семантично-близьких задач на етапі прогнозування,
- отримані у вигляді технічного опису і програмного забезпечення були впроваджені нами у 2020-2021 роках в підсистему автоматизованого скринінгу порушення периметру безпеки при будівництві систем багатокамерного відеоспостереження на підприємствах України з великою та розподіленою територією, де наша компанія діяла в якості системного інтегратора зі встановлення та обслуговування систем відеобезпеки.

В результаті впровадження вищезазначених методів отримані наступні висновки:

Аналіз, моделювання та практичне застосування наданих матеріалів надало можливості на 10% підвищити достовірність автоматизованого розпізнавання людей, що порушують периметр безпеки, на зображеннях з відеокамер, без зниження оперативності самого процесу відеоспостереження.

Даний акт не є підставою для фінансових розрахунків.

Голова комісії
Директор

Члени комісії
Засновник

Технічний директор

Коробко В.О.

Сиропятов О.А.

Каліновський Р.Ю.



Додаток Д. Вихідні коди програмних інструментальних засобів

```

@registry.Criterion
class NoLoss(nn.Module):
    def __init__(self):
        super(NoLoss, self).__init__()

    def forward(self, inputs, targets):
        return inputs

    from .model import Net, OneClassNet, MaskOneClassNet
    import math
    from typing import Optional

    import timm
    import torch
    from efficientnet_pytorch import EfficientNet
    from torch import nn
    from torch.nn import functional as F
    from torch.nn.parameter import Parameter

    class Flatten(nn.Module):
        def forward(self, input):
            return input.view(input.size(0), -1)

    class GlobalStdPool2d(nn.Module):
        def __init__(self, flatten=False):

```

```
super(GlobalStdPool2d, self).__init__()
self.flatten = flatten

def forward(self, x):
    if self.flatten:
        in_size = x.size()
        return x.view((in_size[0], in_size[1], -1)).std(dim=2)
    else:
        return x.view(x.size(0), x.size(1), -1).std(-1).view(x.size(0), x.size(1))

class AdaptiveConcatPool2d(nn.Module):
    def __init__(self, sz=None):
        super().__init__()
        sz = sz or (1, 1)
        self.ap = nn.AdaptiveAvgPool2d(sz)
        self.mp = nn.AdaptiveMaxPool2d(sz)

    def forward(self, x): return torch.cat([self.mp(x), self.ap(x)], 1)

class MeanStdConcatPool2d(nn.Module):
    def __init__(self):
        super().__init__()
        sz = (1, 1)
        self.ap = nn.AdaptiveAvgPool2d(sz)
        self.std = GlobalStdPool2d(flatten=False)

    def forward(self, x):
```

```
return torch.cat([self.std(x), self.ap(x)], 1)

class ArcMarginProduct(nn.Module):
    """Implement of large margin arc distance: :
    Args:
        in_features: size of each input sample
        out_features: size of each output sample
        s: norm of input feature
        m: margin
        cos(theta + m)
    """
    def __init__(self, in_features, out_features):
        super(ArcMarginProduct, self).__init__()
        self.weight = Parameter(torch.FloatTensor(out_features, in_features))
        # nn.init.xavier_uniform_(self.weight)
        self.reset_parameters()

    def reset_parameters(self):
        stdv = 1. / math.sqrt(self.weight.size(1))
        self.weight.data.uniform_(-stdv, stdv)

    def forward(self, features):
        cosine = F.linear(F.normalize(features), F.normalize(self.weight.cuda()))
        return cosine

class FeatureExtractorHead(nn.Module):
```

```
def __init__(self, extractor, classifier):  
    super().__init__()  
  
    assert extractor is not None  
    assert classifier is not None  
  
    self.fe = extractor  
    self.cls = classifier  
  
def forward(self, x):  
    features = self.fe(x)  
    output = self.cls(features)  
    return features, output  
  
def freeze(model):  
    for param in model.parameters():  
        param.requires_grad = False  
  
def unfreeze(model):  
    for param in model.parameters():  
        param.requires_grad = True  
  
def get_classifier_head(input_dim, output_dim, name="concat"):  
    print('Using classification head ', name)  
    fe = None  
    cls = None
```

```
if name == "autoencoder":  
    pass  
if name == "avg_arcface":  
    fe = nn.Sequential(  
        nn.AdaptiveAvgPool2d((1, 1)),  
        Flatten(),  
  
        nn.BatchNorm1d(input_dim),  
        nn.Linear(input_dim, 512),  
        nn.LeakyReLU(),  
  
        nn.BatchNorm1d(512),  
        nn.Dropout(p=0.5),  
  
        nn.Linear(512, 512),  
        nn.BatchNorm1d(512)  
    )  
    cls = ArcMarginProduct(512, output_dim)  
  
if name == "arcface":  
    fe = nn.Sequential(  
        AdaptiveConcatPool2d((1, 1)),  
        Flatten(),  
  
        nn.BatchNorm1d(input_dim),  
        nn.Dropout(p=0.25),  
  
        nn.Linear(input_dim, 512),  
        nn.LeakyReLU(),
```

```
nn.BatchNorm1d(512),  
nn.Dropout(p=0.5),  
  
nn.Linear(512, 512),  
nn.BatchNorm1d(512)  
)  
cls = ArcMarginProduct(512, output_dim)  
  
if name == "std":  
    fe = nn.Sequential(  
        GlobalStdPool2d(flatten=True),  
  
        nn.BatchNorm1d(input_dim),  
        nn.Linear(input_dim, 512),  
        nn.LeakyReLU(),  
  
        nn.BatchNorm1d(512),  
        nn.Dropout(0.5),  
        nn.Linear(512, 512),  
        nn.LeakyReLU()  
)  
    cls = nn.Linear(512, output_dim)  
  
if name == "meanstd":  
    fe = nn.Sequential(  
        MeanStdConcatPool2d(),  
        Flatten(),  
  
        nn.BatchNorm1d(input_dim),
```

```
nn.Linear(input_dim, 512) ,  
nn.LeakyReLU() ,  
  
nn.BatchNorm1d(512) ,  
nn.Dropout(0.5) ,  
nn.Linear(512, 512) ,  
nn.LeakyReLU()  
)  
cls = nn.Linear(512, output_dim)  
  
if name == "concat2":  
    fe = nn.Sequential(  
        AdaptiveConcatPool2d((1, 1)) ,  
        Flatten() ,  
  
        nn.BatchNorm1d(input_dim) ,  
        nn.Linear(input_dim, 512) ,  
        nn.LeakyReLU() ,  
  
        nn.BatchNorm1d(512) ,  
        nn.Linear(512, 512) ,  
        nn.LeakyReLU() ,  
  
        nn.BatchNorm1d(512) ,  
        nn.Dropout(p=0.5)  
)  
    cls = nn.Linear(512, output_dim)  
  
if name == "concat":
```

```
fe = nn.Sequential(
    AdaptiveConcatPool2d((1, 1)),
    Flatten(),
    nn.BatchNorm1d(input_dim),
    nn.Linear(input_dim, 256),
    nn.LeakyReLU(),
    nn.Dropout(0.5)
)
cls = nn.Linear(256, output_dim)

if name == "avg2":
    fe = nn.Sequential(
        nn.AdaptiveAvgPool2d((1, 1)),
        Flatten(),
        nn.BatchNorm1d(input_dim),
        nn.Linear(input_dim, 512),
        nn.LeakyReLU(),
        nn.BatchNorm1d(512),
        nn.Dropout(0.5),
        nn.Linear(512, 512),
        nn.LeakyReLU()
)
cls = nn.Linear(512, output_dim)

if name == "avg":
    fe = nn.Sequential(
        nn.AdaptiveAvgPool2d((1, 1)),
```

```
Flatten() ,  
nn.BatchNorm1d(input_dim) ,  
nn.Dropout(0.5) ,  
nn.Linear(input_dim, 256) ,  
nn.LeakyReLU()  
)  
cls = nn.Linear(256, output_dim)  
  
if name == "max":  
    fe = nn.Sequential(  
        nn.AdaptiveMaxPool2d((1, 1)) ,  
        Flatten() ,  
        nn.BatchNorm1d(input_dim) ,  
        nn.Linear(input_dim, 256) ,  
        nn.LeakyReLU() ,  
        nn.Dropout(0.5)  
)  
    cls = nn.Linear(256, output_dim)  
  
if name == "simple_avg":  
    fe = nn.Sequential(  
        nn.AdaptiveAvgPool2d((1, 1)) ,  
        Flatten() ,  
        nn.BatchNorm1d(input_dim)  
)  
    cls = nn.Linear(input_dim, output_dim)  
  
if name == "one_layer_fc":  
    fe = nn.Sequential(  
        nn.Linear(input_dim, 256) ,  
        nn.LeakyReLU() ,  
        nn.Dropout(0.5) ,  
        nn.BatchNorm1d(256) ,  
        nn.Linear(256, output_dim)
```

```
nn.AdaptiveAvgPool2d((1, 1)),
Flatten(),
)
cls = nn.Linear(input_dim, output_dim)

return FeatureExtractorHead(extractor=fe, classifier=cls)

class Net(nn.Module):
    """
efficientnet-b0-224
efficientnet-b1-240
efficientnet-b2-260
efficientnet-b3-300
efficientnet-b4-380
efficientnet-b5-456
efficientnet-b6-528
efficientnet-b7-600
    """

    def __init__(
        self,
        num_classes: int,
        n_features: int = 2048,
        arch: str = "se_resnext50_32x4d",
        class_head_name: str = "one_layer_fc",
        init_bias: Optional[float] = None):
        super().__init__()
```

```
self.arch = arch
self.n_features = n_features
self.init_bias = init_bias

if "lukemelas" in arch:
    arch = arch.split("_")[1]
self.body = EfficientNet.from_pretrained(arch)
else:
    self.body = timm.create_model(arch, pretrained=True)

self.classification_head = get_classifier_head(n_features, num_classes)

self.fill_bias()

def fill_bias(self):
    bias = self.init_bias
    if bias is not None and hasattr(self.classification_head.cls, "bias"):
        print(f"Initializing bias with {bias}")
        nn.init.constant_(self.classification_head.cls.bias, bias)

def forward(self, x):
    if "lukemelas" in self.arch:
        x = self.body.extract_features(x)
    else:
        x = self.body.forward_features(x)

    features, classification_ohe = self.classification_head(x)
    classification_ohe_softmax = F.softmax(classification_ohe, dim=1)
    classification_logits = torch.argmax(classification_ohe_softmax, axis=1)
```



```
if "lukemelas" in self.arch:
    x = self.body.extract_features(x)
else:
    x = self.body.forward_features(x)

features, logits = self.classification_head(x)
ret = {
    'backbone_features': x,
    'features': features,
    'logits': logits,
}
return ret

def predict(self, x):
    res = self.forward(x)

    return res["features"], res["backbone_features"], res["logits"]

class MaskOneClassNet(Net):
    def __init__(self, num_classes: int = 1,
                 n_features: int = 2048,
                 arch: str = "se_resnext50_32x4d",
                 class_head_name: str = "one_layer_fc",
                 init_bias: Optional[float] = None
                 ):
        assert num_classes == 1
```

```
super().__init__(num_classes=num_classes,
n_features=n_features,
arch=arch,
class_head_name=class_head_name,
init_bias=init_bias)

self.mask_branch = nn.Sequential(
nn.Dropout2d(0.5),

nn.Conv2d(n_features, 256, kernel_size=1),
nn.BatchNorm2d(256),
nn.ReLU(inplace=True),

nn.Conv2d(256, 256, kernel_size=1),
nn.BatchNorm2d(256),
nn.ReLU(inplace=True),

nn.Conv2d(256, 1, kernel_size=1),
)

def forward(self, x):
if "lukemelas" in self.arch:
x = self.body.extract_features(x)
else:
x = self.body.forward_features(x)

features, logits = self.classification_head(x)
mask = self.mask_branch(x).sigmoid()
```

```
melanoma = mask * logits.view(-1, 1, 1, 1)
melanoma = melanoma.sum(dim=(2, 3)) / (mask.sum(dim=(2, 3)) + 1e-4)

ret = {
    'backbone_features': x,
    'features': features,
    'logits': melanoma,
    'mask': mask
}

return ret

def predict(self, x):
    res = self.forward(x)

    return res["features"], res["backbone_features"], res["logits"]

import math
import torch
from torch.optim.optimizer import Optimizer, required

from catalyst.dl import registry

@registry.Optimizer
class RAdamMy(Optimizer):
    def __init__(self, params, lr=1e-3, betas=(0.9, 0.999), eps=1e-4, weight_decay=0):
        if not 0.0 <= lr:
            raise ValueError("Invalid learning rate: {}".format(lr))
```

```
if not 0.0 <= eps:
    raise ValueError("Invalid epsilon value: {}".format(eps))
if not 0.0 <= betas[0] < 1.0:
    raise ValueError("Invalid beta parameter at index 0: {}".format(betas[0]))
if not 0.0 <= betas[1] < 1.0:
    raise ValueError("Invalid beta parameter at index 1: {}".format(betas[1]))

defaults = dict(lr=lr, betas=betas, eps=eps, weight_decay=weight_decay)
self.buffer = [[None, None, None] for ind in range(10)]
super(RAdamMy, self).__init__(params, defaults)

def __setstate__(self, state):
    super(RAdamMy, self).__setstate__(state)

def step(self, closure=None):

    loss = None
    if closure is not None:
        loss = closure()

    for group in self.param_groups:

        for p in group["params"]:
            if p.grad is None:
                continue
            grad = p.grad.data.float()
            if grad.is_sparse:
                raise RuntimeError("RAdam does not support sparse gradients")
            ...
```

```
p_data_fp32 = p.data.float()

state = self.state[p]

if len(state) == 0:
    state["step"] = 0
    state["exp_avg"] = torch.zeros_like(p_data_fp32)
    state["exp_avg_sq"] = torch.zeros_like(p_data_fp32)
else:
    state["exp_avg"] = state["exp_avg"].type_as(p_data_fp32)
    state["exp_avg_sq"] = state["exp_avg_sq"].type_as(p_data_fp32)

exp_avg, exp_avg_sq = state["exp_avg"], state["exp_avg_sq"]
beta1, beta2 = group["betas"]

exp_avg_sq.mul_(beta2).addcmul_(1 - beta2, grad, grad)
exp_avg.mul_(beta1).add_(1 - beta1, grad)

state["step"] += 1
buffered = self.buffer[int(state["step"] % 10)]
if state["step"] == buffered[0]:
    N_sma, step_size = buffered[1], buffered[2]
else:
    buffered[0] = state["step"]
    beta2_t = beta2 ** state["step"]
    N_sma_max = 2 / (1 - beta2) - 1
    N_sma = N_sma_max - 2 * state["step"] * beta2_t / (1 - beta2_t)
    buffered[1] = N_sma
```

```

# more conservative since it's an approximated value
if N_sma >= 5:
    step_size = math.sqrt(
        (1 - beta2_t)
        * (N_sma - 4)
        / (N_sma_max - 4)
        * (N_sma - 2)
        / N_sma
        * N_sma_max
        / (N_sma_max - 2)
    ) / (1 - beta1 ** state["step"])
else:
    step_size = 1.0 / (1 - beta1 ** state["step"])
buffered[2] = step_size

if group["weight_decay"] != 0:
    p_data_fp32.add_(-group["weight_decay"] * group["lr"], p_data_fp32)

# more conservative since it's an approximated value
if N_sma >= 5:
    denom = exp_avg_sq.sqrt().add_(group["eps"])
    p_data_fp32.addcdiv_(-step_size * group["lr"], exp_avg, denom)
else:
    p_data_fp32.add_(-step_size * group["lr"], exp_avg)

p.data.copy_(p_data_fp32)

return loss

```

```
class PlainRAdam(Optimizer):

    def __init__(self, params, lr=1e-3, betas=(0.9, 0.999), eps=1e-8, weight_decay=0):
        if not 0.0 <= lr:
            raise ValueError("Invalid learning rate: {}".format(lr))
        if not 0.0 <= eps:
            raise ValueError("Invalid epsilon value: {}".format(eps))
        if not 0.0 <= betas[0] < 1.0:
            raise ValueError("Invalid beta parameter at index 0: {}".format(betas[0]))
        if not 0.0 <= betas[1] < 1.0:
            raise ValueError("Invalid beta parameter at index 1: {}".format(betas[1]))

        defaults = dict(lr=lr, betas=betas, eps=eps, weight_decay=weight_decay)

        super(PlainRAdam, self).__init__(params, defaults)

    def __setstate__(self, state):
        super(PlainRAdam, self).__setstate__(state)

    def step(self, closure=None):

        loss = None

        if closure is not None:
            loss = closure()

        for group in self.param_groups:

            for p in group["params"]:
                if p.grad is None:
                    continue
                grad = p.grad.data
                if grad.is_sparse:
                    raise RuntimeError("RAdam does not support sparse gradients")
                ... (rest of the code block)
```

```
continue

grad = p.grad.data.float()
if grad.is_sparse:
    raise RuntimeError("RAdam does not support sparse gradients")

p_data_fp32 = p.data.float()

state = self.state[p]

if len(state) == 0:
    state["step"] = 0
    state["exp_avg"] = torch.zeros_like(p_data_fp32)
    state["exp_avg_sq"] = torch.zeros_like(p_data_fp32)
else:
    state["exp_avg"] = state["exp_avg"].type_as(p_data_fp32)
    state["exp_avg_sq"] = state["exp_avg_sq"].type_as(p_data_fp32)

exp_avg, exp_avg_sq = state["exp_avg"], state["exp_avg_sq"]
beta1, beta2 = group["betas"]

exp_avg_sq.mul_(beta2).addcmul_(1 - beta2, grad, grad)
exp_avg.mul_(beta1).add_(1 - beta1, grad)

state["step"] += 1
beta2_t = beta2 ** state["step"]
N_sma_max = 2 / (1 - beta2) - 1
N_sma = N_sma_max - 2 * state["step"] * beta2_t / (1 - beta2_t)

if group["weight_decay"] != 0:
```

```

p_data_fp32.add_(-group["weight_decay"] * group["lr"], p_data_fp32)

# more conservative since it's an approximated value
if N_sma >= 5:
    step_size = (
        group["lr"]
        * math.sqrt(
            (1 - beta2_t)
            * (N_sma - 4)
            / (N_sma_max - 4)
            * (N_sma - 2)
            / N_sma
            * N_sma_max
            / (N_sma_max - 2)
        )
        / (1 - beta1 ** state["step"])
    )
    denom = exp_avg_sq.sqrt().add_(group["eps"])
    p_data_fp32.addcdiv_(-step_size, exp_avg, denom)
else:
    step_size = group["lr"] / (1 - beta1 ** state["step"])
    p_data_fp32.add_(-step_size, exp_avg)

p.data.copy_(p_data_fp32)

return loss

class AdamW(Optimizer):

```



```
continue

grad = p.grad.data.float()
if grad.is_sparse:
    raise RuntimeError(
        "Adam does not support sparse gradients, please consider SparseAdam in
    )

p_data_fp32 = p.data.float()

state = self.state[p]

if len(state) == 0:
    state["step"] = 0
    state["exp_avg"] = torch.zeros_like(p_data_fp32)
    state["exp_avg_sq"] = torch.zeros_like(p_data_fp32)
else:
    state["exp_avg"] = state["exp_avg"].type_as(p_data_fp32)
    state["exp_avg_sq"] = state["exp_avg_sq"].type_as(p_data_fp32)

exp_avg, exp_avg_sq = state["exp_avg"], state["exp_avg_sq"]
beta1, beta2 = group["betas"]

state["step"] += 1

exp_avg_sq.mul_(beta2).addcmul_(1 - beta2, grad, grad)
exp_avg.mul_(beta1).add_(1 - beta1, grad)

denom = exp_avg_sq.sqrt().add_(group["eps"])
bias_correction1 = 1 - beta1 ** state["step"]
```

```

bias_correction2 = 1 - beta2 ** state["step"]

if group["warmup"] > state["step"]:
    scheduled_lr = 1e-8 + state["step"] * group["lr"] / group["warmup"]
else:
    scheduled_lr = group["lr"]

step_size = (
    scheduled_lr * math.sqrt(bias_correction2) / bias_correction1
)

if group["weight_decay"] != 0:
    p_data_fp32.add_(-group["weight_decay"] * scheduled_lr, p_data_fp32)

p_data_fp32.addcdiv_(-step_size, exp_avg, denom)

p.data.copy_(p_data_fp32)

return loss

from .warmup import GradualWarmupScheduler
from torch.optim.lr_scheduler import _LRScheduler
from torch.optim.lr_scheduler import ReduceLROnPlateau

```

```

class GradualWarmupScheduler(_LRScheduler):
    """ Gradually warm-up(increasing) learning rate in optimizer.
    Proposed in 'Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour'
    """

```

Args:

optimizer (Optimizer): Wrapped optimizer.

```

multiplier: target learning rate = base lr * multiplier if multiplier
total_epoch: target learning rate is reached at total_epoch, gradually
after_scheduler: after target_epoch, use this scheduler(eg. ReduceLROn
"""
def __init__(self, optimizer, multiplier, total_epoch, after_scheduler):
    self.multiplier = multiplier
    if self.multiplier < 1.:
        raise ValueError('multiplier should be greater than or equal to 1.')
    self.total_epoch = total_epoch
    self.after_scheduler = after_scheduler
    self.finished = False
    super(GradualWarmupScheduler, self).__init__(optimizer)

    def get_lr(self):
        if self.last_epoch > self.total_epoch:
            if self.after_scheduler:
                if not self.finished:
                    self.after_scheduler.base_lrs = [base_lr * self.multiplier for base_lr
                                                     in self.base_lrs]
                    self.finished = True
            return self.after_scheduler.get_last_lr()
        return [base_lr * self.multiplier for base_lr in self.base_lrs]

        if self.multiplier == 1.0:
            return [base_lr * (float(self.last_epoch) / self.total_epoch) for base_lr
                                                     in self.base_lrs]
        else:
            return [base_lr * ((self.multiplier - 1.) * self.last_epoch / self.total_epoch +
                               self.multiplier) for base_lr in self.base_lrs]

```

```
def step_ReduceLROnPlateau(self, metrics, epoch=None):  
    if epoch is None:  
        epoch = self.last_epoch + 1  
    self.last_epoch = epoch if epoch != 0 else 1 # ReduceLROnPlateau is called at epoch 0  
    if self.last_epoch <= self.total_epoch:  
        warmup_lr = [base_lr * ((self.multiplier - 1.) * self.last_epoch / self.total_epoch) ** self.warmup_exponent for base_lr in self.base_lrs]  
        for param_group, lr in zip(self.optimizer.param_groups, warmup_lr):  
            param_group['lr'] = lr  
    else:  
        if epoch is None:  
            self.after_scheduler.step(metrics, None)  
        else:  
            self.after_scheduler.step(metrics, epoch - self.total_epoch)  
  
    def step(self, epoch=None, metrics=None):  
        if type(self.after_scheduler) != ReduceLROnPlateau:  
            if self.finished and self.after_scheduler:  
                if epoch is None:  
                    self.after_scheduler.step(None)  
                else:  
                    self.after_scheduler.step(epoch - self.total_epoch)  
                    self._last_lr = self.after_scheduler.get_last_lr()  
                else:  
                    return super(GradualWarmupScheduler, self).step(epoch)  
            else:  
                self.step_ReduceLROnPlateau(metrics, epoch)
```