

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»  
Факультет прикладної математики  
Кафедра прикладної математики

Пояснювальна записка до курсового проекту  
із дисципліни  
«Алгоритми і системи комп'ютерної математики»  
на тему  
«Автоматична анотація зображень за допомогою нейронних мереж»

Виконав:  
студент групи КМ-01  
Скорденко Д. О.

Керівник:  
асистент кафедри ПМА  
Ковальчик-Химюк Л. О.

## АНОТАЦІЯ

В даній роботі описано мультимодальну систему маркування зображень, в якій зроблено акцент на трьох аспектах: висока точність, використання тексту в якості додаткової інформації, явна підсистема для передбачення к-сті лейблів. Всі ці рішення значно підвищують точність в порівнянні із існуючими рішеннями.

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

- \* DNN – Глибинна нейронна мережа (Deep Neural Network)
- \* CNN – Згорткова нейронна мережа (Convolutional Neural Network)
- \* RNN – Рекурсивна нейронна мережа (Recursive Neural Network)
- \* Анотація зображень, маркування зображень, мульти-лейбел класифікація – взаємозамінні поняття
- \* Тег (Tag) – шумна інформація надана користувачем у формі тексту (наприклад для зображення кота "Cat, Canada, Cola")
- \* Лейбл (Label) – синонім слова ground truth для класифікації

## ЗМІСТ

Перелік умовних позначень, скорочень і термінів .....	3
1 Вступ .....	5
2 Огляд існуючих рішень .....	6
3 Моделювання .....	8
3.1 VCNN .....	8
3.2 MLP .....	9
3.3 LP .....	9
3.4 LQP .....	10
3.5 Процес тренування .....	10
3.6 Процес тестування .....	13
4 Експерименти .....	14
4.1 Датасет .....	14
4.2 Метрики .....	16
4.3 Тренування .....	18
5 Аналіз результатів .....	20
5.1 Аналіз компонентів системи .....	20
5.2 Порівняння з існуючими рішеннями .....	21
5.3 Демонстративні приклади .....	23
Висновки .....	25
Перелік посилань .....	26
Додаток А Код лістинг .....	28

## 1 ВСТУП

Задача класифікації – це одна із основних задач в аналізі зображень, вона полягає у присвоєнні кожному зображенню один із класів. Таким чином дане формулювання накладає обмеження – зображення містить тільки один об'єкт. Поява DNN [3] та її подальший розвитком у CNN [13, 12] разом із створенням великих датасетів як-от ImageNet [4] дало змогу вирішувати задачу класифікації зображень значно швидше і якісніше ніж люди.

Зрозуміло, що зображення – це той тип даних, який у абсолютній більшості випадків містить більше одного об'єкта. Для поглиблення опису існує задача маркування зображень (image labeling). На відміну від класифікації, вона полягає у маркуванні зображення більше ніж одним класом. Таким чином якість опису зображення кратно зростає у порівнянні із звичайною класифікацією, однак привносить декілька складних завдань.

По-перше, наявність декількох класів у одного зображення створює можливість описувати значно ширший спектр візуальної інформації: різні об'єкти, стилі, дії, і тд. Поява великих хостингів зображень таких як Imgur, Flickr, та ін., де користувачі можуть як завантажувати різноманітні зображення, так і додавати до них описову інформацію у вигляді тегів / анотацій, дала змогу створити досить різноманітні датасети: ImageNet [4], MS-COCO [10], NUS-WIDE [2], та ін.

По-друге, анотація зображень передбачає не лише маркування більше ніж одним класом, а і передбачення к-сті класів. Для опису зображенням із широким спектром понять необхідно  $N$  класів, для зображення із простим вмістом – 2-3 класи.

По-третє, анотація зображень потребує оцінки якості проведеного маркування. Оскільки будь який датасет буде містити в собі дисбаланс класів в тій чи іншій мірі, важливо оцінювати маркування із урахуванням цього.

Все це робить задачу маркування зображення досить складною.

## 2 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

### Базове рішення

Базовим рішенням для більшості робіт із маркування зображення є використання CNN. Більшість робіт використовує різні архітектури ResNet [6], AlexNet [1], GoogleNet [15]. Спільним між ними є те, що вони вже натреновані на великому датасеті, здебільшого ImageNet [4]. Для адаптації моделі до обраного контексту така модель дотреновується (fine tune), замінюючи базовий класифікатор на такий же простий із адаптованою  $k$ -стю вихідних класів [5], або ж на більш складний класифікатор (який надає більш точні результати) [18]. Це працює завдяки тому, що всі архітектури сучасних CNN моделей є багат шаровими, і в них перші шари розпізнають базові особливості (features) зображення, які можна навіть візуалізувати, однак останні шари вивчають більш глибокі особливості зображення, таким чином роблячи модель більш універсальною при зміні класифікатора.

### Додаткова інформація

Більш нові роботи також розглядають додавання сторонньої інформації для класифікації зображень. Існує два основних підходів:

а) Семантичний аналіз лейблів. Даний підхід аналізує зв'язок між різними класами. Схожі за контекстом лейбли знаходяться поруч (наприклад: риба, вода) [7, 9]

б) Аналіз додаткової інформації. Даний підхід аналізує додаткову до зображення інформацію. Це може бути як текстова інформація (теги / анотації) [20], так і метадані зображення [8, 16]

### $K$ -сть лейблів

Всі наведені вище роботи розглядають задачу вибору  $k$ -сті лейблів як найкращі  $k$  (top  $k$ ) маркувань.  $k$  найбільше ймовірних класів, де  $k$  – наперед задана константа. Очевидно, що такий вибір  $k$ -сті класів не є оптимальним,

так як більш змістовні зображення будуть містити менше описової інформації і навпаки – менш змістовні будуть містити лишню інформацію, яка до того ж може не мати нічого спільного із цим зображенням (Рис.2.1)

Image				
Truth	fire	grass	person	grass person water
Top 5 pred	red night asia island child	red green trees tree interestingness	people photography asia china adults	water sea ocean pink rocks
Model pred	fire	grass house plants sky	person	person water

Рис. 2.1 – Приклад адаптивної  $k$ -сті лейблів. 'Truth' – правдиве маркування, 'Top 5 pred' – ілюстрація вибору top  $k$ , при  $k = 5$ , 'Model pred' – приклад роботи нашої моделі.

Один із сучасних підходів як-от CNN-RNN [17], розглядає задачу маркування як задачу перекладу зображення в текст (image to text), де CNN – це кодувальник (encoder), а RNN (decoder) автоматично виконує як задачу маркування, так і задачу динамічного вибору кількості лейблів, однак є певні обмеження накладені на порядок класів.

## 3 МОДЕЛЮВАННЯ

На основі проведеного аналізу альтернатив, дана робота пропонує розглянути мультимодальну систему (обробляє зображення і текст), яка складається із чотирьох компонентів (Рис. 3.1)

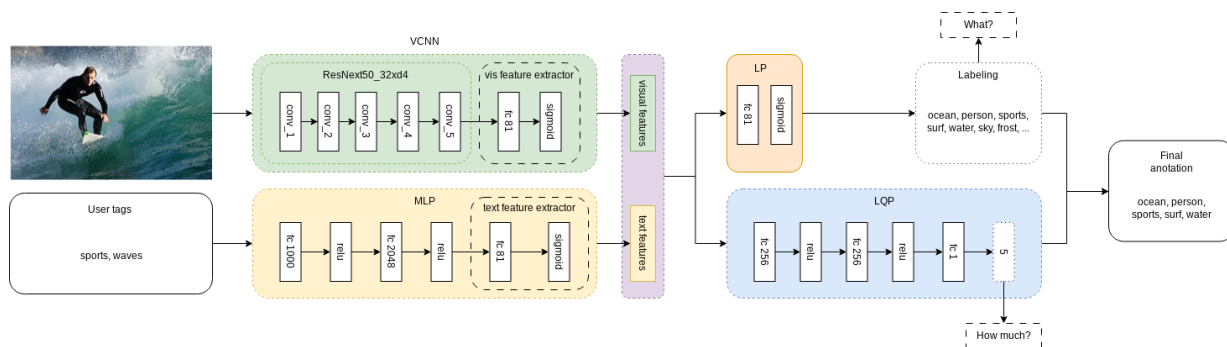


Рис. 3.1 – Архітектура композитної системи

### 3.1 VCNN

Модель VCNN (Рис. 3.1) призначена для вивчення особливостей (features) із зображення. Отримує на вхід піселі зображення  $I$ , у формі матриці розмірності  $(B, C, W, H)$ , де  $B$  – к-сть зображень у групі для тегування,  $C$  – к-сть каналів у зображеннях зазвичай 1 або 3, Grey або RGB відповідно,  $W, H$  – розмірність зображень.

За базове рішення використовується ResNext101\_32x8d [19] (сучасна версія resnet), із адаптованим класифікатором, натреновану на датасеті ImageNet [4].

На виході даної моделі ми отримуємо вектор вірогідностей  $vf$  (visual feature vector), який вказує вірогідність маркування зображення класом  $j$



на основі візуальної інформації.

### 3.2 MLP

MLP (Рис. 3.1) – аналізує текстові особливості (text features) тегів до зображення. Теги до зображення  $i$  репрезентуються як бінарний вектор  $I = [1, 0, 1, 0, \dots, N]$ , де 1 – це наявність тегу, а  $N$  – к-сть тегів.

Головна причина вибору звичайної MLP моделі для аналізу текстової інформації – це те, що вхідна інформація – це шумні теги (наприклад: для фото кота – теги "Канада", "Кіт").

На виході даної моделі ми отримуємо вектор  $tf$  вірогідностей (text feature vector), який вказує вірогідність маркування зображення класом  $j$  на основі текстової інформації.

### 3.3 LP

LP (Рис. 3.1) – аналізує вектор вірогідності  $v$ , який є композицією векторів  $vf$  та  $tf$ :  $v = [vf, tf]$ .

На виході даної моделі ми отримуємо вектор вірогідностей, який комбінує інформацію отриману як із візуальної так і з текстової інформації.

### 3.4 LQP

Модель LQP (Рис. 3.1) аналізує кількість лейблів на основі вектору вірогідностей  $v$ , який є композицією векторів  $vf$  та  $tf$ :  $v = [vf, tf]$ .

Існує два підходи до визначення к-сті за допомогою нейронних мереж: класифікація та регресія. LQP – регресійна модель.

Оскільки регресійні моделі досить швидко перенавчаються (overfitting), то необхідно задіяти регуляризацію. В даній роботі, в якості регуляризатора задіяні Dropout шари [14], із вірогідністю відкидання (dropout rate) 0.5.

На виході даної моделі є число, яке вказує на кількість лейблів у зображенні.

### 3.5 Процес тренування

Система є мультимодальною, і містить досить багато параметрів, тому тренувати її за один раз буде складно і буде виникати проблема затухаючого градієнта.

Тому тренування відбувається у декілька стадій, у якому кожна із моделей тренується окремо (деякі з них можна тренувати синхронно).

#### Цільові функції

Для початку варто розглянути цільові функції (функція втрат, loss function). Дані функції є базовим компонентом глибокого навчання.

В даній роботі використовуються дві функції: BCEWithLogitsLoss та MSELoss.

### *BCEWithLogitsLoss*

Для тренування класифікаційних моделей (VCNN, MLP, LP) вихідні логіти  $z_{ij}$  для групи (batch) зображень  $I_N$  при  $i = 1...N$ ,  $j = 1...C$ , де  $N$  - кількість зображень в групі,  $C$  - кількість цільових класів, цільова функція має вигляд:

$$\mathcal{L}_{cls} = \frac{1}{NC} \sum_i^N \sum_j^C y_{ij} \cdot \ln(\sigma(z_{ij})) + (1 - y_{ij}) \cdot \ln(1 - \sigma(z_{ij})) \quad (3.1)$$

, де  $y_{ij} = 1$  якщо зображення  $i$  анотоване класом  $j$ , інакше -  $y_{ij} = 0$ , а  $\sigma(\cdot)$  - це сигмоїдальна активаційна функція

### *MSELoss*

Для тренування регресійної моделі LQP вихідні логіти  $z_i$  для групи (batch) зображень  $I_N$  при  $i = 1...N$ , де  $N$  - кількість зображень в групі, цільова функція має вигляд:

$$\mathcal{L}_{reg} = \frac{1}{N} \sum_i^N (y_i - z_i)^2 \quad (3.2)$$

, де  $y_i$  - це кількість лейблів для зображення  $I_i$ .

### **Тренування VCNN**

Тренування моделі ResNext [19] з нуля є досить складною задачею (дана модель має  $\approx 80M$  параметрів), адже для цього потрібні значні обчислювальні потужності.

Саме тому виристовується натренована модель із адаптованим класифікатором (visual feature extractor) (Рис. 3.1), яка підганяється

(finetuned) на обраному датасеті.

Існує два підходи для підгонки:

1) Підгонка всієї моделі: всі шари моделі підганяються (дотреновуються) з низькою швидкістю навчання (learning rate). Даний підхід вимагає великої обчислювальної потужності, однак надає високу точність та досить таки швидко тренується (в порівнянні із тренуванням з нуля).

2) Підгонка класифікатора: відбувається тренування тільки класифікатора, фіксуючи всі інші параметри моделі. Даний підхід значно пришвидшує тренування в обмін на певну деградацію точності в порівнянні із 1-им варіантом.

В даній роботі використовується 2 варіант підгонки.

### **Тренування MLP**

Дана модель є звичайним багат шаровим персептроном, тому її тренування досить таки прямолінійне.

### **Тренування LP**

Дана модель призначена для обрахування вірогідностей на основі вектору  $f = [vf, tf]$ , оскільки вона складається із одного шару то її тренування також очевидне.

### **Тренування LQP**

Дана модель є регресійною, її тренування також є очевидним, однак варто нормалізувати вхідну к-сть лейблів, так як це пришвидшить та/або покращить збіжність моделі.

### 3.6 Процес тестування

#### Тестування класифікаційних моделей (VCNN, MLP, LP)

Кожна із даних моделей обраховує вектор ймовірностей  $P$ , для тестування необхідно перевести вектор ймовірностей (наприклад:  $P = [0.9, 0.6, 0.1, 0.4, 0.6]$ ) у вигляд маркування (наприклад:  $M = [1, 1, 0, 0, 1]$ ).

Розглянемо два основних підходи:

1) Порогове значення (threshold): для вектору ймовірностей  $P$  та порогу  $T$  – вектор маркувань обраховується наступним чином: якщо  $y_i > T$ , то маркуємо 1, інакше 0. Наприклад при  $T = 0.5$ :  $[0.9, 0.6, 0.1, 0.4, 0.6] \rightarrow [1, 1, 0, 0, 1]$ .

2) Найкращі  $k$  (top  $k$ ): для вектору ймовірностей  $P$  та числа  $k$  – вектор маркувань обраховується наступним чином: маркуємо 1 найкращі  $k$  ймовірностей, інакше 0. Наприклад при  $k = 4$ :  $[0.9, 0.6, 0.1, 0.4, 0.6] \rightarrow [1, 1, 0, 1, 1]$ .

Оскільки дані моделі **не передбачають** передбачення кількості лейблів, то для тренування даних моделей використовується метод top  $k$ , при чому  $k$  обирається на основі правдивого маркування.

Тобто для зображення  $I$ , із маркуванням  $Y = [1, 0, 0, 1, 1]$ , і вектором ймовірностей  $P = [0.8, 0.9, 0.1, 0.3, 0.2]$  та результуючим вектором маркувань  $M$ :  $k = 3$ , так як  $Y$  містить 3 промаркованих класи, перетворення  $P \rightarrow M \equiv [0.8, 0.9, 0.1, 0.3, 0.2] \rightarrow [1, 1, 0, 1, 0]$

#### Тестування композитної моделі

Для тестування композитної моделі (Рис. 3.1) необхідно обрахувати результуючі значення для моделей LP та LQP, і обрати top  $k$  лейблів LP, де  $k$  – це передбачення LQP.

## 4 ЕКСПЕРИМЕНТИ

## 4.1 Датасет

Один із найбільш часто використовуваних датасетів для тестування моделей анотації зображень – NUS-WIDE [2], він складається із 269,655 зображень, 81 лейблу, та  $\approx 5000$  тегів в якості сторонньої текстової інформації. Для проведення тренування/тестування використовується розподіл, надведений разом із датасетом, так як він є збалансований настільки, наскільки це можливо (Рис. 4.1).

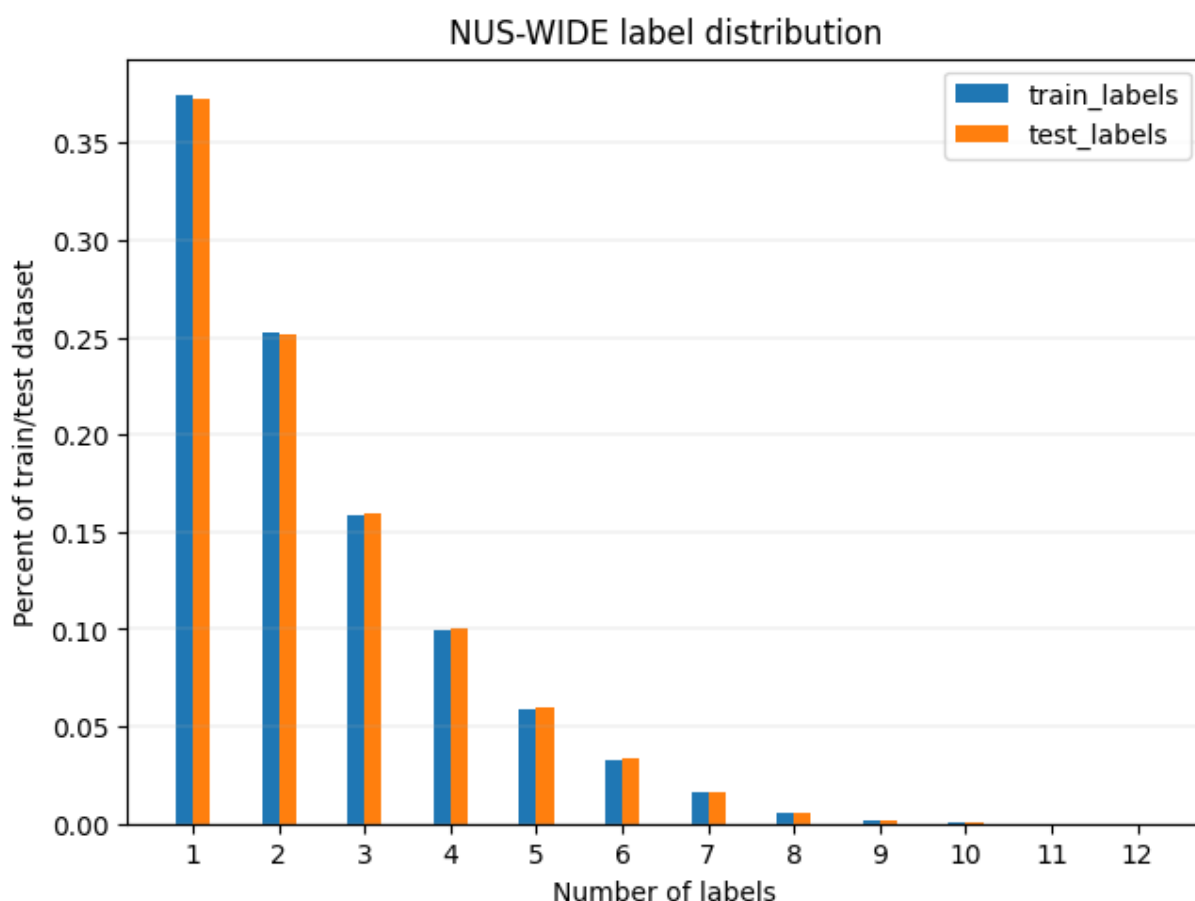


Рис. 4.1 – Розподіл лейблів у тренувальному/тестовому датасеті

Важливо відмітити, що даний датасет містить посилання на зображення на ресурсі Flickr, і деякої частина цих зображень вже там немає. Також

буде використано тільки 1000 найбільш частих тегів з  $\approx 5000$ , при чому зображення, які не містять жодного тега відфільтровано.

Далі наведено деякі числові характеристики тренувальної/тестової вибірок:

	Тренування	Тестування
Кількість зображень	121962	81636
Середня к-сть лейблів	2.42	2.43
Медіана к-сть лейблів	2	2
Мінімальна к-сть лейблів	1	1
Максимальна к-сть лейблів	12	13

Табл. 4.1 – Характеристики тренувальної/тестової вибірок

## 4.2 Метрики

Для оцінки точності будуть використовуватись метрики, які є загально вживаними для оцінки задачі анотації зображень (multi-label image annotation).

$$\begin{aligned}
 P_C &= \frac{1}{C} \sum_{j=1}^C \frac{NI_j^c}{NI_j^p} & P_I &= \frac{\sum_{i=1}^N NL_i^c}{\sum_{i=1}^N NL_i^p} \\
 R_C &= \frac{1}{C} \sum_{j=1}^C \frac{NI_j^c}{NI_j^g} & R_I &= \frac{\sum_{i=1}^N NL_i^c}{\sum_{i=1}^N NL_i^g} \\
 F1_C &= \frac{2 \cdot P_C \cdot R_C}{P_C + R_C} & F1_I &= \frac{2 \cdot P_I \cdot R_I}{P_I + R_I}
 \end{aligned} \tag{4.1}$$

,де

- \*  $C$  – к-сть класів
- \*  $N$  – к-сть тестових зображень
- \*  $NI_j^c$  – к-сть зображень які **коректно** промарковано як клас  $j$
- \*  $NI_j^g$  – к-сть зображень які мають клас  $j$
- \*  $NI_j^p$  – к-сть зображень які промарковано як клас  $j$
- \*  $NL_i^c$  – к-сть **коректно** промаркованих лейблів для зображення  $i$
- \*  $NL_i^g$  – к-сть лейблів які має зображення  $i$
- \*  $NL_i^p$  – к-сть промаркованих лейблів для зображення  $i$

Варто відзначити, що дані метрики є зміщеними (biased), при чому по-класові метрики ( $C$ ) зміщені в сторону рідкісних класів, а загальні метрики ( $I$ ) – в сторону частих класів.



Для того що отримати уніфіковане представлення про ефективність моделі буде використовуватись наступна метрика, яка бере до уваги як  $F1_C$  так і  $F1_I$ , що полегшує інтерпретацію результатів:

$$F1_H = \frac{2 \cdot F1_C \cdot F1_I}{F1_C + F1_I} \quad (4.2)$$

### 4.3 Тренування

Для програмної реалізації запропонованої моделі було використано PyTorch.

Оскільки в даній роботі, використовується модель ResNext [19] натренована на датасеті ImageNet [4], то для вхідних зображень потрібно застосувати певне перетворення:

1) Зміна розміру (Resize)  $232 \times 232$ , використовуючи білінійну інтерполяцію

2) Центральний кроп (Central crop)  $224 \times 224$

3) Зміна масштабу (Rescale)  $[0,1]$

4) Нормалізація на основі статистичних величин ImageNet [4]. А саме:  $\text{mean} = [0.485, 0.456, 0.406]$  та  $\text{std} = [0.229, 0.224, 0.225]$

Дане перетворення доступно у бібліотеці PyTorch.

#### Параметри навчання

Класифікаційні моделі VCNN та MLP навчалися із швидкістю навчання (learning rate) 0.001, а LP – зі швидкістю 0.01. Також для навчання цих трьох моделей використовувався контролер швидкості навчання (learning rate scheduler), який множив швидкість навчання на 0.5, досягаючи 5-ої та 10-ої епохи.

Регресійна модель LQP навчалась із сталою швидкістю навчання (learning rate) 0.0005.

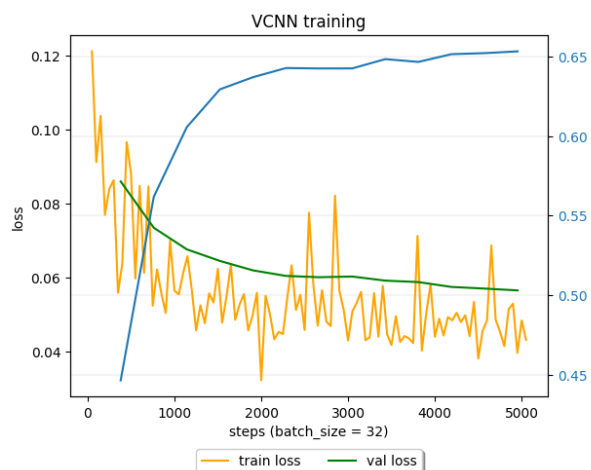
Для всіх навчання всіх вище згаданих моделей використовувався оптимізатор AdamW, із параметром l1 регуляризації (weight decay) 0.0003.

Розмір групи (batch size) 32.

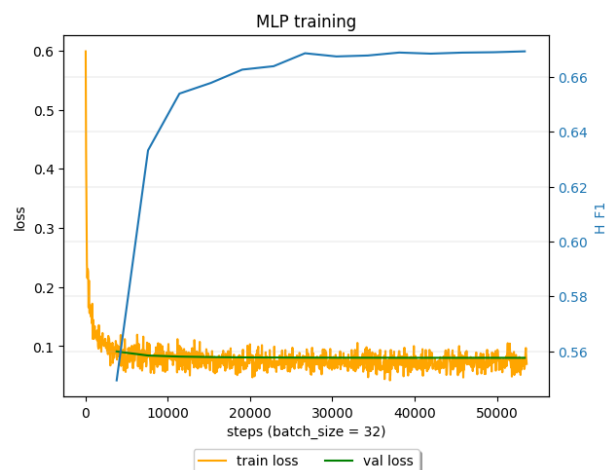
Також варто відзначити, що в даній роботі епоха – це 10% від усіх даних, при чому після кожної епохи дані перемішуються (shuffle), отримавши нові 10% даних.

## Процес навчання

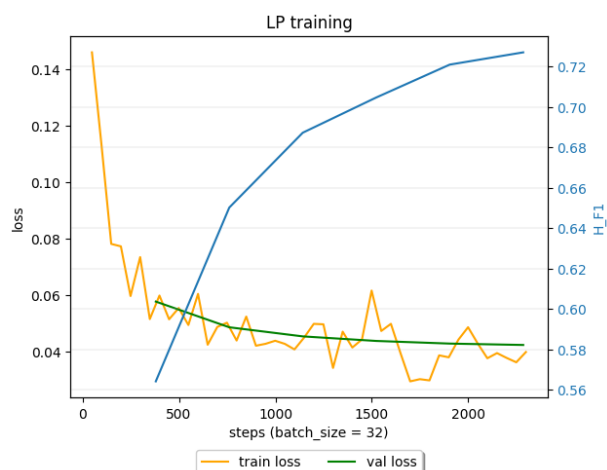
Оскільки дана робота розглядає базове тренування всіх компонентів моделі, то варто розглянути саме процес тренування, для того щоб вказати місця, які потенційно можна покращити для збільшення точності моделі.



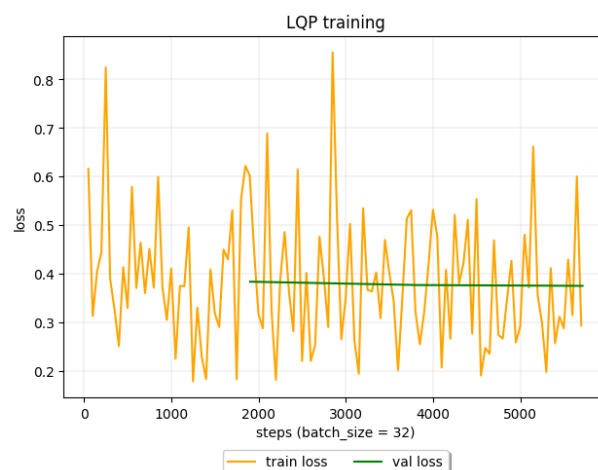
(4.2a) Навчання VCNN



(4.2б) Навчання MLP



(4.2в) Навчання LP



(4.2г) Навчання LQP

Рис. 4.2 – Процес навчання моделей

## 5 АНАЛІЗ РЕЗУЛЬТАТІВ

## 5.1 Аналіз компонентів системи

Модель	Динамічне $k$	Мультимодальність	$P_C$	$R_C$	$F1_C$	$P_I$	$R_I$	$F1_I$	$F1_H$
Композитна	Так	Так	0.71	0.53	0.61	0.74	0.72	0.73	0.66
VCNN+MLP+LP	Hi	Так	0.78	0.60	0.68	0.78	0.78	0.78	0.73
VCNN	Hi	Hi	0.65	0.54	0.59	0.73	0.73	0.73	0.65

Табл. 5.1 – Порівняння компонентів моделі

Результуюча композитна модель показала гірший результат ніж її складові, все це тому, що дані моделі не вміють обирати кількість результуючих класів, а отже для їх оцінки використовується  $k$ -сть класів із тестового датасету (3.6). На реальних даних ці моделі покажуть значно менший результат.

Варто зазначити, що незважаючи на те, що моделі були натреновані не дуже якісно, потенційно точність моделі можна збільшити мінімум до  $F1_H = 0.73$  (при кращому тренуванні LQP). На додачу до цього можна провести більш якісне тренування інших моделей, і підвищити точність.

Додавання модальності у формі тексту дало можливість підвищити точність моделі на  $\approx 12\%$ .

## 5.2 Порівняння з існуючими рішеннями

Модель	Динамічне $k$	Мультимодальність	$P_C$	$R_C$	$F1_C$	$P_I$	$R_I$	$F1_I$	$F1_H$
SR-CNN-RNN [11]	Так	Так	0.72	0.62	0.66	0.77	0.77	0.77	0.71
<b>Композитна</b>	<b>Так</b>	<b>Так</b>	<b>0.71</b>	<b>0.53</b>	<b>0.61</b>	<b>0.74</b>	<b>0.72</b>	<b>0.73</b>	<b>0.66</b>
SRN [21]	Ні	Ні	0.65	0.56	0.58	0.75	0.71	0.73	0.64
SINN [7]	Так	Так	0.58	0.6	0.59	0.57	0.79	0.66	0.63
TagNeighbour [8]	Так	Ні	0.55	0.57	0.56	0.53	0.75	0.62	0.59
CNN-RNN [17]	Ні	Ні	0.4	0.3	0.35	0.5	0.62	0.55	0.43
CNN+WARP [5]	Ні	Ні	0.32	0.36	0.34	0.49	0.6	0.54	0.41
CNN+Softmax [5]	Ні	Ні	0.32	0.31	0.31	0.47	0.59	0.53	0.39

Табл. 5.2 – Порівняння результуючих метрик для різних моделей на датасеті NUS-WIDE

З наведеної вище таблиці можна зробити наступні висновки:

**Точність запропонованого рішення** Композитна модель (VCNN+MLP+LP+LQP) продемонструвала високу точність на тестових метрика у порівнянні із розглянутими альтернативними рішеннями. Згідно із метрикою  $F1_H$  запропоноване рішення є другим по точності, проте розрив між 3 та 4 місцем незначний.








**Вплив додаткової модальності** Задача маркування зображень розглядає зображення як основну модальність, однак додавання модальності, очікувано, покращує результати маркування. Це підтверджують метрики наведені в Табл. 5.2. Варто зазначити, що існують і інші методи додавання

модальності, наприклад: геолокація зображення [16], метадані [8], використання текстових embedding [9].

**Вплив динамічного передбачення кількості лейблів** Використання системи, яка передбачає роботу із динамічною кількістю лейблів при маркуванні значено підвищує точність. Згідно із наведеними метриками покращення складає  $\approx 37\%$  (Табл. 5.2).









### 5.3 Демонстративні приклади

З тестового датасету випадковим чином обрано декілька зображень, для демонстрації роботи моделі:

Image	Truth	Model pred	Image	Truth	Model pred
	nighttime sky water window	sky water window		clouds grass house sky	clouds grass sky
	window	buildings window		leaf plants sky	leaf plants
	clouds road sky	clouds sky		animal grass	animal grass horses
	animal	animal		person	person

(5.1a)

(5.1б)

Image	Truth	Model pred	Image	Truth	Model pred
	water	person water		person sky	person sky
	flowers plants sky	flowers plants		animal flags	animal sky
	animal	animal		sky temple	temple
	grass tree	plants tree		person	person

(5.1в)

(5.1г)

Рис. 5.1 – Демонстративні приклади

Задамо умовне позначення "1::1", що означає демонстративний приклад "а", перше зображення зверху, "3::2" – приклад "в", друге зображення зверху і тд.

Демонстративні приклади вказують, що в деяких випадках модель вказує маркування, якого не було в датасеті, однак яке присутнє на зображенні,

наприклад: [1::2, 2::3, 3::1].

В інших випадках: [1::3, 2::1, 2::2, 3::2, 4::3] - відсікає або неіснуючі класи, або ті, які мало присутні на зображенні.

Звичайно, іноді модель помиляється, вказуючи неіснуючі класи, або навпаки не маркуючи існуючі класи [1::1, 4::2, 4::3].



## ВИСНОВКИ

В даній роботі було запропоновано мультимодальну композитну модель, вказано на деталі тренування, проведено оцінювання її точності на тестових метриках.

Було проведено порівняння запропонованого рішення із альтернативними моделями.

В цілому запропоноване рішення досить непогано вирішує задачу маркування зображення, навіть незважаючи на те, що в даній роботі проведено базове тренування моделі.

Також важливо відмітити обраний даатсет NUS-WIDE [2], в більш нових рішеннях даної задачі використовується датасет MSCOCO [10], оскільки він більш якісний і містить менше шумів.

Результуюча композитна модель збережена у форматі safetensors.

В подальшому варто розглянути більш точне тренування моделі, та обрати датасет MSCOCO. Також можна розглянути можливість тренування моделі цільно, а не по частинах окремо.

## ПЕРЕЛІК ПОСИЛАНЬ

- [1] Krizhevsky Alex, Sutskever Ilya та Hinton Geoffrey. „ImageNet Classification with Deep Convolutional Neural Networks“. B: *Advances in Neural Information Processing Systems*. За ред. F. Pereira та ін. Т. 25. Curran Associates, Inc., 2012. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf).
- [2] Tat-Seng Chua та ін. „NUS-WIDE: A Real-World Web Image Database from National University of Singapore“. B: *Proc. of ACM Conf. on Image and Video Retrieval (CIVR'09)*. Santorini, Greece., July 8-10, 2009.
- [3] Dan Cireşan, Ueli Meier та Juergen Schmidhuber. „Multi-column Deep Neural Networks for Image Classification“. B: (2012). arXiv: 1202.2745.
- [4] Li Deng та ін. „Imagenet: A large-scale hierarchical image database“. B: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, C. 248—255.
- [5] Yunchao Gong та ін. *Deep Convolutional Ranking for Multilabel Image Annotation*. 2013. eprint: arXiv:1312.4894.
- [6] Kaiming He та ін. *Deep Residual Learning for Image Recognition*. 2015. eprint: arXiv:1512.03385.
- [7] Hexiang Hu та ін. *Learning Structured Inference Neural Networks with Label Relations*. CVPR 2016. 2016.

- [8] Justin Johnson, Lamberto Ballan та Li Fei-Fei. *Love Thy Neighbors: Image Annotation by Exploiting Image Metadata*. ICCV 2015. 2015.
- [9] Qing Li та ін. *Learning Category Correlations for Multi-label Image Recognition with Graph Networks*. 2019. eprint: arXiv:1909.13005.
- [10] Tsung-Yi Lin та ін. „Microsoft COCO: Common Objects in Context“. B: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. URL: <http://arxiv.org/abs/1405.0312>.
- [11] Feng Liu та ін. „Semantic Regularisation for Recurrent Image Annotation“. B: (листоп. 2016). arXiv: 1611.05490 [cs.CV].
- [12] Keiron O'Shea та Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. eprint: arXiv:1511.08458.
- [13] Karen Simonyan та Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. eprint: arXiv:1409.1556.
- [14] Nitish Srivastava та ін. „Dropout: A Simple Way to Prevent Neural Networks from Overfitting“. B: *Journal of Machine Learning Research* 15.56 (2014), С. 1929—1958. URL: <http://jmlr.org/papers/v15/srivastava14a.html>.
- [15] Christian Szegedy та ін. *Going Deeper with Convolutions*. 2014. eprint: arXiv:1409.4842.
- [16] Kevin Tang та ін. *Improving Image Classification with Location Context*. 2015. eprint: arXiv:1505.03873.

- [17] Wei Wang та ін. „CNN-RNN: A Unified Framework for Multi-Label Image Classification“. В: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Черв. 2016.
- [18] Yunchao Wei та ін. „CNN: Single-label to Multi-label“. В: (2014). DOI: 10.1109/TPAMI.2015.2491929. eprint: arXiv:1406.5726.
- [19] Saining Xie та ін. „Aggregated residual transformations for deep neural networks“. В: (листоп. 2016). arXiv: 1611.05431 [cs.CV].
- [20] Fengtao Zhou, Sheng Huang та Yun Xing. *Deep Semantic Dictionary Learning for Multi-label Image Classification*. AAAI 2021. 2021.
- [21] Feng Zhu та ін. „Learning spatial regularization with image-level supervisions for multi-label image classification“. В: (лют. 2017). arXiv: 1702.05891 [cs.CV].

## Додаток А

### Код лістинг

Код міститься у github репозиторії [Посилання](#)

Далі наведено загальний опис елементів проекту:

#### **Дані:**

Файли в директорії `scripts` `'nuswide2ndjson.py'` та `'1ktags.py'` призначенні для обробки сирих даних із датасету в формат `ndjson` для подальшого тренування.

Файл `data.py` містить адаптери та визначення датасету для тренування.

#### **Тренування:**

Скрипти для тренування для всіх моделей (VCNN, MLP, LP, LQP) знаходяться в директорії `'scripts/train'`.

#### **Тестування:**

Скрипти для тестування моделей (VCNN та Compose) знаходяться в директорії `'scripts/test'`.

#### **Інше:**

Скрипт `'scripts/compose2safe.py'` призначений для конвертації вагів моделей (VCNN, MLP, LP, LQP) формату `.skpt` у композитну модель формату `.safetensors`.