МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

# Розпізнавання образів

## Комп'ютерний практикум

**Навчальний посібник**

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за освітньою програмою «Системи і методи штучного інтелекту»
спеціальності 122 Комп'ютерні науки

Укладачі: Шаповал Н.В.

Електронне мережне навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2022

Рецензент Зайченко О.Ю., д.т.н., професор, кафедра математичних методів системного аналізу ННІПСА

Відповідальний редактор Джигирей І.М., к.т.н., доцент кафедри штучного інтелекту ННІПСА

*Гриф надано Методичною радою КПІ ім. Ігоря Сікорського*
*(протокол № 4 від 19.01.2023 р.)*
*за поданням Вченої ради навчально-наукового інституту*
*прикладного системного аналізу*
*(протокол № 4 від 19.12.2022 р.)*

В навчальному посібнику викладено основний теоретичний матеріал до відповідних практичних завдань вибіркового ОК «Розпізнавання образів» щодо класичних задач даної дисципліни: кластеризація, сегментація, виділення меж на зображенні тощо. Містяться методичні вказівки щодо виконання практичних завдань, матеріал для підготовки до занять під час самостійної роботи. При складанні посібника автор ставив за мету викласти предмет просто та наочно.

Посібник призначений для студентів, які навчаються за спеціальністю 122 «Комп'ютерні науки», а також для всіх, хто використовує методи розпізнавання образів у своїй практичній діяльності.

Реєстр. № НП 22/23‑363. Обсяг 1,1 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Перемоги, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів
і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

# Зміст

# Передмова

В навчальному посібнику містяться вказівки до виконання лабораторних робіт з дисципліни “Розпізнавання образів”. Метою навчальної дисципліни є:

- систематизоване викладення основ теорії розпізнавання образів;

- ознайомлення з сучасними підходами до розпізнавання шаблонів в різноманітно представлених даних;

- формування у здобувачів знань про принципи роботи систем розпізнавання;

- вироблення практичних навичок пошук раціональних рішень при створенні систем розпізнавання.

В процесі навчання здобувачі мають набути знання:

- статистичних методів розпізнавання;
- методів попередньої обробки зображень;
- методів сегментації зображень на основі піксельної подібності та на основі виділення меж на зображенні;
- методів розпізнавання обличчя людини;
- методів на основі використання нейронних мереж;
- розпізнавання образів в аудіо даних та визначення характеристик пози людини.

З метою практичного набуття здобувачами вміння проектувати систему розпізнавання для конкретної задачі передбачається виконання шести лабораторних робіт. Основними цілями циклу лабораторних робіт є ознайомлення здобувачів з принципами роботи обробки зображень, побудови згорткових нейронних мереж під конкретну задачу та сформувати вміння та навички вибору відповідного методу сегментації кластеризації чи то обробки зображення у відповідності до поставленої задачі, а також аналізу і оцінки одержаних результатів.

Навчальний посібник призначений для бакалаврів спеціальностей 122 «Компютерні науки» та 124 “Системний аналіз”.

Кожна лабораторна робота складається з короткої інструкції щодо виконання роботи та опису змісту звіту щодо виконаної роботи. Також кожна лабораторна робота містить короткі теоретичні відомості щодо теми, яка розглядаються в роботі, переліку запитань для самоконтролю та посилань на джерела і відповідну документацію щодо функцій та бібліотек необхідних для виконання роботи. Набуті при виконанні лабораторної роботи навички знадобляться для виконання наступних робіт. Лабораторні роботи виконуються на мові програмування Python, рекомендоване середовище Google Colab. Вибір середовища обумовлений тим, що воно дозволяє використовувати GPU, і здобувач не залежить від власних апаратних можливостей.

# Вступ

Розпізнавання образів — це автоматичне розпізнавання закономірностей у даних. Задача може полягати як в ідентифікації шаблону загалом так і в категоризації виявлених шаблонів. Отже, розпізнавання образів застосовує різні методи розпізнавання образів залежно від задачі та форми даних.

Розпізнавання образів — це не *одна* техніка, а скоріше широка колекція часто слабо пов’язаних знань і прийомів. Можливість розпізнавання образів часто є необхідною умовою для інтелектуальних систем.

Вхідними даними для розпізнавання образів можуть бути слова або тексти, зображення або аудіофайли. Автоматичне та машинне розпізнавання, опис, класифікація та групування шаблонів є важливими проблемами в різних інженерних та наукових дисциплінах, включаючи біологію, психологію, медицину, маркетинг, комп’ютерний зір та штучний інтелект.

шаблоном може бути будь-який об’єкт, який цікавить, який потрібно розпізнати та ідентифікувати: досить важливо, щоб ми хотіли знати його назву (його ідентичність).

Тому закономірності включають повторювані тенденції в різних формах даних. Наприклад, візерунком може бути зображення відбитка пальця, рукописне скоропис, обличчя людини або мовний сигнал. Шаблон можна спостерігати фізично, наприклад, на зображеннях і відео, або математично, застосовуючи статистичні алгоритми.

Мета розпізнавання образів заснована на ідеї, що процес прийняття рішень людиною певною мірою пов’язаний з розпізнаванням шаблонів. Наприклад, наступний хід у шаховій грі базується на поточному шаблоні дошки, а купівля або продаж акцій визначається складною моделлю фінансової

інформації. Тому метою розпізнавання образів є прояснення цих складних механізмів процесів прийняття рішень та автоматизація цих функцій.

# Комп’ютерний практикум № 1. Алгоритми кластеризації

**Тема:** Алгоритми кластеризації

**Мета:** провести порівняльний аналіз різних алгоритмів кластеризації

Хід виконання роботи

1. Попередньо обробити вихідну вибірку даних, обрану згідно варіантом.
2. Вибрати метод кластеризації та виконати наступне:

   2.1 Виконати кластеризацію для різного числа кластерів від 2 до 5;

   2.2 Для кожного результату зробити візуалізацію;

   2.3 Для кожного числа кластерів виконати дослідження щодо параметрів метода;

   2.4 Для кожного експерименту з п.2.1 оцінити час роботи.
3. Повторити п.2 для для різних методів кластеризації. Перелік методів є в таблиці 1.1. Та звести в таблицю дані по часу роботи.
4. Для кожного експерименту порахувати метрики кластеризації. Використати вбудовані метрик в бібліотеці Scikit-learn: adjusted_rand_index, adjusted_mutual_info, v_measure_score, completeness_score, fowlkes_mallows_score. Звести результати в таблицю.
5. Для неієрархічних методів кластеризації оцінити якість кластеризації $Q^1$, $Q^2$,$Q^3$, зробити порівняння.
6. Зробити висновки по роботі та відповісти на наступні питання:
   - який з алгоритмів спрацював краще за інші? Чому?
   - яка з метрик кластеризації на вашу думку краще оцінює результат? Чому?

Таблиця 1.1

| **Метод** | **Параметри** | **Посилання на документацію** |
|---|---|---|

| **кластеризації** | | |
|---|---|---|
| K-Means | init | https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans |
| Fuzzy K-Means | - | https://scikit-fuzzy.readthedocs.io/en/latest/api/skfuzzy.html#skfuzzy.cmeans |
| EM | - | https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html?highlight=mixture#sklearn.mixture.GaussianMixture |
| Mean shift | - | https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift.html#sklearn.cluster.MeanShift |
| k-medoid | init | https://scikit-learn-extra.readthedocs.io/en/stable/generated/sklearn_extra.cluster.KMedoids.html |
| Affinity propagation | damping_range [0.5,0.7,0.9] | https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html#sklearn.cluster.AffinityPropagation |
| Agglomerative clustering | Linkage [ward,complete, average] | https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html#sklearn.cluster.AgglomerativeClustering |
| Spectral clustering | - | https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html#sklearn.cluster.SpectralClustering |

Звіт має містити:

- результати всіх експериментів зведених у відповідні таблиці;
- візуалізацію результатів роботи алгоритмів;
- висновки по роботі;
- текст програми.

Короткі теоретичні відомості

Вирішення задачі розбиття множини елементів без навчання називають кластерним аналізом. Кластерний аналіз не потребує апріорної інформації про дані та дозволяє розділити множину досліджуваних об'єктів на групи схожих об'єктів - кластери.

У загальному випадку завдання кластеризації можна сформулювати наступним чином. Дана навчальна вибірка Θ = {x1, ..., xN}, де xi характеризується набором з $K$ – ознак (вимірюваних характеристик об'єктів). Потрібно знайти таку функцію кластеризації f, яка кожній точці x є Θ ставила б в однозначну відповідність певний елемент - мітку y є Y з множини міток Y = {y1, ..., ym} (кожна мітка yi відповідає деякому кластеру Yi). У задачі кластеризації множина міток Y невідома. Якщо множина Y відома, то задача кластеризації вироджується в задачу класифікації. Апріорна інформація про класифікацію об'єктів при цьому відсутня.

Основні параметри кластеризації:

- критерій «схожості» елементів Q;
- використовувана метрика d;
- число кластерів.

Для порівняння двох об'єктів ni та nj можуть бути використані наступні міри:

- Евклідова відстань

$$d(n_i, n_j) = \sqrt{\sum_{l=1}^{K} (x_{il} - x_{jl})^2}$$

- Манхеттенська відстань

$$d(n_i, n_j) = \sum_{l=1}^{K} \left| x_{il} - x_{jl} \right|$$

- Відстань Махалонобіса

$d(n_i, n_j) = (n_i - n_j) C^{-1} (n_i - n_j)^T$, де $C^{-1}$ - коваріаційна матриця вхідних даних;

та інші.

Якість кластеризації можна оцінити за допомогою:

- Середня відстань в середині кластеру

$$Q^{(1)} = \sum_i \sum_{\mathbf{x},\mathbf{y} \in X_i} d(\mathbf{x}, \mathbf{y}) \to \min;$$

- Середня міжкластерна відстань

$$Q^{(2)} = \sum_{i<j} \sum_{\substack{\mathbf{x} \in X_i, \\ \mathbf{y} \in X_j}} d(\mathbf{x}, \mathbf{y}) \to \max;$$

- Сумарна вибіркова дисперсія розкиду елементів відносно центру кластерів

$Q^{(3)} = \sum_{i=1}^{K} 1/\left|X_i\right| \sum_{x \in X_i} d^2(x, c_i) \to min$, де $c_i = 1/\left|X_i\right| \sum_{x \in X_i} x$ - центр кластера $X_i$

Можна також використовувати об'єднаний критерій: $Q^1 / Q^2 \to \min$

За способами кластеризації методи кластерного аналізу можна розділити на дві великі групи: ієрархічні та неієрархічні методи. Ієрархічні методи використовують послідовне об'єднання об'єктів у кластери, малих кластерів у великі, що може бути візуально представлене у вигляді дерева вкладених кластерів - дендрограми (рис. 1.1), при цьому кількість кластерів прихована або умовна. Як правило, на графі дендрограми вздовж осі x розташовують номера об'єктів, а вздовж осі y – значення міри схожості.

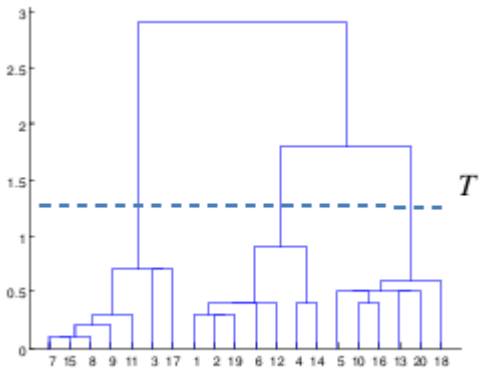


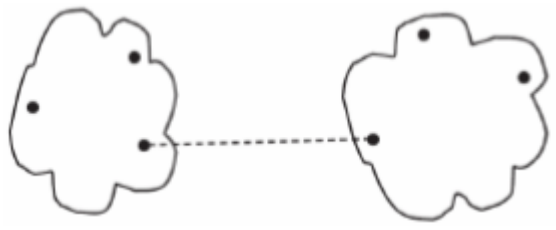

Рис. 1.1 Дендрограма

Алгоритми ієрархічної кластеризації діляться на 2 категорії: зверху вниз або знизу вгору. Знизу вгору алгоритми обробляють кожну точку даних як єдиний кластер на початку, а потім послідовно об'єднують (або агломерують) пари кластерів, поки всі кластери не будуть об'єднані в єдиний кластер, що містить усі точки даних. Тому ієрархічна кластеризація знизу вгору називається ієрархічною агломеративною кластеризацією.

Методи ієрархічного кластерного аналізу відрізняються за способом або методом зв'язування об'єктів у кластери. Якщо кластери *i* та j об'єднуються в кластер *r* та необхідно розрахувати відстань до кластеру *w*, то найбільш часто використовуються такі методи зв'язування об'єктів/кластерів *w* і *s*:

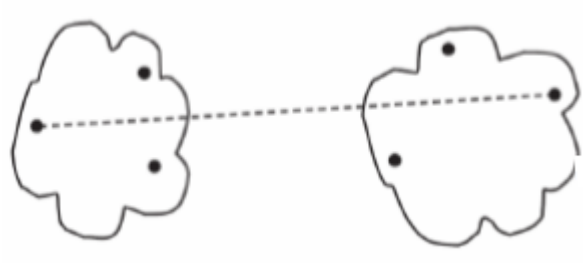
1. метод найближчого сусіда (відстань між найближчими сусідами - найближчими об'єктами кластерів)

$$R^{б}(W,S) = \min_{w \in W, s \in S} \rho(w,s);$$



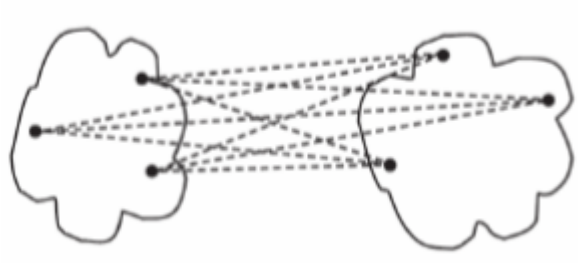
2. метод дальнього сусіда (відстань між найбільш далекими сусідами)

$$R^{д}(W,S) = \max_{w \in W, s \in S} \rho(w,s);$$



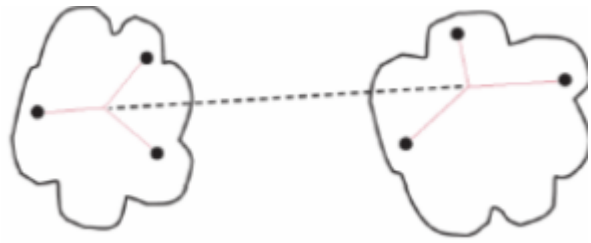
3. метод середнього зв'язку (середня відстань між всіма об'єктами пари кластерів з урахуванням відстані всередині кластерів)

$$R^{г}(W,S) = \frac{1}{|W||S|} \sum_{w \in W} \sum_{s \in S} \rho(w,s);$$



4. Відстань між центрами

$$R^{ц}(W,S) = \rho^2\Big(\sum_{w \in W} \frac{w}{|W|}, \sum_{s \in S} \frac{s}{|S|}\Big);$$



5. При використанні відстані Уорда використовуються методи дисперсійного аналізу - відбувається таке об'єднання кластерів, яке призводить до мінімального збільшення внутрішньогрупової суми квадратів, тобто у великий кластер об'єднуються ті малі кластери, відстань між якими мінімальна.

До неієрархічних методів кластеризації належать K-Means, k-medoid, Mean shift, Affinity propagation, які шукають центри кластерів такі щоб вони представляли найбільш характерні об’єкти кластеру. EM алгоритм моделює суміш розподілів для моделей з прихованими змінними.

## Контрольні завдання

1. В кластеризації k-means виконати вибір кількості кластерів за допомогою коефіцієнту силуету (метод ліктя).
2. Використати для кластеризації алгоритм DBSCAN.

## Контрольні запитання

1. В чому полягає задача кластеризації?
2. Як оцінити якість кластеризації?
3. В чому полягають ієрархічні методи кластеризації?
4. Які неієрархічні методи кластеризації ви знаєте і як вони працюють (Таблиця 1.1)?
5. Як працюють ієрархічні методи?
6. За що відповідає параметр linkage в агломеративній кластеризації?
7. В чому різниця між K-Means та k-medoid?

# Комп’ютерний практикум № 2. Обробка зображень

**Тема:** Обробка зображень

**Мета:** порівняти методи обробки зображень

Хід виконання роботи

1. Завантажити зображення згідно варіанту.
2. Перетворити зображення у відтінки сірого, перевести в формат RGB, HSV. Результати вивести на екран.
3. Вивести на екран зображення у відтінках сірого та його гістограму.
4. Застосувати вирівнювання гістограми та вивести результат зображення та його гістограму на екран.
5. Змінити розмір зображення (зменшити та збільшити) за допомогою піраміди зображень.
6. Виконати згладжування зображення за допомогою фільтра Гауса.
7. Виконати порогову обробку зображення.
8. Виконати масштабування зображення та афінне перетворення зображення. Результати вивести на екран.
9. Виконати ерозію та дилацію над зображення. Використати ядра різного розміру та застосувати функції декілька разів до одного і того ж зображення. Проаналізувати ефект.
10. Побудувати зображення в просторах LoG та DoG. Проаналізувати отриманий ефект.
11. Зробити висновки по роботі та відповісти на наступні питання:
    - Як впливає вирівнювання гістограми, гаусове згладжування на зображення?
    - Навіщо можуть знадобитися всі ці перетворення зображення?

Таблиця 2.1

| **Функція** | **Призначення** |
|---|---|
| blur, GaussianBlur | згладжування |
| pyrUp , pyrDown | Піраміда зображень |
| equalizeHist | Вирівнювання гістограми |
| resize | Масштабування |
| warpAffine, getRotationMatrix2D | Афінне перетворення зображення |
| Erode, dilate | Ерозія та дилація |

Звіт має містити:

- візуальні результати всіх експериментів;
- висновки по роботі;
- текст програми.

## Короткі теоретичні відомості

Піраміда - стандартна структура даних для представлення зображення в різних розмірах. Вихідне зображення знаходиться в основі піраміди, а зображення менших розмірів на наступних рівнях. Якщо кожен раз зменшувати розміри вдвічі, то всі додаткові шари піраміди займають менше третини пам'яті, відведеної під вихідне зображення,

$$1+\frac{1}{2\cdot 2}+\frac{1}{2^2\cdot 2^2}+\frac{1}{2^3\cdot 2^3}+\cdots<\frac{4}{3}.$$

При переході до наступного рівня від низу до верху середнє значення 2x2 пікселів стає значенням пікселя наступного рівня. Щоб уникнути просторових спотворень у вигляді сходинок рекомендується перед обчисленням середнього зробити гаусове згладжування.

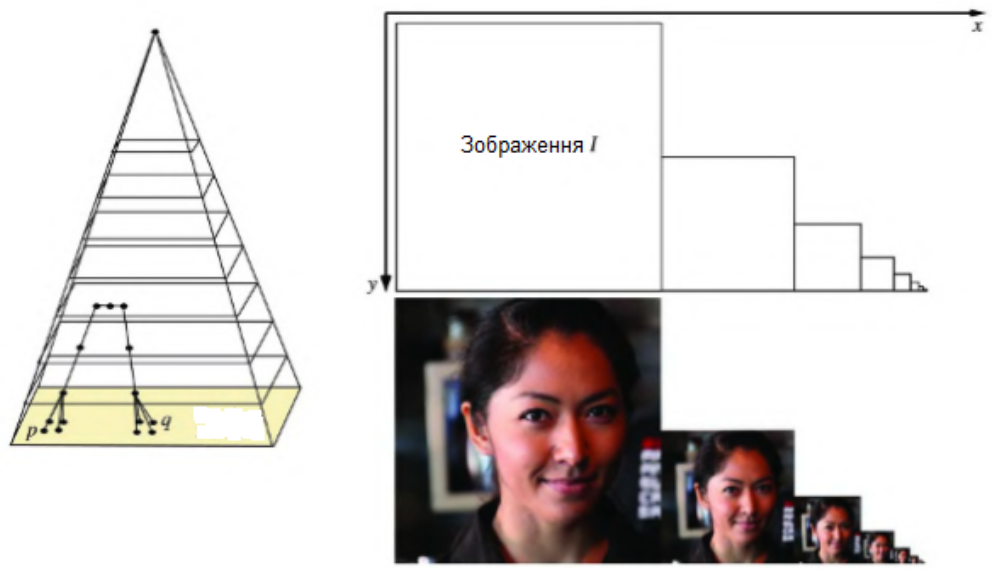



Рис. 2.1

Гістограма цифрового зображення з діапазоном значень яскравості пікселів [0, L-1] є дискретна функція: $h(r_k)=n_k$, де $r_k$ - k-ий рівень яскравості; $n_k$ - число пікселів на зображенні. Тобто рахується скільки разів таке значення яскравості зустрічається на зображення і так для кожного рівня яскравості. Можна визначити гістограму для кожного колірного діапазону зображення.

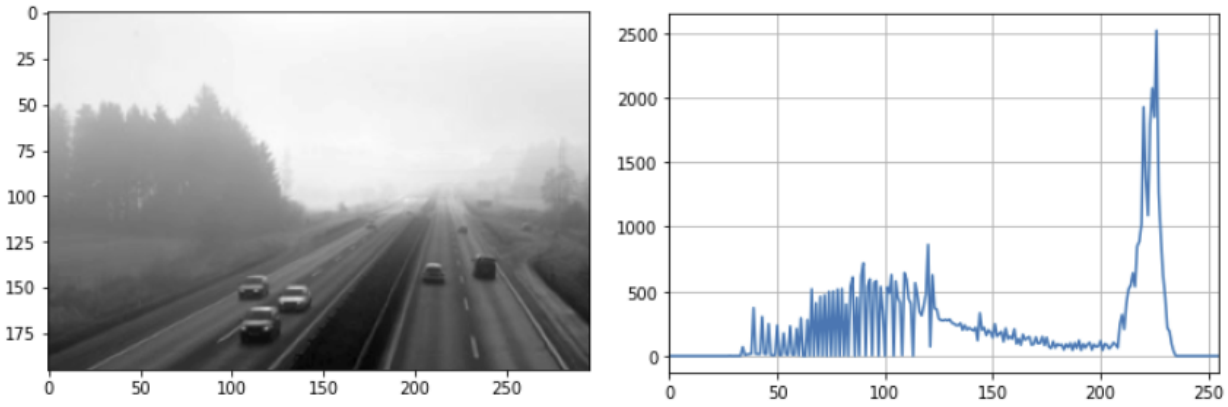



Рис. 2.2 Гістограма зображення

Мета згладжування зображення - усунення «викидів», які вважаються шумом. Фільтр Гаусса є локальною згорткою з ядром, визначеного вибірками з двовимірної функції Гауса. Ця функція є добутком двох одновимірних гаусових функцій:

$$G_{\sigma,\mu_x,\mu_y}(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x-\mu_x)^2+(y-\mu_y)^2}{2\sigma^2}\right) =$$

$$= \frac{1}{2\pi\sigma^2}\cdot e^{-\frac{(x-\mu_x)^2}{2\sigma^2}}\cdot e^{-\frac{(y-\mu_y)^2}{2\sigma^2}},$$

де ($\mu_x$ $\mu_y$)- математичне сподівання по осям, σ - стандартне відхилення, яке ще називається масштабом. Можно побудувати цілий простір зображень гаусівського масштабу застосовуючи різні значення стандартного відхилення.

Гаусівський простір масштабів: L(x,y, aσ), де a зростає, a>1, $a^n\sigma$, n=0,1,2,...m.

Іноді корисним для аналізу зображення є побудова просторів LoG та DoG.

Щоб отримати простір LoG необхідно застосувати до зображення гаусове згладжування з різним масштабом, а потім до кожного рівня масштабу застосувати оператор лапласа. Якщо I - зображення, G - гаусове згладжування, то $LoG(I)=\nabla^2(G_\sigma(I))$.

Щоб отримати простір DoG необхідно застосувати гаусове згладжування з різними значеннями масштабу та знайти їх різницю.

Морфологічні операції-це набір математичних функцій, нелінійних фільтрів, які обробляють зображення на основі морфології або форми. Морфологічні операції виконують над зображенням з використанням структурного елементу. Структурний елемент - деяке двійкове зображення певної форми, зазвичай розміром 3x3 чи 5x5. Структурний елемент ковзає по зображенню і для ерозії справедливо наступне: піксель у вихідному зображенні вважатиметься одиницею, лише якщо всі пікселі під структурним елементом дорівнюють 1, інакше він стає рівним нулю. Ерозія буде видаляти білі шуми та зменшувати об’єкт. Для дилації справедливо наступне: піксель у вихідному зображенні вважатиметься одиницею, якщо принаймні один піксель під структурним елементом дорівнює 1, інакше він стає рівним нулю. При застосувані дилації збільшується біла область на зображенні або збільшується розмір об’єкта переднього плану.

## Контрольні завдання

1. Обрати зображення та виділити на ньому тільки певний колір. Наприклад вивести на екран лише червоний колір що міститься на зображенні (створити та використати маску певного кольору).
2. Вибрати зображення з чіткими контурами та виділити контури за допомогою LoG та DoG
3. Виконати морфологічну операцію розкриття - спочатку застосувати ерозію, потім диляцію. Проаналізувати ефект.
4. Виконати морфологічну операцію зазкриття - спочатку застосувати диляцію, потім ерозію. Проаналізувати ефект.
5. Обрати два подібні зображення, перейти в Гаусівський простір з’єднати половинки зображень в цьому просторі і повернутися в початковий. Проаналізувати результати.

## Контрольні запитання

1. Що таке гістограма зображення? Як і навіщо відбувається вирівнювання гістограми?
2. Що таке гаусове згладжування? Для чого воно використовується?
3. Які функції для перетворення зображення ви знаєте (Табл.2.1)? Як вони працюють?
4. Що таке піраміда зображень?

# Комп’ютерний практикум № 3. Сегментація зображень

**Тема:** Сегментація зображень

**Мета:** провести порівняльний аналіз різних алгоритмів кластеризації для сегментації зображень з класичними методами

Хід виконання роботи

1. Завантажити зображення згідно варіанту.
2. Визначити методи, що будуть використовуватися для попередньої обробки зображень, таблиця 3.1. Провести подальші експерименти з різними методами (або комбінацією).
3. По черзі застосувати наступні методи сегментації:
    - EM
    - Agglomerative clustering
    - Mean shift
    - Fuzzy K-Means
    - K-Means ++
    - Метод росту регіону (реалізувати)
    - Grab cut -https://docs.opencv.org/4.5.2/d3/d47/group__imgproc__segmentation.html#ga3267243e4d3f95165d55a618c65ac6e1
    - Flood fill -https://scikit-image.org/docs/dev/api/skimage.segmentation.html?highlight=flood_fill#skimage.segmentation.flood_fill
    - Watershed - https://docs.opencv.org/4.5.2/d3/d47/group__imgproc__segmentation.html#ga3267243e4d3f95165d55a618c65ac6e1
4. Для кожного алгоритму сегментації порахувати метрики якості:

- adapted_rand_error - https://scikit-image.org/docs/dev/api/skimage.metrics.html#skimage.metrics.adapted_rand_error
- variation_of_information - https://scikit-image.org/docs/dev/api/skimage.metrics.html#skimage.metrics.variation_of_information.

Для цих метрик необхідно заздалегідь визначити кластери на зображенні, тобто необхідно позначити пікселі, які відносяться до одного сегменту за допомогою функції skimage.measure.label. Тьюторіал можна знайти https://scikit-image.org/docs/stable/auto_examples/segmentation/plot_metrics.htm. Звести результати в одну таблицю.

5. Зробити висновки по роботі та відповісти на наступні питання:
   - який з алгоритмів спрацював краще за інші? Чому?
   - яка з метрик кластеризації на вашу думку краще оцінює результат? Чому?
   - Як впливає попередня обробка на результат?

Таблиця 3.1

| **Методи** | **Функція** | **Посилання** |
|---|---|---|
| Морфологічна обробка | cv.erode(), cv.dilate(), cv.morphologyEx() | https://docs.opencv.org/4.5.2/d4/d86/group__imgproc__filter.html#ga67493776e3ad1a3df63883829375201f |
| Гаусовський фільтр | Gaussian або gaussian_filter | https://scikit-image.org/docs/dev/api/skimage.filters.html?highlight=gaussian_filter#skimage.filters.gaussian<br>або<br>https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.gaussian_filter.html#sci |

| | | |
|---|---|---|
| | | py.ndimage.gaussian_filter |
| Порогова обробка | Порогова обробка та бінаризація Оцу разом | https://docs.opencv.org/4.5.2/d7/d4d/tutorial_py_thresholding.html |

Звіт має містити для кожного експерименту:

- зображення після попередньої обробки;
- сегментоване зображення;
- значення метрик в таблиці;
- висновки по роботі;
- текст програми.

## Короткі теоретичні відомості

Сегментація зображень – це процес виділення областей на зображенні, що поєднані певною ознакою. Ознакою може бути колір, чи колір та розташування тощо.

Таким чином, якщо кожен $i$-й атомарний елемент зображення (піксель) представити у вигляді вектору $\overrightarrow{x_i}^T = (x_{i1},...,x_{im})$ у просторі станів зображення (де $x_{ij}$ – $j$-й вимір простору стану), то означення сегментації зображень можна переписати у вигляді:

$$C = f(X),$$

де $X = \left\| x_{ij} \right\|_{i=1,...,n}^{j=1...m}$ – матриця значень $j$-х компонент у просторі станів $i$-х атомарних елементів зображення, $C = (c_1,...,c_n)$ – вектор класифікації атомарних об’єктів зображення: $c_i \in A$ – номер об’єкту на зображенні, до якого класифіковано $i$-й піксель; $A \subset N$ – множина номерів об’єктів на зображенні.

$f$ – деяке функціональне перетворення з простору станів атомарних елементів зображення у $N^n$.

У відповідності до означення, задача сегментації зображення зводиться до відшукання відображення $f$, що відповідає задачі чіткої кластеризації у просторі станів атомарних елементів зображення. Методи, що базуються на кластеризації, сегментують зображення на кластери або несумісні групи пікселів із подібними характеристиками.

Можна виділити наступні типи сегментації:

- Семантична сегментація — класифікує всі пікселі зображення на значущі класи об'єктів. Ці класи «семантично інтерпретуються» і відповідають реальним категоріям. Наприклад, ви можете виділити всі пікселі, пов'язані з котом, і забарвити їх у зелений колір.
- Сегментація екземпляра (instance)- ідентифікує кожен екземпляр кожного об'єкта на зображенні. Він відрізняється від семантичної сегментації тим, що не класифікує кожен піксель. Якщо на зображенні є три машини, семантична сегментація класифікує всі машини як один примірник, тоді як сегментація екземпляра ідентифікує кожен окремий автомобіль.
- Паноптична сегментації - класифікує всі пікселі зображення співставляючи їм мітки класу, але також визначиє, до якого екземпляра цього класу вони належать.

Також методи сегментації можна поділити на такі види:

- Підхід на основі виявлення подібності - виявлення подібних пікселів на зображенні і об'єднанні їх в один сегмент
- Підхід на основі виявлення розривів (границь) - на зображенні виявляються межі об'єктів, які зададуть границі сегментів.

До підходів на основі подібності належить метод росту регіону. Він полягає у тому що на початку обирається початкові пікселі (центри регіону) і поступово додаємо до нього сусідні пікселі, що задовольняють «критерію схожості» на пікселі, що становлять уже вирощену ділянку навколо початкового пікселя. Цей

процес можна повторювати, поки кожен піксель зображення не виявиться в одному із сегментів.

Для методу водорозділу визначаються початкові точки, які обираються серед мінімальних значень інтенсивності. При цьому кількість вихідних областей визначається наперед. Будь-яке зображення у відтінках сірого можна розглядати як топографічну поверхню, де висока інтенсивність позначає вершини та пагорби, а низька інтенсивність позначає долини. Відповідно можна сказати, що алгоритм починає заповнювати (піднімати рівень інтенсивності) кожну ізольовану долину (місцеві мінімуми) водою різного кольору (призначати мітки). Коли вода піднімається до вершин, вода з різних долин, різного кольору, почне зливатися. Щоб уникнути цього, алгоритм будує бар'єри (водорозділи) в місцях злиття води. Алгоритм продовжує наповнювати долини водою та будувати перешкоди, доки всі вершини не опиняться під водою. Тоді бар’єри, які побудовані, дають результат сегментації.

За допомогою алгоритма затоплення, (FloodFill, Seed Fill) можна виділити однорідні за кольором регіони. Для цього обирається початковий піксель і задається інтервал зміни кольору сусідніх пікселів щодо вихідного. Алгоритм буде об'єднувати пікселі в один сегмент (заливаючи їх одним кольором), якщо вони потрапляють в зазначений діапазон. На виході буде сегмент, залитий певним кольором.

## Контрольні завдання

1. Виконати порогову обробку зображення. Застосувати бінаризацію Оцу і використати утворену маску для сегментації.
2. Реалізувати алгоритм split and merge.

## Питання для самоконтролю

1. В чому полягає задача сегментації?
2. Як працює метод водорозділу?
3. Як працює алгоритм затоплення?

4. Як працює метод росту регіонів?
5. Як працюють методи сегментації використані в лабораторній роботі?

# Комп’ютерний практикум № 4. Виділення меж на зображенні

**Тема:** методи виділення меж зображення.

**Мета:** провести порівняльний аналіз методів виділення меж: детектор меж Кенні, детектор меж за допомогою оператора Лапласіана, детектор меж за допомогою оператора Собеля, детектор меж за допомогою оператора Превітта, детектор меж за допомогою оператора Робертса та HED.

Хід виконання роботи

1. Завантажити та відповідно обробити зображення для подальшого використання. Варіанти зображень згідно номеру варіанту.
2. Визначити межі: за допомогою адаптивного порогу обробити зображення і знайти контури (findContours).
3. Визначити межі зображення методами Собеля, Превітта, Робертса, та за допомогою оператора Лапласіана. Навести результати роботи.
4. Визначити межі зображення методом Кенні досліджуючи різні значення σ та параметри гістерезису. Відобразити результати.
5. Визначити межі зображення за допомогою HED.
6. Отримані зображення оцінити візуально. Кожне зображення повинно отримати оцінку від 1 до 10, що визначає якість виділення меж.
7. Зробити висновки по роботі та відповісти на наступні питання:
    - який з алгоритмів спрацював краще за інші? Чому?
    - Які параметри для методу Кенні виявилися найкращими? Чому?
    - Як впливає попередня обробка на результат?

Звіт має містити для кожного експерименту:

- зображення та результати визначення меж, а також оцінку;
- висновки по роботі;
- текст програми.

Короткі теоретичні відомості

Сегментація на основі контурів передбачає наявність перепадів яскравості які позначають межі зображення (об'єкта на зображенні).

При сегментації представляють інтерес перепади яскравості, обумовлені межами об'єктів, оскільки метою сегментації, як правило, є виділення об'єктів на зображеннях. Перепади цього типу в ідеалі є різкі скачки яскравості, однак на практиці частіше доводиться мати справу з розмитими світловими межами внаслідок апаратурних спотворень внесених системою. На практиці застосовують два методи виявлення перепадів яскравості, утворених світловими межами: градієнтний метод і метод з використанням Лапласіан.

За визначенням градієнт аналогового зображення L (x, y) в точці x, y є вектор G, орієнтований в напрямку максимального зміни яскравості, модуль якого дорівнює

$$|\mathbf{G}| = \sqrt{G_x^2 + G_y^2}\ , \quad \text{где } G_x = \partial L_c(x,y)/\partial x,\ G_y = \partial L_c(x,y)/\partial y$$

компоненти цього вектора.

У разі дискретних зображень часткові похідні Gx і Gy визначаються наближено одним із таких способів:

Перехресний градієнтний оператор Робертса. Відповідно до цього способу дискретне зображення сканується вікном розміром 2 × 2 пікселя, яке показано на рис. 4.1.

| | |
|---|---|
| $L_c(k,n)$ | $L_c(k,n+1)$ |
| $L_c(k+1,n)$ | $L_c(k+1,n+1)$ |

Рис.4.1

Для кожного положення вікна обчислюються значення Gx і Gy за формулами

$$G_x = L_c(k+1, n+1) - L_c(k, n),$$

$$G_y = L_c(k+1, n) - L_c(k, n+1),$$

а потім за формулою для пікселя, розташованого в k-му рядку і в n-му стовпці, обчислюється модуль градієнта.

При використанні оператора Превітта дискретне зображення сканується вікном розміром 3 × 3 пікселя, при цьому для кожного положення вікна значення Gx і Gy обчислюються за формулами

$$G_x = (L_2 + L_3 + L_4) - (L_0 + L_6 + L_7),$$

$$G_y = (L_4 + L_5 + L_6) - (L_0 + L_1 + L_2),$$

де як і раніше L0, L1, L2, L3, L4, L5, L6, L7, L8 - значення яскравості пікселів, що опинилися в межах вікна. Далі обчислюється модуль градієнта по формулі для пікселя зображення, що знаходиться в центрі скануючого вікна.

Оператор Собеля відрізняється від оператора Превітта тим, що при обчисленні компонентів градієнта Gx і Gy значення відліків яскравості L1, L3, L5 і L7 беруться з ваговим коефіцієнтом 2, а значення Gx і Gy для кожного положення вікна обчислюються за формулами

$$G_x = (L_2 + 2L_3 + L_4) - (L_0 + L_6 + 2L_7),$$

$$G_y = (L_4 + 2L_5 + L_6) - (L_0 + 2L_1 + L_2).$$

Після цього за формулою обчислюється модуль градієнта для пікселя зображення, що знаходиться в центрі скануючого вікна. Збільшення ваги відліків яскравості L1, L3, L5 і L7 при використанні оператора Собеля дозволяє дещо зменшити вплив шуму на результат обчислення градієнта.

Лапласіан аналогового зображення Lc (x, y) в точці x, y визначається виразом

$$\Delta(x, y) = \partial^2 L_c(x, y) / \partial x^2 + \partial^2 L_c(x, y) / \partial y^2 .$$

Лапласіан дискретного зображення Lc (k, n) для пікселя, розташованого в k-му рядку і в n-м стовпці визначається наближено одним з наступних двох способів $\Delta(k,n) = 4L_8 - (L_1 + L_3 + L_5 + L_7)$ або

$$\Delta(k,n) = 8L_8 - (L_0 + L_1 + L_2 + L_3 + L_4 + L_5 + L_6 + L_7).$$

У першому випадку результат виявляється інваріантним до повороту на кути, кратні 90 °, у другому випадку інваріантним до повороту на кути 45 °.

На межу буде вказувати значення градієнта. Щоб визначити межу необхідно позначити пікселі як межу. Тобто в якості межі будемо позначати точки локального максимуму в напрямку градієнта, значення градієнта буде перевищувати деякий поріг. В результаті такої обробки зображення виходить його контурне представлення. Виділені контури на контурному представленні зазвичай розірвані в багатьох місцях, крім того, на ньому є точки і штрихи, які сприймаються зоровою системою як шумний фон. Вибираючи величину порога, можна дещо зменшити цей ефект однак зовсім від нього позбутися не вдається. Чим вище поріг, тим менше на контурному зображенні буде окремих точок і штрихів, які не є елементами контурів що виділяються, однак при цьому в виділених контурах збільшується кількість і протяжність розривів. Зменшення величини порога призводить до зворотної картини.

Для того щоб краще визначити межу необхідно зменшити шум на зображення. Це можна зробити за допомогою Гаусового згладжування. Параметром згладжування виступає значення масштабу σ. Зазвичай будь-який вибір σ не дає хорошої карти меж: велике σ створить краї, що утворюють лише найбільші об’єкти, і вони не будуть точно розмежовувати об’єкт, оскільки згладжування зменшує деталі форми; невелике σ створить множину меж і дуже нерівні межі багатьох об’єктів.

Метод Кенні створений щоб впоратися з цими обмеженнями. Він переслідує три основні цілі:

1. Низька частота помилок. Повинні виявлятися всі контури з мінімумом

помилкових спрацьовувань.

2. Хороша локалізація контурних точок.

3. Одиночний відгук на точку контуру. Для кожної точки істинного контуру детектор повинен виявляти тільки одну точку.

Етапи методу Кенні

1. Згладжування зображення за допомогою гаусової фільтрації

$$h(x, y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right),$$

де σ - параметр, що визначає ступінь згладжування на присутній в зображенні шум.

2. Знаходження градієнта зміни яскравості в зображенні:

$$|\mathbf{G}| = \sqrt{G_x^2 + G_y^2}\ ,\ \text{где}\ G_x = \partial L_c(x, y)/\partial x\ ,\ G_y = \partial L_c(x, y)/\partial y\ .$$

3. Порогова обробка результатів обчислення градієнтів в кожній точці зображення. При цій обробці здійснюється так зване «не-максимальне" пригнічення стрибків яскравості зображення (non-maximal suppression), в результаті якого зберігаються тільки ті значення обчислених градієнтів, які перевищують значення градієнтів в двох сусідніх точках на зображенні у напрямку градієнта зображення. Іншими словами, зберігаються тільки значення градієнтів в точках максимальної крутості зміни яскравості на межах яскравості.

4. Морфологічна обробка результатів, отриманих на попередньому кроці алгоритму. При цій обробці задаються два порога: нижній і верхній (параметри гістерезису). При формуванні контурів всі точки, що перевищили верхній поріг, зберігаються. Що ж стосується точок, які перевищили нижній поріг, то зберігаються тільки ті точки, які безпосередньо є сусідами з точками, що перевищили верхній поріг, інші ж виключаються з формованого контурного зображення.

Перелік стандартних функцій наведено в таблиці 4.1

Таблиця 4.1

| **Стандартні функції** | **Посилання на тьюторіал** |
|---|---|
| Детектор меж Кенні | 1. https://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html?highlight=canny#canny<br>2. https://docs.opencv.org/3.4.3/da/d5c/tutorial_canny_detector.html |
| Детектор меж за допомогою оператора Собеля | https://docs.opencv.org/3.4.3/d2/d2c/tutorial_sobel_derivatives.html |
| Детектор меж за допомогою оператора Лапласіана | https://docs.opencv.org/3.4.3/d5/db5/tutorial_laplace_operator.html |
| Створення фильтру користувача (використовуйте для створення детектору меж Первітта та Робертса) | https://docs.opencv.org/3.1.0/d4/d13/tutorial_py_filtering.html |
| Holistically-Nested Edge Detection | https://paperswithcode.com/paper/holistically-nested-edge-detection |
| Пошук контурів | https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contours_begin/py_contours_begin.html |

Контрольні завдання

1. Вибрати зображення з певною геометричною фігурою. Застосувати функцію знаходження контурів та порахувати периметр і площу фігури. Також побудувати обмежуючий прямокутник навколо фігури.

## Контрольні запитання

1. Що показує похідна зображення?
2. Як можна порахувати похідну зображення?
3. Як працюють оператор Робертса, оператор Собеля, оператор Превітта, оператор Лапласіана.
4. Як працює метод Кенні?
5. Що таке гістерезис в методі Кенні?
6. Як відбувається не максимальне пригнічення стрибків яскравості в методі Кенні?

# Комп’ютерний практикум № 5. Розпізнавання облич

**Тема:** Розпізнавання облич на зображеннях класичними методами

**Мета:** провести порівняльний аналіз різних алгоритмів локалізації обличчя на зображенні для подальшого розпізнавання

Хід виконання роботи

1. Створити вибірку фотографій відомих особистостей, яка містить різнопланові фотографії особистостей, фотографії схожих людей(декілька), фотографії інших людей. За необхідністю використати вже створені вибірки.
2. За необхідності зменшіть розмір зображень.
3. Зафіксувати час роботи алгоритмів та звести ці дані в таблицю.
4. Використати метод Віоли-Джонса для виділення областей, що містять обличчя, встановити різні параметри minNeighbors, scaleFactor.
5. Розділити отриману вибірку з виділеними обличчями на тестову та перевірочну.
6. За необхідності використайте методи вилучення ознак (наприклад PCA) для швидшої роботи алгоритмів.
7. Використати RandomForestClassifier для розпізнавання обраної знаменитості.
8. Повторити пункт 7 для SVM, MLPClassifier та запропонуйте свій класифікатор.
9. Порахувати точність, метрики precision та recall на обох вибірках. Звести результати в таблицю.
10. Зробити висновок, щодо впливу параметрів minNeighbors, scaleFactor на розпізнавання.
11. Для виділення областей що містять обличчя використати HOG детектор та повторити пункти 3-6.

12. Зробити висновки по роботі та відповісти на наступні питання:
    - який з алгоритмів знаходження обличчя спрацював краще за інший? Чому?
    - який з алгоритмів розпізнавання обличчя виявився краще за інший?
    - Який з варіантів швидше відпрацював?

Звіт має містити:

- опис створеної вибірки;
- результати експериментів зведені в таблиці;
- висновки по роботі;
- текст програми.

## Короткі теоретичні відомості

Локалізація обличчя – знайти обличчя (розташування та розмір) на зображенні та, можливо, виділити їх для використання алгоритмом розпізнавання облич.

Розпізнавання обличчя - зображення обличчя (обрізані, змінені та зазвичай перетворені у відтінки сірого) надаються алгоритму розпізнавання облич, який відповідає за пошук характеристик, які найкраще описують зображення.

Метод Віоли Джонса використовує технологію ковзного вікна. Тобто рамка, розміром, меншим, ніж вихідне зображення, рухається з деяким кроком по зображенню, і за допомогою каскаду слабких класифікаторів визначає, чи є в даному вікні обличчя.

Основні принципи, на яких базується метод, такі:

- використовуються ознаки Хаара, за допомогою яких відбувається пошук потрібного об'єкта (в даному контексті, обличчя і його риси);
- використовуються зображення в інтегральному уявленні, що дозволяє обчислювати швидко необхідні об'єкти;
- використовується бустинг для вибору найбільш підходящих ознак для шуканого об'єкта на даній частині зображення;
- всі ознаки надходять на вхід класифікатора, який дає результат «вірно» або «ні»;

- використовуються каскади ознак для швидкого відкидання вікон, де не знайдено обличчя.

Інтегральне представлення можна представити у вигляді матриці, розміри якої збігаються з розмірами вихідного зображення I (або лише прямокутного вікна), де кожен елемент розраховується для пікселя p = (x, y) як сума пікселів що знаходяться вище та лівіше від пікселя p. Для кожного пікселя:

$$I_{int}(p) = \sum_{1 \le i \le x \wedge 1 \le j \le y} I(i,j)$$

Розглянемо прямокутне вікно W (рис. 5.1), яке визначається чотирма пікселями p, q, r, s, так що q, r і s розташовані від границі W на один піксель.

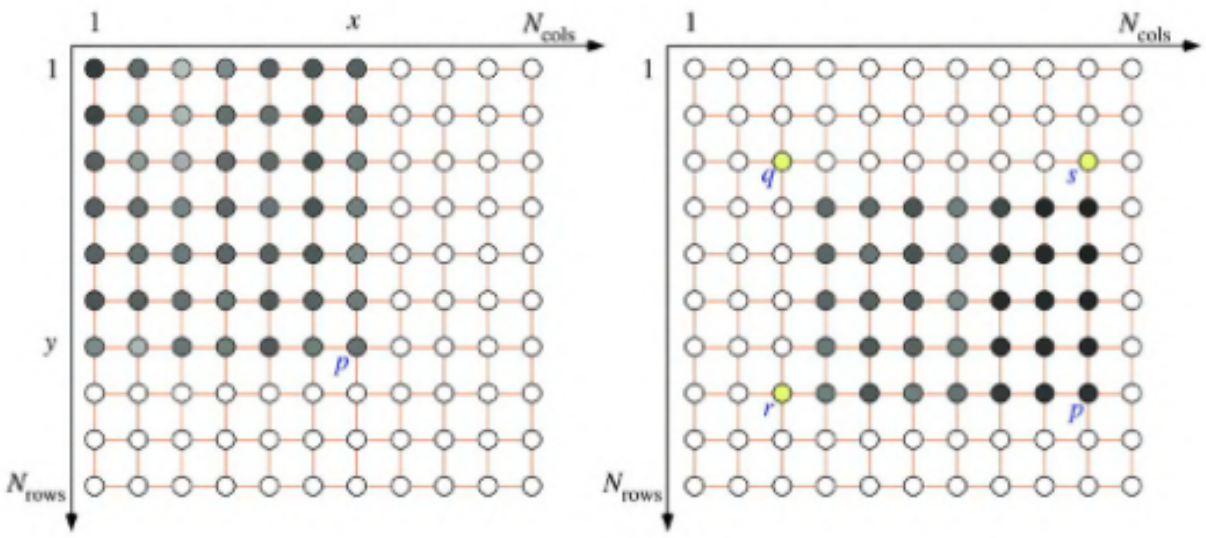



Рис.5.1

Тоді сума Sw значень всіх пікселів всередині W дорівнює:

$$S_w = I_{int}(p) - I_{int}(r) - I_{int}(s) + I_{int}(q)$$

Незалежно від розміру вікна W ми повинні виконати тільки одне додавання і два віднімання.

Ознаки Хаара, які використовуються для побудови слабких класифікаторів представляють собою області що складаються зі світлих і темних областей. Кожна ознака характеризується розміром світлої і темної областей, пропорціями, а також мінімальним розміром (Рис.5.2). Білому пікселю в ознаці відповідає вага +1, а чорному - вага -1. Тобто сумуються пікселі, що потрапили

під білу область і від цієї суми віднімається сума пікселій, які потрапили під темну область. Ознаки застосовуються для встановлення «приблизного подібності яскравості» ділянки зображення з такими паттернами.

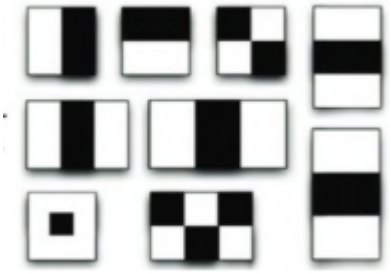


Рис. 5.2

На першому етапі алгоритму будується w слабких класифікаторів на основі ознак Хаара. Далі використовується бустинг для отримання сильного класифікатора. Далі застосовується каскад таких класифікаторів. Використовується принцип ковзного вікна, коли по зображенню “пробігають” вікном меншого розміру ніж зображення, в якому застосовують класифікатор. Якщо об’єкт виявлено, то ця область надходить на вхід наступного класифікатора, інакше вікно відкидається.

Основні етапи локалізації обличчя за допомогою HOG дескриптора:

1. Необов'язкова попередня нормалізація зображень. В результаті виходять ознаки, слабо залежать від змін освітленості.
2. Операція згортання зображення за допомогою двох фільтрів, чутливих до горизонтальних і вертикальних перепадів яскравості. Це дозволяє вловити інформацію про межі, контури і текстури зображення.
3. Розбивка зображення на комірки заздалегідь визначеного розміру і обчислення гістограми напрямків градієнтів в кожній з комірок.
4. Нормалізація гістограм в кожній з комірок шляхом порівняння з декількома сусідніми комірками. Це ще більше пригнічує вплив освітленості на зображенні.
5. Формування одновимірного вектора ознак з інформацією по кожній комірці.

6. За допомогою вибірки з таких ознак навчити лінійний SVM-класифікатор для детекції обличчя.

Гістограма напрямків градієнтів будується наступним чином:

Для кожної комірки вираховується напрямок та значення градієнта. Напрямки градієнтів розбиваються на блоки наприклад на 9 інтервалів шириною 20°, що покривають весь діапазон 180°. Значення градієнтів “розподіляються” по цим коміркам у відповідності до напрямку. В результаті виходять вектори (наприклад, довжини 9) для кожної комірки. Для ефективного обчислення сум можна скористатися інтегральними зображеннями.

## Контрольні завдання

1. Використати дескриптор LBP для знаходження обличчя на зображенні.
2. За допомогою методу Віоли Джонса виділити посмішку на зображенні.
3. За допомогою методу HOG дескриптора виділити очі на зображенні.

## Контрольні запитання

1. В чому полягає задача розпізнавання обличчя?
2. Які основні принципи алгоритму Віоли Джонса?
3. В чому полягає кожний етап алгоритму Віоли Джонса?
4. Що таке HOG дескриптор?
5. Як будується гістограма напрямків градієнтів?
6. Як за допомогою HOG дескриптора розпізнати особу на зображенні?

# Комп’ютерний практикум № 6. Згорткові нейронні мережі

**Тема:** Розпізнавання облич за допомогою згорткових нейронних мереж.

**Мета:** вивчити різні шари та параметри згорткових нейронних мереж та провести порівняльний аналіз різних архітектур згорткових нейронних мереж для аналізу зображень.

Хід виконання роботи

1. Якщо ви виконуєте завдання в Google Colab і хочете використати GPU для прискорення навчання, то в ноутбуці Google Colaboratory встановити Runtime -> Change Runtime Type GPU. Або ви можете скористатися тьюторіалом https://www.tensorflow.org/guide/gpu.
2. Завантажити оброблені дані з лабораторної №5 (задача розпізнати знаменитість).
3. Для створення власної згорткової мережі необхідно лаштувати наступні параметри: кількість шарів мережі, кількість фільтрів на кожному рівні, наявність та параметри dropout, вид функції активації, наявність batchNormalization, алгоритм навчання, швидкість навчання (learning rate), вид пулінгу, кількість прихованих шарів в повнозв’язному шарі, кількість епох для навчання і так далі. В якості експерименту будемо налаштовувати наступні параметри: кількість шарів, наявність та параметри dropout, вид функції активації. Всі інші параметри в рамках експериментів оберіть одні і ті ж. Вихідним шаром встановіть softmax. Тобто необхідно провести експерименти в різній конфігурації та обрати найкращу (оберіть метрики порівняння). Параметри:
    - Кількість шарів min=2, max=6, step=1
    - Наявність та параметри dropout min= 0, max=0.5, step=0.1
    - Вид функції активації - relu, lrelu, selu, mish

Результати експериментів зведіть в табличку. Оберіть кращий результат.

4. Для того ж набору даних для підбору гіперпараметрів використати вбудовану бібліотеку Keras Tuner. Для встановлення:

   import tensorflow as tf

   from tensorflow import keras

   pip install -q -U keras-tuner

   import keras_tuner as kt

   Спочатку підберіть ті ж самі параметри, що і в пункті 3. Порівняйте результат з кращим результатом пункту 3.
5. Виконати підбір параметрів за допомогою бібліотеки Keras Tuner для таких параметрів:
   - Кількість шарів min=2, max=6, step=1
   - Наявність та параметри dropout min= 0, max=0.5, step=0.1
   - Вид функції активації - relu, lrelu, selu, mish
   - Швидкість навчання
   - Вид шару пулінгу - AvgPool2D, MaxPool2D
   - Наявність BatchNormalization
   - Кількість фільтрів в кожному шарі

Внести в звіт отриману архітектуру та порівняти всі результати і зробити висновок.

6. Використати вбудовані архітектури згорткових нейронних мереж для розпізнавання знаменитості, на власних даних. Використовуйте вже попередньо начені ваги. Порівняти якість та швидкість наступних мереж VGG16, InceptionV3, ResNet50 та найкращого вашого варіанту. Результати занести в таблицю.
7. Зробити висновки по роботі та відповісти на наступні питання:
   - Яка з мереж показала найкращий результат та чому?
   - Як впливає на результат кількість шарів, dropout, вид функції активації і т.д. ?

Звіт має містити:

- опис створених мереж;
- результати експериментів зведені в таблиці;
- висновки по роботі;
- текст програми.

Короткі теоретичні відомості

Основними шарами ЗНМ є шар згортки та шар пулінгу. Згортка - процес застосування фільтра («ядра») до зображення. Ядро - обмежена матриця ваг невеликого розміру, яку «рухають» по всьому оброблюваного шару (на самому початку - безпосередньо по вхідному зображенню), формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічною позицією. Функція активації додає нелінійності. Деякі функції активації:

- Relu (rectified linear unit) визначена наступним чином

f(x)=max(0, x)

- Softplus визначена наступним чином

f(x)=log(1+exp(x))

- Lrelu (leaky relu) визначена наступним чином

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise} \end{cases}$$

- Elu (exponential linear units) визначена наступним чином

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ a\left(e^x - 1\right) & \text{otherwise} \end{cases}$$

- Selu (Scaled Exponential Linear Unit) визначена наступним чином

$$f(x) = \lambda x \text{ if } x \geq 0$$

$$f(x) = \lambda\alpha(\exp(x) - 1) \text{ if } x < 0$$

- Mish визначена наступним чином

$$f(x) = x \tanh\left(\text{softplus}(x)\right)$$

Операція пулінгу - це процес стиснення (зменшення розмірів) зображення шляхом складання значень блоків пікселів. Початкове зображення ділиться на блоки розміром w × h і для кожного блоку обчислюється деяка функція. Найчастіше використовується функція максимуму або (зваженого) середнього.

Dropout - це підхід до регуляризації нейронних мереж, який допомагає зменшити взаємозалежне навчання нейронів. Підхід полягає у випадковому відключенні кожного нейрона на заданому шарі з ймовірністю p на кожній епосі. Після навчання мережі, на стадії розпізнавання, ваги шарів, до яких був застосований dropout, повинні бути помножені на 1 / p.

BatchNormalization - це техніка нормалізації, яка виконується між шарами нейронної мережі, а не в необроблених даних. Це робиться з використанням miniBatch замість повного набору даних. BatchNormalization служить для прискорення навчання та дозволяє використовувати більше значення швидкості навчання, що полегшує навчання.

Для створення власної згорткової мережі в keras:

Визначення послідовної моделі model = Sequential()

Послідовне додавання шарів model.add

Шар згорки Conv2D

Шар функції активації Activation

Шар субдисретизації MaxPooling2D

Шар Flatten перетворює двовимірний вектор виходу ЗНМ в одновимірний

Повнозв’язний шар Dense

Шар Dropout для зменшення перенавчання

Компіляція мережі model.compile

Навчання мережі model.fit

Приклад побудованої простої мережі:

```
model = Sequential()
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1))
model.add(Activation('sigmoid'))
```

Трансферне навчання (Transfer learning) - це метод машинного навчання при якому модель, розроблена для певної задачі, повторно використовується як відправна точка для моделі для другої задачі. В глибокому навчання для того щоб гарно навчити глибоку нейронну мережу необхідно мати великий обсяг даних, якого може не бути для поточної задачі. Тож можна використати досвід набутий в іншій задачі для пришвидшення пошуку розв’язку поточної задачі. Так для навчання нейронної мережі розпізнавати наявність пожежі на зображенні, в якості відправної точки можна обрати вже навчену мережу на іншому наборі даних. Таким чином ваги нейронів мережі фіксується і далі мережа донавчається на наборі даних для поточної задачі.

## Контрольні завдання

1. Дослідити вплив різних алгоритмів навчання мереж на результати класифікації.
2. Вивести на екран карту активації нейронів.

Контрольні запитання

1. Що таке згорткова мережа?
2. Що таке шар згортки? Його призначення?
3. Що таке шар пулінгу? Його призначення?
4. Що таке функція активації і навіщо вона потрібна?
5. Що таке BatchNormalization?
6. Що таке dropout?
7. Відомі архітектури ЗНМ та їхні особливості (згортка 1x1, residual block).
8. Що таке трансферне навчання?

# Комп’ютерний практикум № 7. Детекція об’єктів

**Тема:** Детекція об’єктів та сегментація зображень за допомогою згорткових нейронних мереж.

**Мета:** провести порівняльний аналіз різних архітектур згорткових нейронних мереж для задачі сегментації.

Хід виконання роботи

1. Інсталювати бібліотеку detectron2.
2. Завантажити попередньонавчену mask r-cnn.
3. Підготувати власний датасет згідно варіантом.
4. Донавчити модель та вивести на екран результати навчання за допомогою TensorBoard.
5. Порахувати метрики, час навчання та вивести на екран результати сегментації.
6. Виконати пункти 2-5, для детекції об’єктів з використанням чотирьох вибраних вами моделей (варіаціїї F-RCNN та Retina Net).

   Допоміжні посилання для виконання цього етапу:
   - https://detectron2.readthedocs.io/en/latest/
   - https://colab.research.google.com/drive/16jcaJoc6bCFAQ96jDe2HwtXj7BMD_-m5#scrollTo=tjbUIhSxUdm_
7. Виконати пункти 2-5 для Fully convolutional network.
8. Зробити висновки по роботі та відповісти на наступні питання:
   - Яка з мереж показала найкращий результат та чому?

Звіт має містити:

- опис створених мереж;
- результати експериментів зведені в таблиці;
- висновки по роботі;
- текст програми.

Короткі теоретичні відомості

Детекція об'єктів полягає у знаходженні екземплярів семантичних об’єктів певного класу - виділенні цих об’єктів обмежувальною рамкою з зазначення ймовірності з якою цей об’єкт належить до певного класу. Семантична сегментація полягає в тому що кожному пікселю співставляється мітка класу.

Для сегментації зображення за допомогою згорткових нейронних мереж найпершою була запропонована Fully convolutional network. ЇЇ умовно можна поділити на енкодер та декодер. Енкодер виділяє ознаки з зображення за допомогою шарів згортки та пулінгу - зменшуючи зображення. Декодер відтворює зображення з цих ознак збільшуючи карти ознак до оригінального розміру зображення за допомогою шарів upsampling та транспонованої згортки.

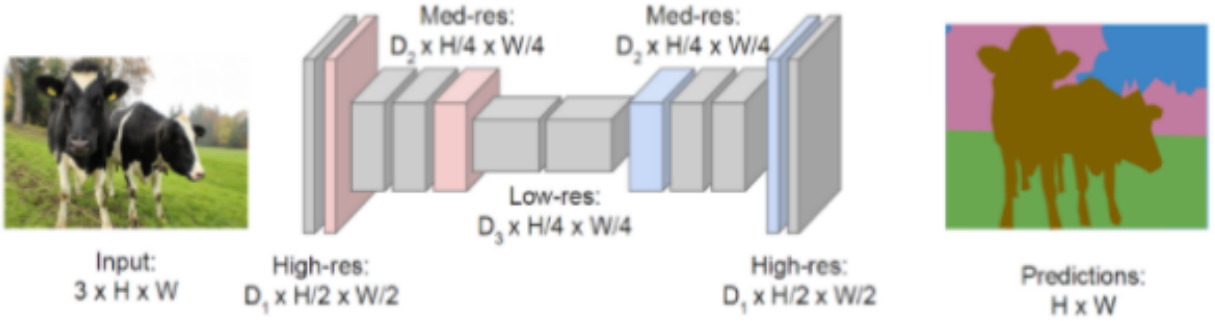



Рис. 7.1

Операція upsampling зворотна операція до пулінгу. Тобто маємо збільшити масштаб з низької роздільної здатності до початкової роздільної здатності вхідного зображення. Від одного значення до блоку значень. Це можна робити декількома способами:

- Метод найближчого сусіда. Вибираємо значення та заповнюємо сусідні комірки

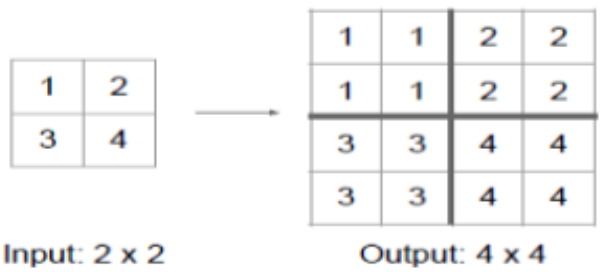


Рис.7.2

- Bed of Nails. Замість того, щоб заповнити всі сусідні комірки однаковими значеннями, заповнюємо значеннями заздалегідь визначені комірки, а решту – нулями.

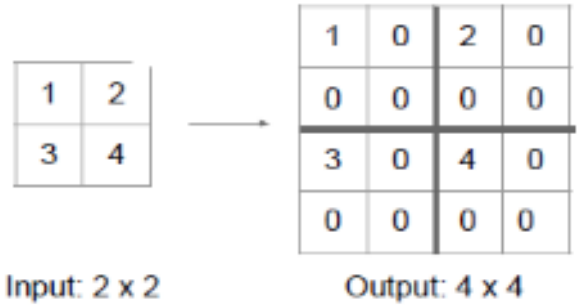



Рис. 7.3

- Запам'ятати розташування максимального елементу (для maxpooling), на те ж місце записати елемент, решта - нулі.

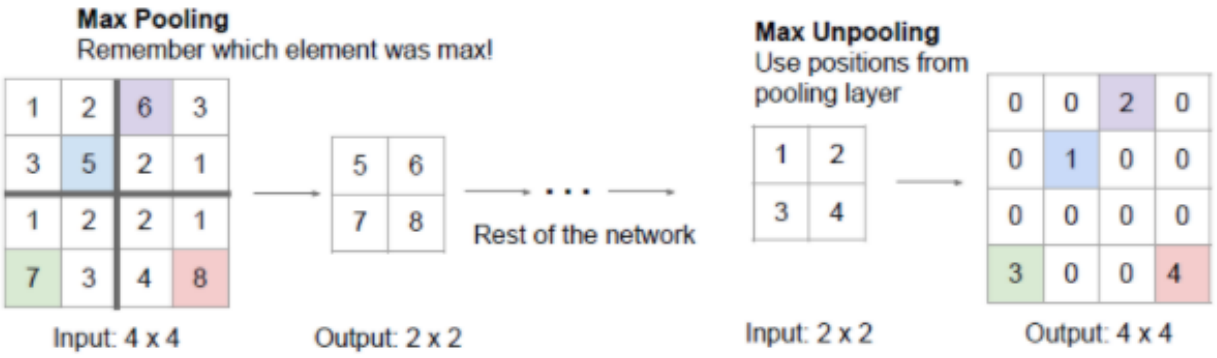



Рис.7.4

Транспонована згортка виконується наступним чином:

Нехай в нас 2x2 зворотня згортка з зсувом 1, padding 0. Беремо фільтр і перемножаємо його на перше значення зображення (карти ознак). Повторюємо процедуру для інших значень і сумуємо результат там де це треба.

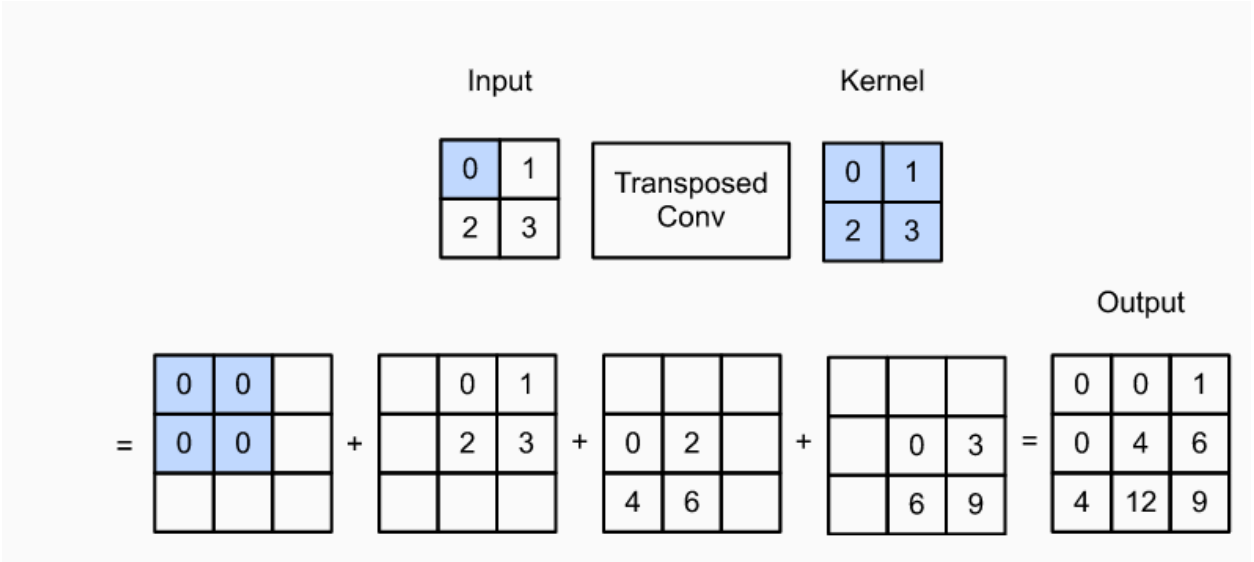



Рис. 7.5

В енкодері відбувається зменшення роздільної здатності вхідного сигналу, і навіть при використанні upsampling та транспонованої згортки дуже важко відтворити більш точні деталі навіть після використання складних методів, таких як Transpose Convolution. Тому керуючись тим що при заглибленні в мережу отримується більш загальні ознаки, при цьому втрачаємо інформацію про просторове розташування. Отже інформація з початкових (мілких) шарів містить більше просторової інформації, щоб її використати додаються пропускні з'єднання (skip connections). В кроці upsampling декодера ми сумуємо отримані на цьому шарі карти ознак з картами ознак відповідних ранніх шарів енкодера, таким чином додаючи точності отриманій карті сегментації.

R-CNN (Region Based Convolutional Neural Networks) належить до двоетапних методів детекції об'єктів, коли на одному етапі відбувається пошук частини зображення з можливим розташуванням об'єкта, а на другому етапі відбувається класифікація об'єкта та уточнення його розташування.

В R-CNN наявний модуль пропозиції регіонів, де генеруються регіони (частини зображення), де можливо присутній об'єкт. Другий модуль застосовує згорткову нейронну мережу до регіонів, щоб отримати карти ознак. В третьому модулі результуючі карти ознак подаються на

класифікатор - лінійну машину опорних векторів для визначення класу об’єкта та bounding box regressor, де відбувається уточнення обмежувальної рамка, що виділяє об’єкт і яка була отримана на першому етапі.

## Контрольні завдання

1. Реалізувати U-Net та застосувати для сегментації зображень.
2. Використати DeepLab для сегментації зображення для власного датасету.

## Контрольні запитання

1. Яка різниця між детекцією об’єктів і сегментацією зображення?
2. Як працює транспонована згортка?
3. Як працює зворотня операція до пулінгу?
4. Що дає процедура Upsampling?
5. Що таке двоетапні методи детекції об’єктів?
6. Які основні складові R-CNN?

# Список використаної літератури


1. Литвин В. В. Методи та засоби інженерії даних та знань/ В.В. Литвин , Ю.В.Нікольський, В.В. Пасічник - Львів: Магнолія 2006, 2021.- 252 с.
2. Clustering [Електронний ресурс]. — Режим доступу: https://scikit-learn.org/stable/modules/clustering.html/ (дата звернення: 05.10.2022). — Назва з екрана.
3. Programming Computer Vision with Python [Електронний ресурс]. — Режим доступу: http://programmingcomputervision.com/downloads/ProgrammingComputerVision_CCdraft.pdf / (дата звернення: 05.10.2022). — Назва з екрана.
4. Dive into Deep Learning [Електронний ресурс]. — Режим доступу: https://d2l.ai/index.html# / (дата звернення: 05.10.2022). — Назва з екрана.