УДК 009.4

АЛГОРИТМИ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ НА ЗОБРАЖЕННІ

Б. М. Гавриш[1], О. В. Тимченко[2, 3], М. П. Кляп[4]

[1]Національний університет «Львівська політехніка», вул. С. Бандери, 12, Львів, 79013, Україна

[2]Українська академія друкарства, вул. Під Голоском, 19, Львів, 79020, Україна

[3]Uniwersytet Warmińsko-Mazurski w Olsztynie, ul. M. Oczapowskiego, 2, Olsztyn, 10-719, Polska

[4]Ужгородський національний університет, пл. Народна, 3, Ужгород, 88000, Україна

Розглянуто алгоритми розпізнавання об'єктів на зображенні, проведено аналіз методів, що застосовуються під час обробки зображень для розпізнавання, а також описано використання засобів машинного навчання для поліпшення розпізнавання в межах роботи із зображеннями. Вибір конкретних методів зумовлений особливостями об'єкта, який потрібно розпізнати. Для вирішення такого завдання необхідно сформулювати властивості потрібного об'єкта і створити стійкий метод для виявлення об'єктів, що відповідають заданим параметрам. Також важливою є швидкість знаходження області зображення, де є потрібний об'єкт. Це можливо тільки за умови класифікації великої кількості елементів під час опрацювання кожного зображення.

Ключові слова: *розпізнавання образів, обробка зображень, комп'ютерний зір, машинне навчання.*

Постановка проблеми. Існує велика кількість методів розпізнавання об'єктів на зображенні. Деколи завдання розпізнавання ставиться неформальним чином — властивості шуканого об'єкта задаються без строгих математичних параметрів. Головна складність пошуку полягає в тому, що описати всі властивості об'єкта практично неможливо і ці властивості можуть відповідати не всім об'єктам. Тому в процесі математичної формалізації допускаються спрощення, які в результаті знижують якість алгоритму пошуку і знижують точність.

Тому під час вирішення задачі розпізнавання необхідно знайти оптимальне співвідношення складності обчислень і бажаної точності розпізнавання.

Аналіз останніх досліджень та публікацій. Головними критеріями якості ознак для вирішення широкого кола завдань, пов'язаних із розпізнаванням, враховуючи зорові образи, є роздільні властивості ознак, а також складність їх отримання.

Аналіз наявних алгоритмів розпізнавання об'єктів та методів виявлення об'єктів показав, що основною проблемою є низька стійкість до впливу зовнішніх умов, що ускладнюють якість розпізнавання: рівень освітлення, якість зображення, нахили зображення. Водночас алгоритми, стійкі до зовнішніх умов (змінюваний задній план, змінення освітлення, рухомі тіні тощо), часто виявляються вимогливими до апаратних обчислювальних ресурсів, що ускладнює застосування їх в системах, які працюють в реальному часі. Для вирішення поставленого завдання необхідно знайти, узагальнити і сформулювати в математичних термінах емпіричні спостереження, тобто формалізувати параметри шуканого об'єкта.

Мета статті — порівняти сімейства алгоритмів на основі AdaBoost, перетворення Хафа, методу Віоли-Джонса для розпізнавання об'єктів на зображенні за критеріями точності та швидкості, провести оцінку алгоритмів для можливості застосування машинного навчання.

Виклад основного матеріалу дослідження. *Примітиви Хаара.* Для вирішення завдань, пов'язаних з розпізнаванням, зручно використовувати прості алгоритми отримання ознак, наприклад використання алгоритмів розпізнавання на основі примітивів Хаара. Примітиви Хаара є результатом порівняння яскравості у двох прямокутних областях (рис. 1).

а б

Рис. 1. Ознаки, які використовуються для розпізнавання об'єктів:
а) — області, які не перетинаються; б) — області, які перетинаються

Ознаку для цієї області (відгук) на наведену властивість можна обчислити, використовуючи вираз (1) [1, 2]:

$$R = \frac{Sб}{Nб} - \frac{Sч}{Nч} \tag{1}$$

для областей, які не перетинаються (рис. 1 а), і

$$R = \frac{Sб}{Nб} - \frac{Sч - Sч \cap б}{Nч - Nч \cap б} \tag{2}$$

для областей, які перетинаються (рис. 1 б).

Чорна та біла області позначається «*ч*» і «*б*» відповідно.

ч∩*б* — область перетину; *S* — сума яскравостей, які знаходяться під областю пікселів зображення; *N* — кількість пікселів, що знаходяться в цій же області.

Таким способом можна отримати відгуки, які означають різницю середніх яскравостей пікселів, що знаходяться в зображенні, яке опрацьовується, котрі знаходяться під білою і чорною частинами зображення (рис. 1 а, б).

Перевага відгуків подібної властивості в тому, що вони не залежать від масштабування зображення, а також зміщення за шкалою яскравості. Крім вищенаведених формул, для обчислення відгуків на зазначену область зображення можна використовувати такі формули:
для областей, які не перетинаються:

$$R = Sб - Sч; \tag{3}$$

для областей, які перетинаються:

$$R = Sб - \left(Sч - Sч \cap б\right); \tag{4}$$

для областей, які не перетинаються:

$$R = \begin{cases} 1, \dfrac{Sб}{Nб} > \dfrac{Sч}{Nч} \\ -1, \dfrac{Sб}{Nб} \le \dfrac{Sч}{Nч} \end{cases}; \tag{5}$$

для областей, які перетинаються:

$$R = \begin{cases} 1, \dfrac{Sб}{Nб} > \dfrac{Sч - Sч \cap б}{Nч - Nч \cap б} \\ 1, \dfrac{Sб}{Nб} \le \dfrac{Sч - Sч \cap б}{Nч - Nч \cap б} \end{cases}. \tag{6}$$

Використання формул (5), (6) для обчислення значень відгуків з позиції інваріантності значень властивостей стосовно можливих яскравісних перетворень зображень є найбільш доцільним.

Використовуючи ці вирази, отримуємо значення властивості, які будуть симетричними до будь-яких можливих лінійних перетворень (монотонно зростаючими) яскравості, за умови, що перетворення не змінять приналежність до класу, але допускається значна зміна яскравості розподілу.

До подібних перетворень належить зміна контрасту і яскравості, що застосовуються під час обробки фото та відео в багатьох пристроях для підвищення якості зображення.

Алгоритми Adaptive Boosting

Існує підхід до вирішення завдань розпізнавання (класифікації) як посилення простих класифікаторів.

Підхід заснований на поєднанні декількох простих класифікаторів в один сильніший. Ефективність класифікатора — це сила або якість вирішення поставленого завдання класифікації.

Проаналізуємо сімейство алгоритмів, в основі яких лежить алгоритм AdaBoost (англ. «adaptive» — адаптивність і «Boosting» — посилення), який описали в 1996 р. Йоав Фройнд та Роберт Шапіра [11]. Цей алгоритм успішно застосовували для вирішення широкого кола завдань, пов'язаних з розпізнаванням на зображенні, а також для пошуку і розпізнавання осіб. Цей підхід досі успішно використовується в багатьох галузях, а також продовжуються як прикладні, так і теоретичні дослідження алгоритмів AdaBoost.

В основі методів розпізнавання, заснованих на посиленні простих класифікаторів, лежить така ідея [6]: поєднати кілька простих (елементарних) ознак для того, щоб одержати більш потужну комбіновану ознаку.

Розглянемо приклад з автомобільними гонками: нехай людина, яка знайома зі світом автоспорту вирішить розробити програму, що допомагає визначати переможця і пророкує шанси на перемогу різних спортсменів.

У результаті перегляду деякої кількості змагань і провівши опитування глядачів, які роблять ставки, ця людина виділила кілька емпіричних правил:

– потрібно ставити на машину, яка перемогла в чотирьох попередніх колах;
– потрібно ставити на машину, на яку зроблено максимальну кількість ставок;
– фаворитами завжди є чемпіони попередніх гонок тощо.

Зрозуміло, що будь-яке з перерахованих вище правил ненадійне і використовувати ці правила окремо недоцільно. Тому виникає необхідність оптимально скомпонувати ці правила для досягнення надійного результату.

Набір алгоритмів, які працюють з використанням методу посилення простих класифікаторів, дає змогу зібрати прості ознаки так, щоб отримати одну, але потужнішу ознаку.

Розглянемо один з перших алгоритмів з цієї групи — AdaBoost. На основі цього алгоритму була побудована на сьогодні найбільш ефективна за якістю розпізнавання і швидкості виконання поставленого завдання система розпізнавання об'єктів на зображенні — Viola-Jones ObjectDetector [10]. Серед основних переваг методів AdaBoost можна виділити:

– високу швидкість роботи;
– високу ефективність (точність);
– простоту реалізації [5].

Нехай потрібно створити функцію класифікації $F : X \to Y$, де X — простір векторів ознак, Y — простір міток класів. Нехай є навчальна вибірка $(x_1, y_1, ..., x_N, y_N)$, де $x_i \in X$ — вектор ознак, а $y_i \in Y$ — мітка класу, до якого належить $x_i \in X$. Розглянемо задачу з двома класами, тобто $Y = \{-1; +1\}$. Також існує сімейство простих функцій класифікації H: $X \to Y$. Побудуємо підсумковий класифікатор у такому вигляді:

$$F(x) = sign\left[\sum_{m=0}^{M} \alpha_m h_m(x)\right], \tag{7}$$

де $\alpha_m \in R$, $h_m \in H$. Створимо ітеративний процес, в який буде додаватись нове значення на кожному кроці

$$f_m = \alpha_m h_m(x). \tag{8}$$

Обчислення проведемо з урахуванням вже створеної частини нашого класифікатора.

1. Нехай $(x_1, y_1, ..., x_N, y_N)$ навчальна вибірка, а $D_0(i) = \frac{1}{N}$.

2. Для кожного кроку m = 1,2,…,M :

а) Обираємо кращий слабкий класифікатор $h_m(x) \in H$, а розподіл

$$D_m(i) h_m = \arg\min\left(e_m = \sum_{i=1}^{N} D_m(i) \cdot \left(h_m(x_i) \neq y_i\right)\right); \tag{9}$$

b) Обчислимо коефіцієнт

$$\alpha_m = \log\frac{1}{2}\left(\frac{1-e_m}{e_m}\right); \tag{10}$$

c) Зберігаємо $f_m = \alpha_m h_m(x)$ та оновлюємо розподіл

$$D_{m+1}(i) = \frac{D_m(i)\exp\left(-y_i f_m(x_i)\right)}{Z_i}. \tag{11}$$

3. Створюємо сильний класифікатор у такий спосіб:

$$F(x) = sign\left[\sum_{i=1}^{M} f_m(x)\right]. \tag{12}$$

Для кожного кроку прикладу (x_i, y_i) навчальної вибірки обчислимо його вагу: нехай $D_0(i) = \frac{1}{N}$, тоді

$$D_{m+1}(i) = \frac{D_m(i)\exp\left(-y_i f_m(x_i)\right)}{Z_i}, \tag{13}$$

де Z_i — коефіцієнт нормалізації, причому

$$\sum_{i=1}^{N} D_{m+1}(i) = 1. \tag{14}$$

Після присвоєння значення ваги кожного елементу навчальної вибірки задається рівень важливості саме цього прикладу для кожного кроку навчального алгоритму. Залежно від ваги елементу алгоритм докладає різні «зусилля»: що більша вага, то більше алгоритм намагається класифікувати цей приклад як правильний.

Аналізуючи формули (7)–(14), стає очевидно, що вага прикладу залежить від впевненості розпізнавання на попередніх кроках, тобто меншу вагу отримують приклади, які класифіковані правильно, а більшу ті, котрі неправильно.

По суті, ваги варіюються таким чином, щоб класифікатор, який використовується на цьому етапі, зосереджувався на прикладах, які на попередніх етапах не дали хорошого результату. Тобто на кожному етапі алгоритм працює з тією частиною даних, яку алгоритм на попередніх кроках погано класифікував. У підсумку з'єднуються результати проміжних етапів.

Наступний класифікатор оберемо, враховуючи зважену з розподілом D_m помилку. Візьмемо (будемо тренувати) $h_m \in H$, котрий мінімізує зважену помилку класифікації

$$e_m = \sum_{i=1}^{N} D_m(i) \cdot \left(h_m(x_i) \neq y_i\right). \tag{15}$$

Зазначимо, що D_m розглядаємо як розподіл ймовірності над X, що є правильним, оскільки

$$\sum_{i=1}^{N} D_{m+1}(i) = 1 \tag{16}$$

$$e_m = \Pr_{x \sim D_m}\left[h(x) \neq y_i\right]. \tag{17}$$

Тепер обчислюємо внесок доданка функції класифікації на поточному етапі

$$a_m = \log\left(\frac{1-e_m}{e_m}\right). \tag{18}$$

Процес триває до наперед визначеного кроку M, який задається самостійно.

Прості класифікатори

Розглянемо основу всіх методів посилення простих класифікаторів — прості класифікатори типу $H : X \to Y$. Наприклад, нехай дані, які подаються на вхід, — це n-мірні вектори $X = R^n$, тоді

$$H = \left\{ h^{\Theta,k} \left(x = \left(x_1, ..., x_k, ..., x_n \right) \right) \right\} = \begin{Bmatrix} 1, x_k > \Theta \\ -1, x_k < \Theta \end{Bmatrix} \quad (19)$$

це поріг по k-тій координаті. Незважаючи на свою простоту, цей класифікатор, посилений алгоритмом AdaBoost, дає вражаючі результати [4]. Система пошуку об'єктів на зображенні Viola-Jones знаходить 95 % всіх розшукуваних об'єктів із 0,001 % помилкових результатів.

Основна властивість, якою повинен володіти простий класифікатор, — ймовірність помилки має бути меншою за 1/2. Тобто потрібно, щоб він працював точніше ніж рівноймовірний:

$$\exists \gamma > 0 : \Pr_{x \sim D_m} \left[h(x) \neq y \right] \leq \frac{1}{2} - \gamma. \quad (20)$$

Механізм AdaBoost

Механізм AdaBoost виконує дві основні задачі:

– відбирає прості класифікатори (прості ознаки);
– комбінує відібрані класифікатори (ознаки).

Перше завдання вирішується відображенням простору вхідних векторів у простір значень простих класифікаторів:

$$x = \left(x_1, x_2, ..., x_N \right) \to \left(h_1(x), h_2(x), ..., h_M(x) \right). \quad (21)$$

Прості класифікатори з'єднуються лінійно, тобто шляхом складання лінійної комбінації, а прийняття рішення залежить від знака цієї комбінації.

Отже, простір значень простих класифікаторів розділяється гіперплощиною, і рішення приймається залежно від того, з якого боку цієї гіперплощини знаходиться відображення вектора ознак. Тобто підсумковий класифікатор спочатку створює відображення в певний простір, де проводиться лінійна класифікація зазвичай з розрядністю більшою, ніж вихідна.

Під час тренування алгоритм поетапно відображає і створює гіперплощину.

Існує інтерпретація сімейства алгоритмів на основі AdaBoost, яка спирається на поняття граней. Для алгоритму AdaBoost поняття «грані» визначається так:

$$\mu(x, y) = \frac{\sum_{m=1}^{M} y \cdot f_m(x)}{\sum_{v=1}^{M} \alpha_m}. \quad (22)$$

Отримане значення цієї величини можна прийняти за степінь «впевненості» класифікатора у прикладі (x,y). У випадку правильної класифікації грань більша від нуля, в протилежному разі — від'ємна. Розмір грані залежить від кількості класифікаторів, які правильно класифікують приклад, що більше їх — то більша грань [8].

З огляду на такий спосіб, яким обчислюється вага кожного прикладу, помітно, що на кожному етапі алгоритм намагається максимально збільшити найменшу

грань навчальної вибірки. Вважається, що такий підхід сприятливо впливає на узагальнюючі здатності алгоритму [3].

Сьогодні методи розпізнавання, засновані на посиленні простих класифікаторів, є популярними та досить ефективними методами класифікації. З огляду на високу швидкість роботи алгоритму, простоту його реалізації і високий ступінь ефективності ці методи отримали широке застосування в галузях, пов'язаних з розпізнаванням образів (в медицині, під час аналізу тексту, зображень тощо).

Метод Віоли-Джонса

У класичному методі Віоли-Джонса застосовуються ознаки прямокутної форми, такі як зображені на рис. 2.

У доповненому методі Віоли-Джонса, який використовується в бібліотеках OpenCV, застосовуються ознаки, доповнені примітивами [5], зображеними на рис. 3.

Для обчислення значення ознаки подібного типу застосовується формула:

$$F = X - Y, \tag{23}$$

де X — сумарне значення яскравості точок, закритих світлою частиною ознаки, а Y — сумарне значення яскравості точок, закритих темною частиною ознаки. Примітиви Хаара дають точкове значення перепаду яскравості по осі X і Y відповідно [9].

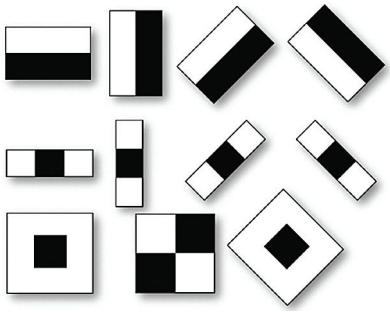

Рис. 2. Примітиви Хаара

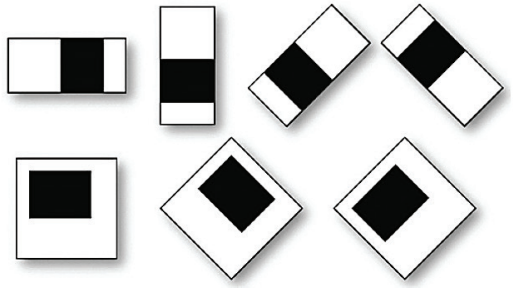

Рис. 3. Примітиви Хаара, які використовуються в розширеному методі

Алгоритм Хафа

В основі алгоритму Хафа лежить метод виявлення ліній на зображенні. Використання цього методу дає можливість задавати параметри пошуку ліній або сімейства кривих і знаходити на зображенні множину кривих цього сімейства.

Припустимо, існують точки в просторі R^m:

$$X = \{x_1, ..., x_n\}. \tag{24}$$

Сімейство кривих задане параметрично:

$$F = (\varphi, x) = 0, \tag{25}$$

де F — певна функція; φ — фактор параметрів сімейства кривих; x — координати точок простору R^m. Значення φ визначає кожну криву, а множина значень φ — утворює простір усіх кривих заданого сімейства.

У теорії цей метод можна застосовувати не тільки на площині, а й в *n*-вимірному просторі. Але на практиці використовується здебільшого пошук у 2-вимірному просторі, оскільки складність алгоритму достатньо висока [7].

Стандартний алгоритм складається із таких кроків:

– вибір кроку дискретності;
– заповнення матриці;
– аналіз матриці;
– виділення кривих.

На першому кроці здійснюється вибір сітки дискретності. При виборі сітки важливо знайти оптимальний розмір, оскільки у випадку занадто великої сітки є ймовірність попадання точок, які лежать на різних кривих. Якщо обрати занадто дрібну, то існує ймовірність розмиття максимумом шуканої кривої, оскільки точки однієї кривої можуть потрапити в різні комірки.

Другий крок є найбільш трудомістким етапом в алгоритмі, складність якого залежить від просторової розмірності та частоти дискретності. Кількість комірок зростає зі збільшенням розмірності і зменшенням сітки.

Існує декілька модифікацій стандартного алгоритму Хафа [2].

Алгоритм порівняння зі шаблоном

Основний принцип цього методу полягає в порівнянні зображення, яке опрацьовується, з наявним шаблоном-зображенням. Завдання алгоритму полягає в пошуку областей, які збігаються із шаблоном. Для роботи алгоритму необхідно мати вихідне зображення і шаблон. Процес відбувається у такий спосіб:

– зображення, яке опрацьовується, накладається на шаблон;
– зображення переміщається по шаблону в пошуках збігів;
– з кожною зміною положення зображення обчислюється метрика, що відображає збіг.

Метрика записується в підсумкову матрицю R для кожного положення у вигляді (x,y). Після закінчення порівняння збіги знаходяться в глобальних максимумах або мінімумах, залежно від обраного методу. Існують методи порівняння із шаблоном, які використовують методи найменших квадратів, кореляції тощо.

Алгоритм порівняння із шаблоном на основі методу найменших квадратів

Нехай $R(x,y)$ — результуюча матриця.

$$R(x,y)=\sum_{x,y}\left(T\left(x',y'\right)-I\left(x+x',y+y'\right)\right)^2, \tag{26}$$

де x', y' — поточні координати шаблону.

$x'=0\ldots w\text{-}1$; $y'=0\ldots h\text{-}1$, де w — ширина, а h — висота шаблону.

Цей метод є найпростішим алгоритмом розпізнавання зображення, але має низьку точність.

Сегментація зображення

Сегментація зображень відіграє важливу роль у системах комп'ютерного зору для вирішення завдань, пов'язаних із розпізнаванням сцен і виділення (визначення) об'єктів.

Існують певні виставлені вимоги до оброблюваних областей:

– області повинні бути однорідними щодо заданих характеристик, тобто внутрішня частина області має бути простою, не мати великої кількості отворів, що не належать до цієї області;
– суміжні області повинні мати суттєві відмінності за характеристиками, щодо яких вони вважаються однорідними;
– межі між сегментами повинні бути достатньо виразними.

Сегментація є важливою процедурою, оскільки результати виконання цієї процедури значно впливають на подальший процес аналізу зображення.

Існують такі алгоритми сегментації:

– порогове оброблення;
– класифікація пікселів;
– сегментація колірних зображень;
– виділення країв.

Порогове оброблення

Існують завдання, пов'язані з перетворенням півтонового зображення в бінарне. Подібні перетворення часто використовуються для скорочення інформації про зображення. Результатом може бути виділення контурів шуканих об'єктів і виключення фону.

Суть порогового опрацювання полягає у поділі об'єктів зображення на два типи за ознакою яскравості. При пороговому опрацюванні задається певне значення порогу, щодо якого відбувається поділ.

Класифікації пікселів

Для класифікації пікселів використовується оптичний потік. Оптичний потік — технологія, що використовується в різних областях комп'ютерного зору для визначення зрушень, сегментації, виділення об'єктів, стиснення відео. Оптичний потік — зображення видимого руху, який є зсувом кожної точки між двома зображеннями.

Оптичний потік використовується під час визначення руху, також існують методи, що обчислюють рух між двома кадрами, узятими в момент часу t і t1, в кожному пікселі. Алгоритми на його основі застосовуються під час вирішення завдань, пов'язаних з кодуванням рухів і розробленням систем стереозору.

За допомогою алгоритмів, що використовують оптичний потік, можна визначити не тільки рух об'єктів, а й створити тривимірну структуру сцени.

Сегментація кольорових зображень

Процес сегментації кольорового зображення полягає у виділенні на зображенні пов'язаних областей за критеріями однорідності, заснованими на ознаках, які обчислюються із значень декількох колірних компонентів, залежно від обраної колірної моделі.

Область можна описати як пов'язану підмножину пікселів, задану в просторі кольорів.

Алгоритми, що використовують таке подання, називаються піксельними алгоритмами сегментації. Існує сімейство алгоритмів, що використовують поняття порогового відсікання для опрацювання кольорових зображень. Так само існують алгоритми, засновані на кластеризації простору кольорів. Пікселі збираються в групи, ґрунтуючись на функціях присвоєння, і потім розбиваються на пов'язані області. Областю у цьому випадку є чітка множина пікселів, які задовольняють заданий критерій однорідності.

Висновки. Здійснено аналіз алгоритмів, застосовуваних під час опрацювання зображень. Алгоритм Віоли-Джонса найкращий, оскільки існує можливість навчання. Використовуючи цей алгоритм в системах виявлення об'єктів, можна змінювати характеристики шуканого об'єкта, проводячи навчання на конкретних прикладах. У поєднанні з алгоритмом AdaBoost цей підхід забезпечує відповідну ефективність виявлення об'єктів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Limin Xia. Vehicle Recognition Using Boosting Neural Network Classifiers. 2016 6th World Congress on Intelligent Control and Automation. Dalian, 2006. Pp. 9641–9644. doi: 10.1109/WCICA.2016.1713873.
2. Дурняк Б. В., Медиковський М. О., Тимченко О. В. Інформаційні технології пошуку та зберігання даних про графічні об'єкти на основі їх семантики. Львів : Українська академія друкарства, 2020. 220 с.
3. Hsu Y., Ciou Y., Perng J. Object recognition system design in regions of interest based on AdaBoost algorithm. 2017 20th International Conference on Information Fusion (Fusion). Xi'an, 2017. Pp. 1–5. doi: 10.23919/ICIF.2017.8009787.
4. Tsai P. Y., Hsu Y., Chiu C., Chu T. Accelerating AdaBoost algorithm using GPU for multi-object recognition. 2015 IEEE International Symposium on Circuits and Systems (ISCAS). Lisbon, 2017. Pp. 738–741. doi: 10.1109/ISCAS.2015.7168739.
5. Ge J., Lu D., Fang Y. A revised training mechanism for AdaBoost algorithm. 2010 IEEE International Conference on Software Engineering and Service Sciences. Beijing, 2020. Pp. 491–494. doi: 10.1109/ICSESS.2010.5552322.
6. Zhang P., Yang Z. A Novel AdaBoost Framework With Robust Threshold and Structural Optimization. in IEEE Transactions on Cybernetics. Jan. 2018. Vol. 48. No. 1. Pp. 64–76. doi: 10.1109/TCYB.2016.2623900.
7. Ren J., Kehtarnavaz N., Estevez L. Real-time optimization of Viola -Jones face detection for mobile platforms. 2018 IEEE Dallas Circuits and Systems Workshop: System-on-Chip - Design, Applications, Integration, and Software. Dallas, TX, 20108. Pp. 1–4. doi: 10.1109/DCAS.2008.4695921.
8. Masek J., Burget R., Uher V., Güney S. Speeding up Viola-Jones algorithm using multi-Core GPU implementation. 2018 36th International Conference on Telecommunications and Signal Processing (TSP). Rome, 2018. Pp. 808–812. doi: 10.1109/TSP.2013.6614050.
9. Lienhart R., Kuranov A., Pisarevsky V. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. MRL Technical Report, 2018.

10. Rosset S., Zhu Ji., Hastie T. Boosting as a Regularized Path to a Maximum Margin Classifier. Journal of Machine Learning Research. 2004. 5. 941–973.
11. Yoav Freund, Robert E. Schapire. A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence. September, 2019. 14 (5):771–780.

REFERENCES

1. Limin, Xia. (2006). Vehicle Recognition Using Boosting Neural Network Classifiers. 2016 6th World Congress on Intelligent Control and Automation. Dalian, 9641–9644. doi: 10.1109/WCICA.2016.1713873 (in English).
2. Durniak, B. V., Medykovskyi, M. O., & Tymchenko, O. V. (2020). Informatsiini tekhnologii poshuku ta zberihannia pro hrafichni yikh stmantyky. Lviv : Ukrainska akademiia drukarstva (in Ukrainian).
3. Hsu, Y., Ciou, Y., & Perng, J. (2017). Object recognition system design in regions of interest based on AdaBoost algorithm. 2017 20th International Conference on Information Fusion (Fusion). Xi'an, 1–5. doi: 10.23919/ICIF.2017.8009787 (in English).
4. Tsai, P. Y., Hsu, Y., Chiu, C., & Chu, T. (2017). Accelerating AdaBoost algorithm using GPU for multi-object recognition. 2015 IEEE International Symposium on Circuits and Systems (ISCAS). Lisbon, 738–741. doi: 10.1109/ISCAS.2015.7168739 (in English).
5. Ge, J., Lu, D., & Fang, Y. (2020). A revised training mechanism for AdaBoost algorithm. 2010 IEEE International Conference on Software Engineering and Service Sciences. Beijing, 491–494. doi: 10.1109/ICSESS.2010.5552322 (in English).
6. Zhang, P., & Yang, Z. (Jan. 2018). A Novel AdaBoost Framework With Robust Threshold and Structural Optimization. in IEEE Transactions on Cybernetics, 48, 1, 64–76. doi: 10.1109/TCYB.2016.2623900 (in English).
7. Ren, J., Kehtarnavaz, N., & Estevez, L. Real-time optimization of Viola -Jones face detection for mobile platforms. 2018 IEEE Dallas Circuits and Systems Workshop: System-on-Chip - Design, Applications, Integration, and Software. Dallas, TX, 20108, 1–4. doi: 10.1109/DCAS.2008.4695921 (in English).
8. Masek, J., Burget, R., Uher, V., & Güney, S. (2018). Speeding up Viola-Jones algorithm using multi-Core GPU implementation. 2018 36th International Conference on Telecommunications and Signal Processing (TSP). Rome, 808–812. doi: 10.1109/TSP.2013.6614050 (in English).
9. Lienhart, R., Kuranov, A., & Pisarevsky, V. (2018). Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. MRL Technical Report (in English).
10. Rosset, S., Zhu, Ji., & Hastie, T. (2004). Boosting as a Regularized Path to a Maximum Margin Classifier. Journal of Machine Learning Research, 5, 941–973 (in English).
11. Yoav, Freund, & Robert, E. (September, 2019). Schapire. A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence, 14 (5):771–780 (in English).

doi: 10.32403/0554-4866-2022-1-83-47-58

OBJECT RECOGNITION ALGORITHMS

B. M. Havrysh[1], O. V. Tymchenko[2, 3], M. P. Klyp[4]

[1] Lviv Polytechnic National University,
12, S. Bandery St., Lviv, 79013, Ukraine

[2]Ukrainian Academy of Printing,
19, Pid Holoskom St., Lviv, 79020, Ukraine

[3]Uniwersytet Warmińsko-Mazurski w Olsztynie,
2, M. Oczapowskiego St., Olsztyn, 10-719, Polska

[4]Uzhhorod National University,
3, Narodna Square, Uzhhorod, 88000, Ukraine
dana.havrysh@gmail.com
olexandr.tymchenko@uwm.edu.pl

This article discusses the algorithms for recognizing objects in the image, analyses the methods used in image processing, and describes the use of machine learning tools in working with images. The selection of specific methods is determined by the characteristics of the object to be recognized. To solve this problem, it is necessary to formulate the properties of the desired object and create a stable method for identifying objects that meet the specified parameters. The importance of finding the area of the desired object as soon as possible is also taken into account. This is possible only if a large number of elements are classified during the processing of each image.

There are many methods for recognizing objects in an image. Sometimes the task of recognition is set informally – the properties of the desired object are set without strict mathematical parameters. The main difficulty is that it is almost impossible to describe all the properties and these properties may not correspond to all objects. Therefore, in the process of mathematical formalization, simplifications are allowed, which in turn reduce the quality of the algorithm and reduce the accuracy.

As a result, one can say that in solving the problem of recognition it is necessary to find the optimal ratio of computational complexity and the desired accuracy.

The main criteria for the quality of symbols to solve a wide range of problems related to recognition, taking into account visual images, are the separate properties of symbols, as well as the difficulty of obtaining them. Analysis of existing algorithms for object recognition and methods of object detection showed that the main problem is low resistance to external conditions that complicate the quality of recognition: light level, image quality, image tilt. At the same time, algorithms that are resistant to external conditions (changing backgrounds, changes in lighting, moving shadows, etc.) are often demanding on hardware computing resources, which complicates their application in systems that work in real time. To solve this problem, it is necessary to find, summarize and formulate empirical observations in mathematical terms. That is to formalize the parameters of the desired object.

Keywords: *pattern recognition, image processing, computer vision, machine learning.*

Стаття надійшла до редакції 11.04.2022.
Received 11.04.2022.