КІБЕРНЕТИКА та КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ

УДК 519.85 DOI:10.34229/2707-451X.21.4.6

К.П. СОСНЕНКО

ДВА ПІДХОДИ ДЛЯ РОЗПІЗНАВАННЯ СТРУКТУРИ БЛОК-СХЕМ

Розглядаються два варіанти підготовки навчальних і тестових прикладів для розпізнавання графічних зображень. Проведено зрівняльний аналіз нейромережевих та синтаксичних структурних підходів для рішення цієї задачі. Обґрунтовано доцільність поєднання синтаксичних та нейронних методів розпізнавання графічних зображень.

Ключові слова: *розпізнавання зображень, згорткові нейронні мережі, синтаксичний аналіз.*

Вступ. Робота з графічними зображеннями є важливим елементом практично будь-яких сучасних систем автоматизованого проектування.

Результатом нейромережевої, у тому числі глибокої обробки зображень, може бути розпізнаванням класів приналежності присутніх на них об'єктів. Об’єкти реального світу вимагають значних витрат на розробку і реалізацію вузькоспеціалізованих систем комп’ютерного зору. За останні роки спостерігається поліпшення якісних характеристик, отриманих в галузі технічного зору. Це стало можливим завдяки штучним нейронним мережам (ШНМ).

За останні кілька років вони замінили багато алгоритмів машинного навчання і комп'ютерного зору [1]. Базова модель нейронної мережі складається з нейронів, організованих у шари. Кожна нейронна мережа має вхідний і вихідний шар та кілька прихованих шарів, доданими до неї у залежності від складності проблеми. При передачі даних через шари, нейрони навчаються і розпізнають ознаки (створюється модель). Після того, як модель навчена, мережа робить на основі тестових даних прогнози [2].

Висока ефективність машин стала можливою завдяки особливому типу нейронної мережі, так званої згорткової нейронної мережі (ЗНМ). Сучасні фреймворки глибокого навчання, такі як Tensorflow і PyTorch, спрощують навчання машин зображень. ЗНМ це особливий тип нейронної мережі, який працює добре з зображеннями.

Проведено зрівняльний аналіз нейромережевих та синтаксичних структурних підходів для рішення цієї задачі, визначені графічні потреби до початкових документів і формати їх визначення.

Розроблений алгоритм синтаксичного аналізу і розпізнавання структури графічних зображень блок-схем (БС) ефективно реалізується магазинним програмним автоматом з двома стеками. В основі програмної реалізації отримано підтвердження доцільності застосування синтаксичних методів на рівні розпізнавання всього зображення БС і нейромережевих підходів на рівні елементів і вузлів її з’єднань [3].

Постановка завдання. Розпізнавання структури БС – необхідна складова для систем сучасного автоматизованого проектування. БС використовують у ряді випадків для наглядності при програмуванні складних процесів [4].

В цій статті розглядається розпізнавання плоского, чорно-білого зображення блок-схеми. Це двовимірні зображення або їх частини що виділені, які відображаються в довільному графічному форматі системними засобами на екрані монітора комп'ютера. Базові фігури БС: прямокутник, ромб, паралелограм, коло, еліпс (овал) і т. п. [5].

Обмеженнями на розпізнавання структури графічного зображення блок-схеми повинно бути: замкнутість опуклих контурів фігур, відсутність внутрішніх від них відводів і стрілок на контурних лініях; відсутність циклів у лініях зв'язку фігур і їх об'єднаннях; лінії зв'язку повинні бути вертикальними або горизонтальними, закінчуватися у вершинах ромбовидних фігур предикатів або на сторонах інших фігур і не бути прямолінійними продовженнями сторін цих фігур. Перераховані вище обмеження є типовими для зображення БС. Крім того, допускаються зв'язки між фігурами типу дерев, що сходяться і розходяться для відображення паралельних процесів.

Лінії відображають залежності управління між фігурами і можуть закінчуватись стрілками. Стрілки можна не вказувати при направленні лінії зліва направо і зверху вниз. Лінії повинні підходити до фігури зліва або зверху, а виходити знизу або праворуч. Хоча можуть бути і винятки.

Основна складність нейромережевого розпізнавання це підготовка навчальних і тестових прикладів та тривалий процес машинного навчання. Для ефективної роботи потрібен великий і різноманітний набір навчальних даних [6].

Алгоритм розпізнавання. Для розпізнавання фігур та зв'язків між ними, графічні зображення розглядаються як множина замкнутих контурів фігур і ліній зв'язків між фігурами.

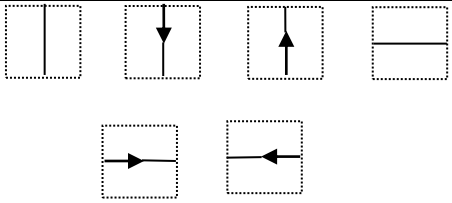
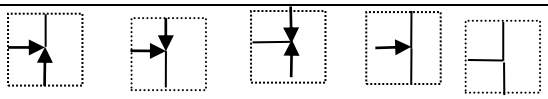
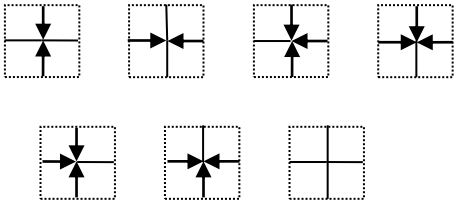
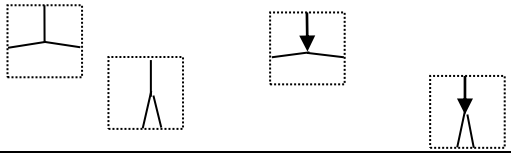
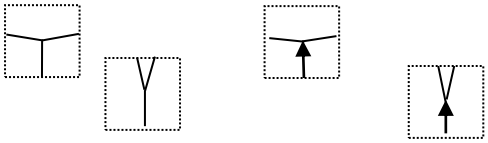
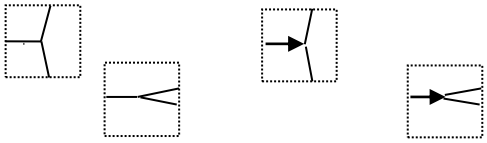
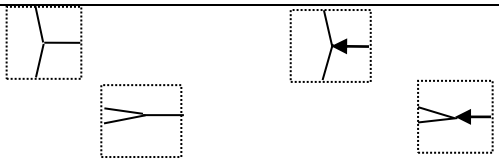
Алгоритми, пов'язані з розпізнаванням, зазвичай, функціонують шляхом збереження унікальних для об'єкта характеристик [7]. Приймаємо, що навчальні зображення однакові за розміром і центруються відносно відстані до рамки. Не потрібно обрізати або масштабувати зображення, перш ніж їх можна буде використовувати.

Щодо до автоматизованих систем обробки технічної документації, то аналіз структур блок-схем можливий на рівні відрізків ліній і їх з'єднань.

Зображення можна представити у вигляді масиву 28 х 28 пікселів. Вводим поняття «клас», що відповідає різновидності вузлів і представляє собою масив цілих чисел від 1 до 124. Для класифікації вузлів за типами: поворот лівий, правий, відведення і т. п. застосовується сигнатура. Сигнатурою вузла $z_q^{\rho} = i_0 i_1 ... i_{\rho}$, де ρ – кількість променів вузла, називається зростаюча послідовність індексів апроксимуючих промені зображення векторів, у якій замість постійного нульового значення початкового індексу $i_0 = 0$ зберігається його абсолютне значення $i_0 = \varphi_0 / \Delta\varphi$ щодо осі 0X. За виключенням початкового індексу сигнатура вузла інваріантна до його масштабу, повороту і зсуву [8]. У табл. 1 наведено різновидності вузлів, їх клас та сигнатури в блок-схемах.

В табл. 2 на прикладі вузла, який відображає вхід у ромб зверху, наведені варіанти зміщення центру рисунку БС, товщини ліній, кривизни, шуму.

ТАБЛИЦЯ 1. Різновидності вузлів БС

Клас	Тип вузла	Рисунки можливих вузлів	Сигнатура
1…4	1-но променевий		18 #18 0 #0
5…..10	2-х променевий		6, 18 #6, 18 6, #18 0, 12 0, #12 #0, 12
11….15	3-х променевий, відвід зліва*		6, #12, #18 #6, -#2,18 #6, 12, #18 6, #12,18 6,12,18
31…37	4-х променевий		0, #6,12, #18 #0, 6, #12,18 #0, #6,12, #18 #0, #6, #12,18 0, #6, #12, #18 #0, 6, #12, #18 0, 6,12,18
38…47	Вхід у ромб зверху**		6,13,23 #6,13,23 6,14,22 #6,14,22 6,15,21 #6,15,21 6,16,20 #6,16,20 6,17,19 #6,17,19
48…57	Вхід у ромб знизу		1,11,18 1,11,#18 2,10,18 2,10,#18 3, 9,18 3, 9,#18 4, 8,18 4, 8,#18 5, 7,18 5, 7,#18
58…67	Вхід у ромб зліва		5,12,19 5,#12,19 4,12,20 4,#12,20 3,12,21 3,#12,21 2,12,22 2,#12,22 1,12,23 1,#12,23
68…77	Вхід у ромб справа		0, 7,17 #, 7,17 0, 8,16 #0, 8,16 0, 9,15 #0, 9,15 0,10,14 #0,10,14 0,11,13 #0,11,13

Примітки. *Ще не показані 3 типи вузлів: 3-х променеві з відводом справа (клас: 16…20, сигнатури: #0,6, #18; #0,#6,18; 0, #6, #18; #0,6,18; 0,6,18); 3-х променеві з відводом знизу (клас: 21…25, сигнатури: #0,12, #18; #0, #12,18; 0,#12, #18; 0,12, #18; 0,12,18); 3-х променеві з відводом зверху (клас: 26…30, сигнатури: #0, #6,12; #0,6, #12; 0, #6, #12; 0, #6,12; 0,6,12).

**В кожному рядку входу в ромб не показані рисунки проміжних вузлів (сигнатури показані).

ТАБЛИЦЯ 2. Варіанти зміщення центру рисунку БС, товщини ліній, кривизни, шуму

Клас	Рисунок вузла	Кривизна	Зміщення центру (пікс)	Товщина ліній (пікс)	Шум (дисперсія діапазон)	Сигнатура
45		0	2	0.75	0	#6, 15, 21
44		2 %	0	0.75	0	#6, 14, 22
43		0	0	4.5	0	#6, 13, 23
44		0	0	0.75	1 – 3п	#6, 13, 23

Для реалізації підготовчих процесів у даній статті розглядається два варіанти

1. Двовимірні зображення Image (x, y) або їх частини виділяються квадратами 28*28 пікселів.

У даній статті наводиться результат роботи програми, розробленої для підготовки різноманітних можливих вузлів при створенні блок-схеми. Використовується Delphi – імперативна, структурована, об'єктно-орієнтована, високорівнева мова програмування [9]. Інтерфейс програми показано на рис. 1.

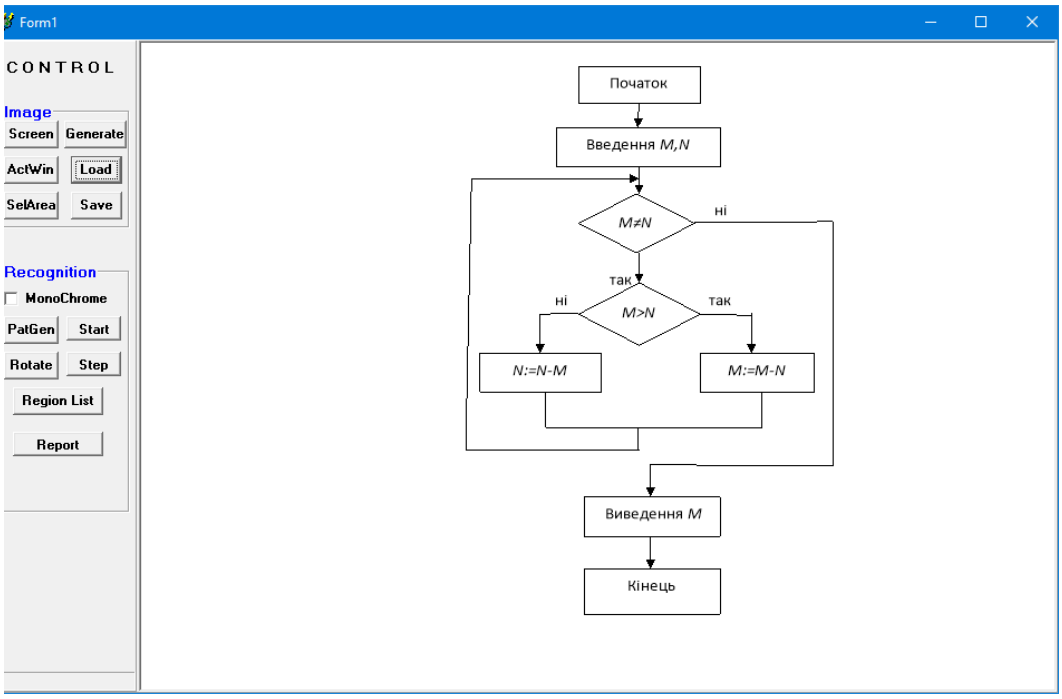


РИС. 1. Інтерфейс програми розпізнавання структури блок-схем

Послідовність синтаксичного розбору структури блок-схем

1. Графічне зображення БС – це орієнтований навантажений текстом граф з безліччю вершин і ребер. В силу вищеприйнятих обмежень, ребра графа утворюють замкнуті, можливо дугоподібні контури фігур *F* БС і деревовидні об'єднання *C* ліній зв'язку між контурами, вершини – вузли з'єднань ліній контурів і ліній зв'язків.

2. Просування сканера по ребрах і вузлах графа здійснюється за координатами чергових центрів кругового сканування можливих продовжень, поворотів і променів, отримуваних шляхом порівняння з заданим порогом δ локальної області БС з еталонними векторами *V* з центром в по-точній точці *O*(*x*, *y*). Якщо за напрямок вектора φ прийняти напрямок лінії, що сканується з центру обертання (зворотнє напрямку просування), то при $k = 24$ можливі такі варіанти кругових діаграм: однопроменеві, двохпроменеві, трьохпроменеві, чотирьохпроменеві, п'ятипроменеві.

3. Розпізнання елементів БС базується на послідовному скануванні проти годинникової стрілки ліній контурів фігур та ліній зав'язків – елементарних відрізків БС, порівняння їх з тестовими і подальші дії, які дозволяють отримати їх сигнатури. При цьому розпізнавання і локалізація об'єктів здійснюється в рамках переміщуваного вікна.

4. Сканування проти годинникової стрілки ліній контурів фігур дозволяє отримати координати вузлових точок υ_q, зміну максимальних індексів i_ρ променів, в яких у порівнянні з попереднім вузлом υ_{q-1} перевищується порогове значення $i_0^q + i_\rho^q - i_0^{q-1} > \Delta i$. При запису контуру деякої фігури у вигляді ланцюжка з індексів i_ρ його вузлів, починаючи з крайнього верхнього лівого вузла $\upsilon_q(x_{\min} \mid y_{\min})$, сигнатура що отримується, наприклад, ромба – 16201620 4 = *gZ*, інваріантна до масштабування і зсуву фігури, використовується для порівняння з еталонними значеннями або їх варіантами.

Реалізація та практичні результати

Основні процедури алгоритму розпізнавання зображень БС були частково апробовані на програмному рівні, що дозволяють зробити висновок про ефективність запропонованих структурних методів. Процес розпізнавання включає у себе наступні етапи.

1. Перенесення зображення *Image* (*x, y*) з файлу (Інтернету) або виділеної екранної області монітора на план – в масив *Img* (*x, y*) швидкої пам'яті для всієї подальшої прискореної його обробки.

2. Попередня обробка зображення, призначена для підвищення його якості перед виконанням операцій векторного сканування, може включати фільтрацію шумів, зміни колірного балансу, підвищення контрасту, бінаризацію й інші операції підкреслення і виділення контурів.

3. Автоматична генерація еталонних радіус-векторів (зі стрілками і без них) виконується за замовчуванням при активації програми розпізнавання, або після внесення змін до її установки з урахуванням характеристик зображення. Еталонні вектори включають відносні координати чергового кроку для прискорення просування по лініях БС.

4. Сканування ліній БС, поєднане з процесом його синтаксичного аналізу (виділення діаграм вузлів з'єднань, ідентифікація контурів фігур та об'єднань з'єднань) з метою скорочення числа додаткових проходів алгоритму розпізнавання.

5. Виділення фігур, зв'язків між ними, областей тексту, підготовка і зберігання файлу звіту.

Час розпізнавання типових зображень повнозв'язних графів БС розміром 1024 х 768 ріх на комп'ютері з двоядерним процесором з частотою 2.5 МГц орієнтовно становить до десятка секунд.

2. Другий спосіб. Створюємо свою нейромережу. Використовуємо PyTorch – науковий обчислювальний пакет на основі Python. Це також одна з кращих дослідницьких платформ глибокого навчання, створена для забезпечення максимальної гнучкості й швидкості. Він відомий тим, що забезпечує дві з найбільш високорівневих функцій, а саме: тензорні обчислення з сильною підтримкою прискорення GPU і побудова глибоких нейронних мереж [10].

PyTorch має простий інтерфейс, він пропонує простий у використанні API, через що простий в управлінні і працює як Python. PyTorch забезпечує відмінну платформу, яка пропонує динамічні обчислювальні графіки.

Бібліотека PyTorch спеціально розроблена, щоб бути інтуїтивно зрозумілою і простою у використанні. Коли виконується рядок коду, вона запускається, що дозволяє виконувати відстеження у реальному часі, створюючи моделі нейронних мереж.

На рис. 2 показано фрагмент роботи програми для створення набору зображень для підготовки тестів та навчання з використанням PyTorch (варіанти розмножуються).

Базу даних, створену таким методом, потрібно піддати випадковим викривленням. Для цього до пікселя можна додавати яке-небудь рандомне значення. Чим більше розкид цих значень, тим більший шум.

Щоб створити нейронну мережу в PyTorch, використовується клас nn.Module. Щоб ним скористатися, також необхідно спадкування, що дозволить використовувати всі функції базового класу nn.Module.

Вдалим у застосуванні для вирішення завдань комп’ютерного зору являються згорткові нейронні мережі (ЗНМ). Вони – це особливий тип нейронної мережі, який працює добре з зображеннями. Також ЗНМ використовують кореляції між суміжними входами на рисунках. Таким чином немає необхідності з'єднувати кожен вузол з кожним у наступному шарі. Такий прийом зменшує кількість вагових параметрів для тренування моделі.

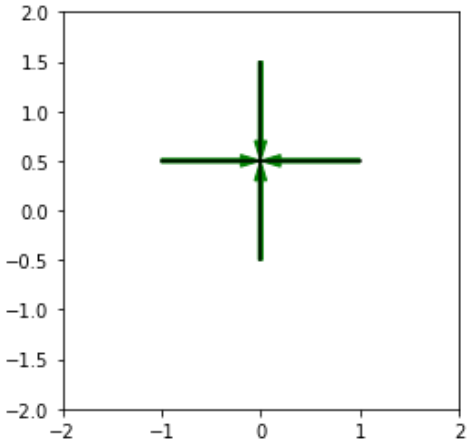


РИС. 2. Фрагмент роботи програми для підготовки тестів з використанням PyTorch

Для добування і «просіювання» інформації використовуються фільтри. На початку, низькорівневі функції можуть бути вилучені з урахуванням конкретного фільтру. Фільтром являється набір значень пікселів, аналогічно зображенню. Його можна розуміти як ваги, які з'єднують шари у згортковій нейронній мережі. Ці ваги або фільтри множаться на вхідні значення для отримання проміжних зображень, які представляють розуміння зображення комп'ютером. Потім вони множаться ще на кілька фільтрів, щоб розширити огляд.

Переваги згорткових мереж над повнозв'язаними полягають у використанні спільної ваги у згорткових шарах. Це означає, що для кожного пікселя шару використовується один і той же фільтр. В порівнянні з повнозв'язною мережею згортальні нейронні мережі дієво зменшують кількість параметрів для тренування. Ось кілька властивостей згорткових нейронних мереж, які суттєво зменшують кількість параметрів і ваг та прискорюють процес тренування у порівнянні з повнозв'язаними мережами: не кожен вузол у першому (вхідному) шарі з'єднаний з кожним вузлом у другому шарі; постійні параметри фільтру (при русі фільтру по зображенню однакові ваги застосовуються для кожного 2 x 2 набору вузлів). Кожен фільтр має певний набір ваг, які застосовуються для кожної операції згортки. Це зменшує кількість параметрів.

Шари, що використовуються в ЗНМ:

– згортковий шар;

– шар ПУЛІНГ;

– повнозв'язаний шар.

Послідовність дій виглядає наступним чином. У вхідних даних міститься вихідна інформація про зображення (в даному випадку 28 – ширина, 28 – висота). Згортковий шар перемножує значення фільтру на вихідні значення пікселів зображення (поелементне множення), після чого всі добутки додаються, тобто цей шар виконує згортку зображення. Його гіперпараметри включають у себе розмір фільтру і крок. Отриманий результат називається картою ознак, в якій всі ознаки розраховані з використанням вхідних шарів і фільтрів. Для прикладу, якщо використовується 12 фільтрів, обсяг зображення буде [28 × 28 × 12].

Шар пулінгу (об'єднання) виконує операцію для зниження розмірності, внаслідок чого обсяг може зменшитися [14 × 14 × 24]. Шар об'єднання використовується для ущільнення ознак, які звичайно застосовуються після шару згортки. Існує два типи операцій об'єднання – це максимальне і середнє об'єднання, де береться максимальне і середнє значення ознак, відповідно.

Логіка роботи така: якщо на попередній операції згортки вже були виявлені деякі властивості (ознаки), то для подальшої обробки такий детальний образ вже непотрібний, і він «ущільнюється» до менш детальної картинки.

Повнозв'язний шар виводить N-мірний вектор (N – кількість класів) для визначення потрібного класу. Це відбувається шляхом звернення до виходу попереднього шару (карта ознак) та визначення властивостей, які найбільш характерні для певного класу. Повнозв'язні шари працюють з плоским входом, де кожен пов'язаний з усіма нейронами. Вони використовуються у кінці мережі для підключення прихованих шарів до вихідного прошарку, що допомагає оптимізувати оцінки класів.

Саме у такий спосіб ЗНМ шар за шаром перетворює вхідне зображення, починаючи з вихідних значень пікселів і закінчуючи визначенням класу.

Згорткові мережі відрізняються дуже високою здатністю до розпізнавання патернів на зображеннях.

PyTorch може з легкістю справитися з операціями згорткової нейромережі. На самому початку на вхід подаються рисунки вузлів у чорно-білому кольорі розміром 28 x 28 пікселів кожен.

Фільтри у даному разі – це колекція ваг (матриці чисел), які «навчаються» для пошуку на зображеннях певних характеристик зображення (наприклад, прямих ліній). Фільтр переміщається уздовж зображення для визначення присутності даної характеристики у цій його (зображення) частині.

Результатом переміщення даного фільтру уздовж всього зображення є матриця, що складається з результатів одиничних згорток.

Кількість вихідних значень після операції згортки може бути розрахована за формулою:

$$n_{out} = floor\left(\frac{n_{in} - f}{s}\right) + 1,$$

де n – кількість вхідних пікселів, f – кількість пікселів у фільтрі, s – крок переміщення фільтру.

Кожен згортковий фільтр може бути навчений для пошуку різних ознак у зображенні, які потім можуть бути використані в класифікації. Оскільки ваги окремих фільтрів залишаються постійними, будучи застосованими на вхідних вузлах, вони навчаються вибирати певні ознаки з вхідних даних (лінії, вузли та інші форми досліджуваного об'єкту).

Вибір конкретних значень залежить від навчання кожного фільтру.

Наступний крок в структурі ЗНМ – проходження виходу операції згортки через нелінійну активаційну функцію. Кожен вузол приймає зважений вхід, активує активаційну функцію. Завдяки їй картина, яка формується за допомогою операції згортки, отримує деяке спотворення, що дозволяє нейронній мережі більш ясно оцінювати ситуацію.

Архітектура містить у собі 3 основних парадигми (в даному разі організацію обчислень і структурування для роботи програми):

– локальне сприйняття;

– спільні ваги;

– субдескретизацію.

Локальне сприйняття має на увазі, що на вхід одного нейрона подається не все зображення, а лише деяка його область. Такий підхід дозволить зберігати топологію зображення від шару до шару.

Концепція спільних ваг припускає, що для великої кількості зв'язків використовується відносно невеликий набір ваг. На якості розпізнавання це позначиться в кращу сторону. Штучно введене обмеження на ваги покращує узагальнюючі властивості мережі, що в результаті позитивно позначається на здатності мережі знаходити інваріанти в зображенні і реагувати головним чином на них, не звертаючи уваги на інший шум. У разі розпізнавання зображень на практиці більшість таких систем будуються на основі двовимірних фільтрів. Матриця коефіцієнтів, якою являється фільтр, застосовується до зображення за допомогою математичної операції (згортки). Суть цієї операції у тому, що кожен фрагмент зображення множиться на матрицю (ядро) згортки поелементно і результат підсумовується та записується в аналогічну позицію вихідного зображення. Основна властивість таких фільтрів полягає у тому, що значення їх виходу тим більше чим більший фрагмент зображення схожий на сам фільтр. Таким чином зображення згорнуте з якимось ядром дасть нам інше зображення, кожен піксель якого буде означати ступінь схожості фрагмента зображення на фільтр. Іншими словами це буде карта ознак.

Суть субдискретизації полягає у зменшенні просторової розмірності зображення. Тобто вхідне зображення грубо (усередненням) зменшується в задану кількість разів.

Структура майбутньої моделі

Перший шар згорткової мережі, який на вхід приймає зображення, на виході віддає карти ознака.

Для зниження розмірності карт ознак використовується шар макспулінгу. Тут відбувається вибір максимального значення із заданого вікна. «Класичні» значення параметрів макспулінгу в параметрах згортки – це крос-кореляція і позиція центрального елемента в лівому верхньому кутку.

Другий шар згорткової мережі приймає отримані на попередньому кроці карти і на виході дає інші карти ознака.

Перший шар повнозв'язної мережі приймає вектор, виробляє обчислення для прихованого повнозв’язного шару.

В другому шарі повнозв'язної мережі кількість вихідних нейронів дорівнює кількості класів yData set., який використовується.

Вихід всієї моделі подається у функцію втрат (функція втрат знаходиться у самому кінці), яка порівнює прогнозоване значення з істинним і розраховує різницю між цими значення. Завершальний етап мережі – функція, яка оцінює якість роботи всієї моделі.

Завдяки бібліотеці NumPy програмування згорток не складе великих труднощів.

На самому верхньому рівні буде знаходитися цикл for, який буде застосовувати фільтри до вихідного зображення. В рамках кожної ітерації буде використано два цикли while, які будуть переміщати фільтр уздовж зображення (горизонтально і вертикально).

Висновки. Розглянуто два методи підготовки навчальних і тестових прикладів. Запропоновані два методи розпізнавання плоских графічних фігур та розпізнавання зв'язків між фігурами в блок-схемах: метод рекурсивного обходу всіх гілок дерева поточного об'єднання зв'язків між фігурами та самих фігур, а також проведено дослідження для створення нейронної мережі в PyTorch.

Список літератури

1. Howard A.G., Zhu M., Chen B., Kalenichenko B., Wang W., Wey T., Andreetto M., Adam H. MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017. https://arxiv.org/abs/1704.04861 (звернення: 09.12.2021)
2. Хант Е. Искусственный интеллект. М .: Мир, 1978. 560 с.
3. Патент США № 6775411, кл. МПК G 06 K 9/62, від 13.01.2005.
4. Фу К. Структурные методы в распознавании образов. М.: Мир, 1977. 320 с.
5. Чичирин Е.Н. Распознавание структуры графических изображений блок-схем. *Комп`ютерні засоби, мережі та системи*. 2017. № 16. С. 87 – 96. http://dspace.nbuv.gov.ua/handle/123456789/131513
6. LeCun Y. LeNet-5, convolutional neural networks. November 2015. LeNet-5, convolutional neural networks. (звернення: 09.12.2021)
7. Палагін О.В, Чічірін Є.М., Сосненко К.П. Спосіб розпізнання структури плоских графічних зображень. Патент на корисну модель № 13061.
8. Чічірін Є.М., Сосненко К.П. Нейромережеве моделювання процесів розпізнавання графічних схем. *Комп`ютерні засоби, мережі та системи*. 2019. № 18. С. 79 – 85. http://dspace.nbuv.gov.ua/handle/123456789/168481
9. Палагін О.В., Семотюк М.В., Чічірін Е.Н., Сосненко К.П. Моделююче середовище для створення і налагодження систем цифрової обробки. *Управляющие системы и машины*. 2013. № 1. С. 37 – 41, 70. http://usim.org.ua/arch/2013/1/5.pdf
10. Canziani A., Paszke A., Culurciello E. An analysis of deep neural network models for practical applications. ISCAS, 2017. https://arxiv.org/abs/1605.07678 (звернення: 09.12.2021)

Одержано 09.12.2021

Сосненко Катерина Петрівна,
молодший науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України, Київ.
sosnenko.kate@ukr.net

MSC 90C15

Kateryna Sosnenko

Two Approaches for Recognizing the Structure of Block Diagrams

V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine, Kyiv
Correspondence: sosnenko.kate@ukr.net

Introduction. Working with graphic images is an essential element of almost any modern computer-aided design systems.

The result of neural network, including deep image processing, can be recognition of the belonging classes of objects present on them. Objects of the real world require large expenditures for the development and implementation of highly specialized computer vision systems.

In recent years, there has been an improvement of the quality characteristics obtained in the field of technical vision. This is made possible by artificial neural networks.

The article deals with the recognition of a flat, black and white flowchart image. These are two-dimensional images or their selected parts, which are displayed in an arbitrary graphic format by system means on a computer monitor screen. Basic block diagram shapes: rectangle, rhombus, parallelogram, circle, ellipse (oval), etc.

The purpose of the article is to solve the problem of graphic image recognition. The systems for the recognition of graphic images include modern systems of computer-aided design, management and document management. The article has formed a basic set of training and test images of block diagram nodes. Neural network models are proposed to improve the detection accuracy of block diagram nodes based on fully connected and convolutional neural networks.

Results. The basic procedures of the block diagram image recognition algorithm were partially tested at the software level, and allow us to conclude the effectiveness of the proposed structural methods. A comparative analysis of neural network and syntactic structural approaches for solving this problem is carried out.

Conclusions. Two methods of recognizing flat graphic figures and recognizing connections between figures in flowcharts are proposed: a method of recursive traversal of all branches of the tree of the current union of connections between figures and the figures themselves, and also a study was carried out for the created neural network in PyTorch to solve this problem using trained neural network methods.

Keywords: image recognition, convolutional neural networks, syntactic analysis.