УДК 519.688

DOI 10.31651/2076-5886-2019-1-61-75

PACS 02.60.-x, 02.60.Pn

КУЧЕР Олександр Іванович,
аспірант кафедри прикладної математики
та інформатики Черкаського національного
університету імені Богдана Хмельницького
e-mail: olexandr.kucher@gmail.com
ORCID 0000-0002-6902-5068

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ЗАДАЧ ОБРОБКИ ЗОБРАЖЕНЬ

У статті дано базове поняття та характеристики згорткових нейронних мереж, описана структура та математичні підходи до реалізації нейронної мережі цього типу. Наданий опис основних шарів згорткових нейронних мереж, наведені параметри, завдяки яким можна змінювати процес навчання мережі. Детально розглянутий процес навчання кожного з шарів нейронної мережі. Додатково описані техніки, що дають можливість збільшити набір даних та покращити процес навчання мережі за допомого кількох перетворень

Ключові слова: *глибоке навчання, машинне бачення, нейронні мережі, розпізнавання облич, згорткові нейронні мережі, рецептивні поля, глибоке навчання, нейронні мережі, згорткові нейронні мережі, класифікація, локалізація, виявлення, сегментація, випрямлені лінійні одиниці.*

Постановка проблеми

Згорткові нейронні мережі (ЗНМ, Convolutional Neural Networks, CNN) – одні з найвпливовіших інновацій в області комп'ютерного зору. Вперше нейронні мережі привернули до себе увагу в 2012 році на конкурсі ImageNet. За допомого ЗНМ було встановлено новий рекорд помилок при класифікації – 15% (попереднє значення було 26%) [1]. На сьогоднішній день глибоке навчання (deep learning) використовують безліч відомих компаній:

- Facebook – використовує для алгоритмів автоматичного виставлення тегів
- Google – для пошуків по фотографіях
- Amazon – для генерації рекомендації товарів

Але класичний і найбільш популярний спосіб застосування нейронних мереж це обробка зображень. Розглянемо як ЗНМ використовуються для класифікації зображень.

Виклад основного матеріалу

Задача класифікації зображень – це обробка зображення і визначення класу до якого воно належить (автомобіль, тварина і т. д.) або групи класів, які найкраще його характеризують.

Вхідні і вихідні дані

Коли комп'ютер отримує на вхід оцифроване зображення, він працює з масивом пікселів. Залежно від розширення зображення розміри масиву можуть бути, наприклад, 64x64x3 (де 3 – це значення кожного з каналів RGB). Для кращого розуміння уявимо, що в нас є кольорове зображення в форматі JPG розміром 360x360 пікселів. Воно буде представлено масивом з розмірами 360x360x3. Кожне із значень масиву набуває значення від 0 до 255, що описує інтенсивність пікселя. Для людини ці значення не мають сенсу, але це єдиний спосіб представити зображення в комп'ютері. Ідея полягає в тому, щоб надати програмі (алгоритму) цю матрицю і отримати в результаті значення

що описують ймовірність відношення цього зображення до певного класу (0.8 – автомобіль, 0.15 – тварина, 0.05 - людина).

Рис. 1. Що бачить людина та що «бачить» комп'ютер

Зв'язок з біологією

ЗНМ – це прототип зорової кори головного мозку. Зорова кора має невеликі ділянки клітин, що є чутливими до певних областей поля зору. Цю ідею детально розглянули за допомогою експерименту Г'юбел і Візел в 1962 році в якому показали, що окремі нервові клітини реагували лише при візуальному сприйнятті меж певної орієнтації. Наприклад, деякі нейрони активувалися, коли сприймали вертикальні границі, а деякі – горизонтальні або діагональні. Г'юбел і Візел з'ясували, що всі нейрони розташовані в виді стержневої архітектури і разом формують візуальне сприйняття. Цю ідею спеціалізованих компонентів, що розв'язують якусь задачу і використовують машини, і ця ідея – основа ЗНМ. [7]

Структура

Зображення пропускається через серію згорткових, нелінійних шарів, шарів об'єднання і повнозв'язних шарів в результаті чого генеруються вихідні дані. Вихідними даними може бути клас або ймовірність класів, які найкраще описують зображення.

Перший шар – математична частина

Перший шар в ЗНМ завжди згортковий. Найлегше зрозуміти, що таке згортковий шар, допоможе представлення його у вигляді ліхтарика, який світить на верхню ліву частину зображення. Припустимо, що світло з ліхтарика покриває площу 5х5 пікселів. В термінах комп'ютерного навчання цей ліхтарик – це фільтр, а область на яку він світить називається рецептивним полем. Глибина фільтра повинна бути такою ж як глибина вхідного зображення (5х5х3). Оскільки фільтр здійснює згортку, тобто рухається по зображенню, він перемножує значення фільтра на вихідні значення пікселів зображення (поелементно). В результаті отримуємо одне число. Тепер необхідно повторити цей процес в кожній позиції (Наступний крок – це переміщення фільтра вправо на 1 піксель). Кожна унікальна позиція зображення генерує одне число. Після проходження фільтра по всіх позиціях (для зображення 32х32х3) отримуємо матрицю 28х28х1, яку називають функцією активації або картою ознак. Матриця 28х28 отримується тому, що є 784 різні позиції, які можуть пройти через фільтр 5х5. Ці 784 числа перетворюються в матрицю 28х28. [6]

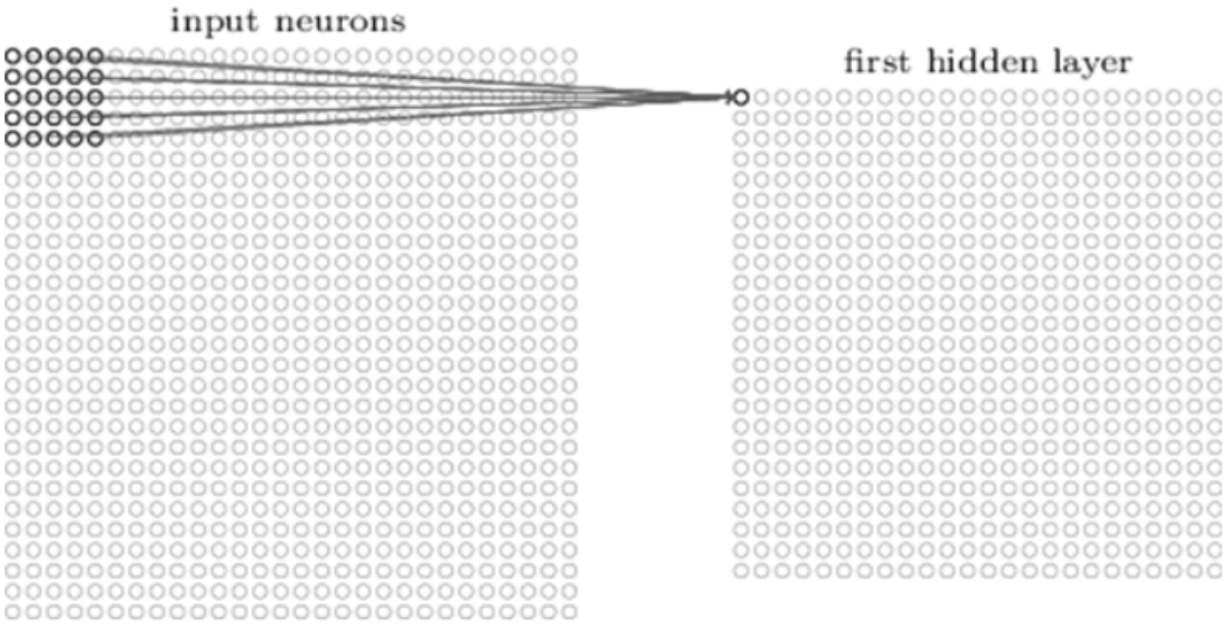

Рис. 2. Візуалізація фільтра 5x5, що створює карту активації з вхідних даних

Якщо ми використовуватимемо два 5x5x3 фільтри замість одного, то отримаємо матрицю 28x28x2.

Перший шар

Кожен фільтр можна розглядати як ідентифікатор властивості. Під властивостями маються на увазі прямі межі, прості кольори і криві. Припустимо, що наш перший фільтр 7x7x3 і він буде детектором кривих. (Для простоти розглянемо лише верхні шари фільтра і зображення). Фільтр має піксельну структуру, у якої числові значення більші вздовж області, що визначає форму кривої.

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

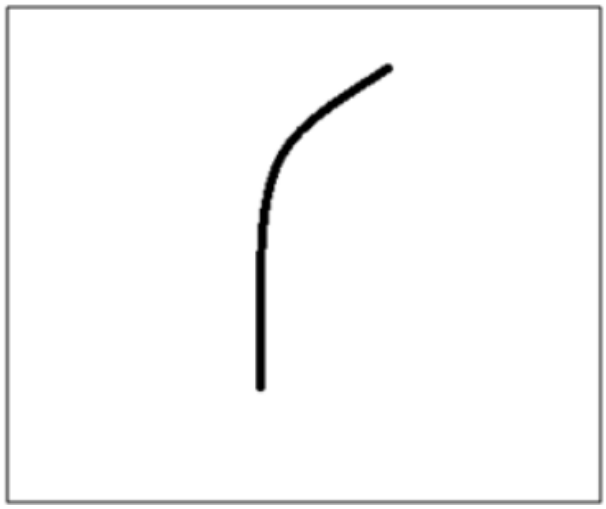

Рис. 3. Цифрове та візуальне представлення фільтра що знаходить криві

Повернемося до математичної візуалізації. Коли в верхньому лівому куті вхідного зображення є фільтр, він здійснює множення значень фільтра на значення пікселів області. Розглянемо приклад зображення, яке потрібно класифікувати, і встановимо фільтр в верхньому лівому куті.

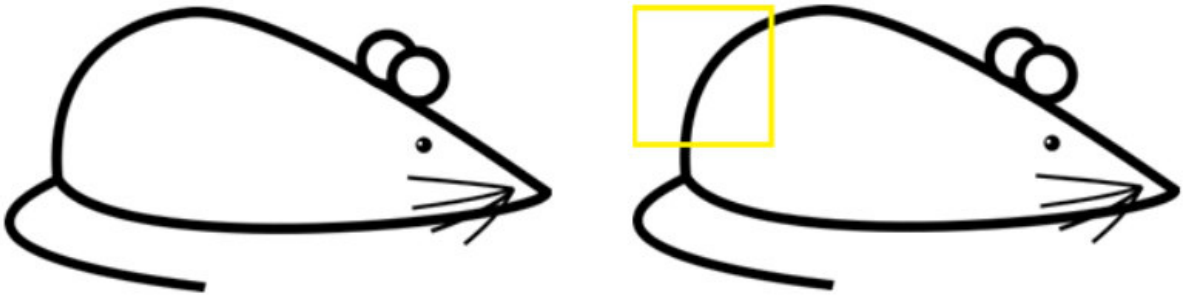

Рис. 4. Візуалізація фільтра на зображенні

Все що необхідно зробити – це перемножити значення фільтра на значення пікселів зображення.

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Рис. 5. Застосування фільтра в верхньому лівому куті зображення

Multiplication and Summation =

$$(50\times 30)+(50\times 30)+(50\times 30)+(20\times 30)+(50\times 30) = 6600\,.$$

Якщо на зображенні є форма, що схожа на криву, яку представляє фільтр, то всі перемножені і додані значення утворюють значення на кілька порядків більше за 0 (порогове значення встановляється програмістом).

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Рис. 6. Застосування фільтра в верхньому правому куті зображення

***Multiplication and Summation* = 0**

Коли ми перемістимо фільтр, то отримаємо значення близьке до нуля. Причиною є те, що в цій області зображення немає нічого, що фільтр кривої міг виявити. Пам'ятайте – результат цього згорткового шару – це карта властивостей. В найпростішому випадку, при наявності одного фільтра згортки, карта властивостей виявить області, в яких з більшою ймовірністю є криві (якщо фільтр – детектор кривих). В цьому прикладі значення карти властивостей у верхньому лівому куті буде 6600. Це значення показує, що, можливо, щось схоже на криву є на зображенні, і така ймовірність активувала фільтр. В правому верхньому куті значення карти властивостей

буде 0, тому що на зображенні не було нічого, що могло б активувати фільтр. Ці значення є характеристиками лише одного фільтра, фільтра що знаходить криві зі згином назовні. Можуть бути й інші фільтри. Чим більше фільтрів, тим більша глибина карти властивостей і більше інформації про зображення можна отримати. [8]

Глибше через мережу

Зараз в традиційній згортковій мережі існують й інші шари, що перемножуються з згортковими. Класична архітектура ЗНМ матиме наступний вигляд:

Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool -> Fully Connected

де ***Conv*** – згортковий шар, ***ReLU*** – функція активації, ***Pool*** – об'єднуючий шар, ***Fully Connected*** – повнозв'язний шар.

Ми розглянули фільтри першого шару. Вони виявляють властивості базового рівня, такі як межі та криві. Для того щоб розпізнавати типи об'єктів на зображеннях, нам потрібна мережа, що здатна розпізнавати об'єкти більш високого рівня, наприклад руки, лапи або вуха. Результат обробки зображення першим фільтром має розміри 28x28x3 (за умови що використовувався фільтр 5x5x3). Коли зображення проходить через один згортковий шар, результат першого шару стає вхідними даними другого. Коли ми описували перший шар, вхідними даними було лише початкове зображення. Але коли ми перейшли до другого шару, вхідними значеннями для нього стала одна або декілька карт властивостей – результат обробки попереднього шару. Кожен набір даних описує позиції, де на початковому зображенні знаходяться певні базові ознаки. [1]

Тепер, коли застосовується набір фільтрів поверх цього (зображення пропускається через другий згортковий шар), в результаті будуть активовані фільтри, які представляють властивості більш високого рівня. Типами цих властивостей можуть бути півкільця або квадрати. Чим більше згорткових шарів застосовується до зображення, тим більш складні характеристики проявляються в картах активації. В кінці мережі можуть бути фільтри, що активуються при наявності рукописного тексту на зображенні, при наявності зелених об'єктів і т. д.

При русі в глибину мережі, фільтри працюють з усе більшим полем сприйняття, а значить, вони в змозі опрацьовувати інформацію з більшої площі початкового зображення (вони краще адаптуються до обробки більшої області піксельного простору).

Повнозв'язні шари

Тепер коли можливо виявити високорівневі властивості, можна прикріпити повнозв'язний шар вкінці мережі. Цей шар бере вхідні дані і виводить N-мірний вектор, де N – число класів, серед яких програма обирає потрібний. Наприклад, якщо ви працюєте над програмою розпізнавання цифр, N буде рівне 10. Кожне значення в цьому векторі представлятиме собою ймовірність конкретного класу. Наприклад, якщо результуючий вектор для програми розпізнавання цифр це: ***[0 0.1 0.1 0.75 0 0 0 0 0 0.05]*****,** значить існує ймовірність 10% що на зображенні «1», ймовірність 10%, що на зображенні «2», ймовірність 75%, що на зображенні «3», і ймовірність 5%, що на зображенні «9».

Спосіб, за допомогою якого працює повнозв'язний шар – це взаємодія з вихідними даними попереднього шару (який повинен виводити високорівневі карти властивостей) і визначення властивостей, які більш зв'язані з певним класом. Повнозв'язний шар виявляє, що функції високого рівня сильно зв'язані з певним класом і мають певні коефіцієнти ваги, тобто, коли відбувається обрахунок ваги з попереднім шаром, отримуються правильні ймовірності для різних класів.

Навчання (або що змушує це працювати)

Спосіб за допомогою якого комп'ютер здатний корегувати значення фільтра (або коефіцієнтів ваги) – це процес навчання, який називається методом зворотного розповсюдження помилки.

Що ж нейронній мережі необхідно для роботи. До моменту побудови мережі, ваги або значення фільтра випадкові. Фільтри не вміють шукати межі і криві. Фільтри верхніх шарів не вміють шукати руки, лапи і вуха. Для навчання мережі застосовується механізм надання їй зображення і ярлика чи класу, що йому відповідає.

Метод зворотного розповсюдження помилки можна розділити на 4 окремих блоки: пряме розповсюдження, функція втрати, зворотне розповсюдження і оновлення ваги. Під час прямого розповсюдження, береться тренувальне зображення (це матриця 32x32x3) і пропускається через всю мережу. В першому прикладі, оскільки всі ваги або значення фільтрів були ініційовані випадковим чином, вихідним значенням буде щось на зразок ***[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1]***, тобто таке значення, яке не надає перевагу жодному з наявних класів. Мережа з такими властивостями не здатна знайти властивості базового рівня і не здатна аргументовано визначити клас зображення. Це веде до функції втрати. На цьому етапі використовуються дані для навчання. У таких даних є зображення і ярлик. Якщо перше зображення для навчання це цифра 3, то ярликом буде ***[0 0 0 1 0 0 0 0 0 0]***. Функція втрати може бути виражена по різному, але часто використовується середньоквадратична помилка.

$$E_{total} = \sum \frac{1}{2}(target - output)^2 . \quad (1)$$

Нехай це значення буде **L.** Втрата буде дуже великою для перших двох зображень. Для того щоб прогнозований ярлик був таким же, як ярлик зображення для навчання, потрібно звести до мінімуму кількість втрат, які в нас є. Розглядаючи цю задачу як задачу оптимізації з математичного аналізу, необхідно з'ясувати, які вихідні дані сприяли втратам (помилкам) мережі.

Один зі способів візуалізувати ідею мінімізації втрат – це 3-D графік, де ваги нейронної мережі (їх більше 2-х, але тут приклад спрощений) це незалежні змінні, а залежна змінна – це втрата. Задача мінімізації втрат – відрегулювати ваги таким чином, щоб знизити втрату. Візуально необхідно наблизитися до найнижчої точки чашоподібного об'єкту. Щоб зробити це, необхідно знайти похідну втрати (в рамках зображеного графіка – розрахувати кутовий коефіцієнт в кожному з напрямків) з врахуванням ваг.

Це математичний еквівалент ***dL/dW***, де W – ваги одного шару. Далі необхідно виконати **зворотне розповсюдження** через мережу, яке визначає, які ваги здійснили більший вплив на втрати, і знайти способи, як їх налаштувати, щоб зменшити втрати. Після знаходження похідної, переходимо до останнього етапу – оновлення ваг. Необхідно взяти всі ваги фільтрів і оновити їх так, щоб вони змінювалися в напрямку градієнту.

$$w = w_i - n\frac{dL}{dW}, \quad (2)$$

де w - вага, w_i - початкова вага, n - швидкість навчання.

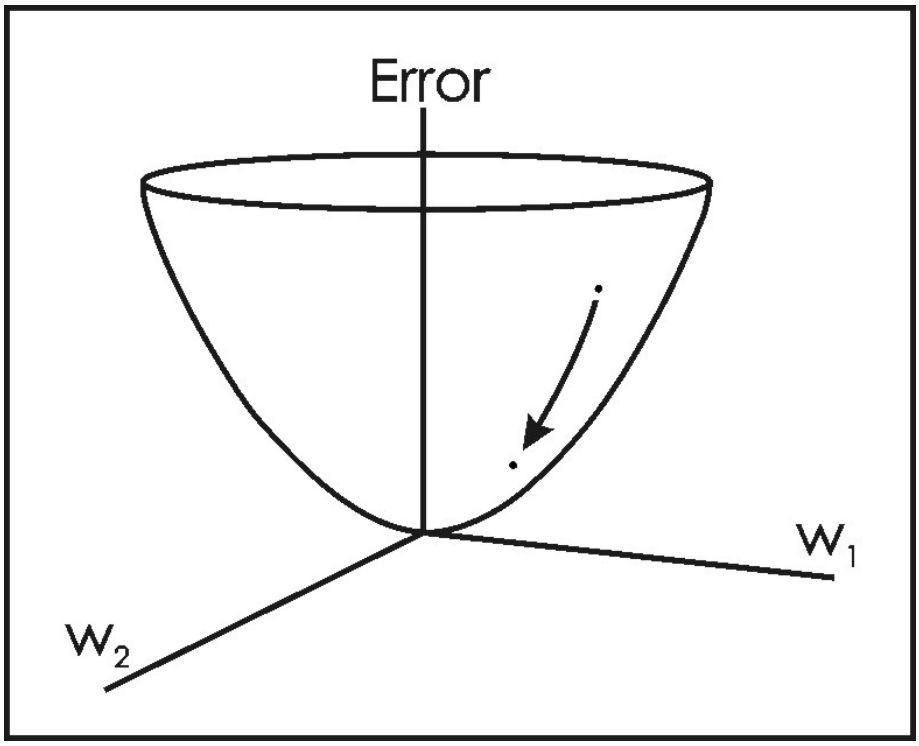

Рис. 7. Візуалізація задачі мінімізації

Швидкість навчання – це параметр, який обирається програмістом. Висока швидкість навчання значить, що в нових коефіцієнтах ваги були зроблені більші кроки, завдяки чому може знадобитися менше часу, щоб отримати оптимальний набір ваг. Але занадто велика швидкість навчання може привести до дуже великих і не достатньо точних стрибків, які завадять досягненню оптимальних показників.

Процес прямого розповсюдження, функцію втрати, зворотне розповсюдження і оновлення ваг, зазвичай називають одним періодом дискретизації (або epoch - епохою). Програма буде повторювати цей процес фіксовану кількість періодів для кожного тренувального зображення. Після того, як оновлення параметрів завершиться на останньому тренувальному зразку, мережа повинна бути достатньо добре навчена і ваги шарів повинні бути налаштовані правильно.

Тестування

Нарешті, щоб побачити, чи працює ЗНМ, ми беремо інший набір зображень і ярликів і пропускаємо зображення через мережу. Порівнюємо результати з реальними даними і перевіряємо чи працює наша мережа.

Крок і відступ

При роботі з згортковими нейронними мережами (ЗНМ) ми можемо змінювати 2 параметри, для того, щоб змінити поведінку кожного з шарів. Після обранні розміру фільтра, ми можемо також обрати **крок** і **відступ**.

Крок – це параметр, що контролює процес згортки вхідного зображення за допомогою фільтра. На Рис. 8 зображено приклад згортки вхідних даних зі зміщенням на одну комірку за раз. Кількість комірок, на які зміщується фільтр на кожному кроці, називається кроком. На Рис. 8 розмір кроку – 1. За крок завжди беруть ціле число.

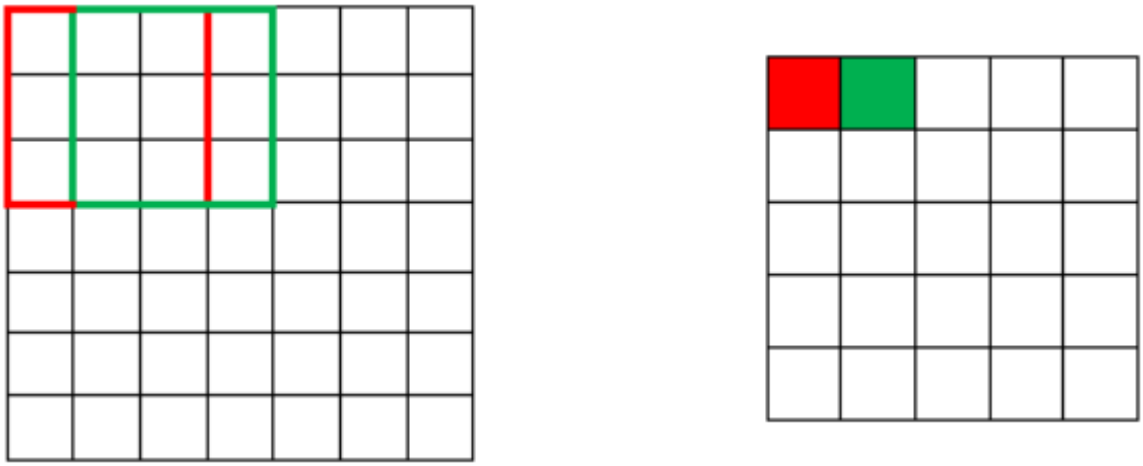

Рис. 8. Вхідні і вихідні дані фільтра з кроком 1

На Рис. 9 зображено фільтр з кроком 2, використання якого призводить до зменшення даних на виході. Як правило, крок фільтру збільшують для зменшення перекриття рецептивних полів та зменшення кількості даних на виході фільтра.

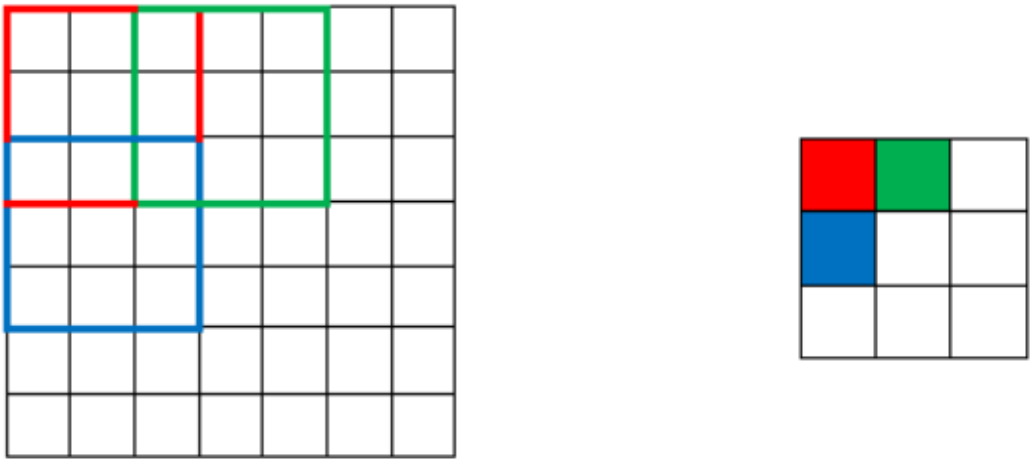

Рис. 9. Вхідні і вихідні дані фільтра з кроком 2

Тепер давайте поглянемо на відступ. Що відбудеться, якщо застосувати три фільтри 5x5x3 до вхідних даних розміром 32x32x3? Розмір вихідних буде 28x28x3. Оскільки ЗНМ побудована на застосуванні ряду згорткових шарів, то розмір вихідних даних після застосування кожного з них зменшуватиметься швидше, ніж хотілося б. У ранніх шарах мережі ми хочемо зберегти максимальну кількість інформації про вхідні дані. Скажімо, ми хочемо застосувати один і той же згортковий шар, але хочемо, щоб обсяг вихідних даних залишався 32x32x3. Для цього ми можемо застосувати нульовий відступ розміру 2 до цього шару. Нульовий відступ збільшує розмір матриці вхідних даних за допомогою додавання нулів «навколо».

На Рис. 10 зображено вхідні дані, представлені матрицею з розміром 32x32x3. Якщо додати нулі «навколо», ми отримаємо матрицю з розміром 36x36x3. Після застосування згорткового шара з 3-ма фільтрами 5x5x3 і кроком 1, отримаємо вихідну матрицю все того ж розміру – 32x32x3. [3]

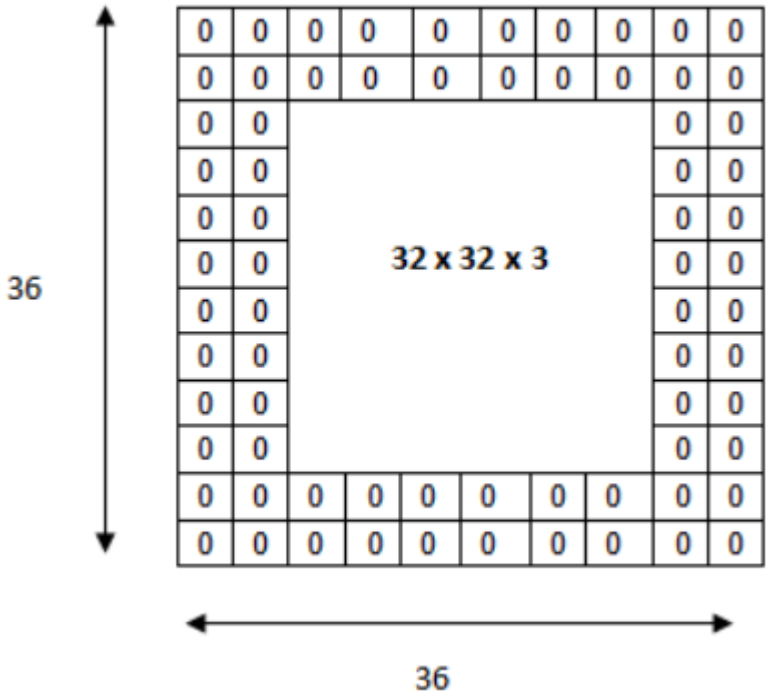

Рис. 10. Вхідні дані розміром 32x32x3 з відступом – 2

Якщо розмір кроку – 1 і розмір відступу задати за формулою:

$$P = \frac{K-1}{2}, \tag{3}$$

де K - розмір фільтру. Розмір матриці вхідних даних завжди дорівнювати розміру матриці вихідних даних.

Формула для обрахунку розмірів вихідних даних для будь-якого згорткового шару має наступний вигляд:

$$O = \frac{W - K + 2 \cdot P}{S} + 1, \tag{4}$$

де O – вихідна висота або ширина, W – вхідна висота або ширина, K – розмір фільтра, P – відступ, S – крок.

Вибір параметрів навчання

Як визначити, скільки шарів слід використовувати, скільки згорткових шарів, які розміри фільтрів та значення для кроку і відступу? Це не тривіальні питання і немає встановленого стандарту, який використовують усі дослідники. Це пояснюється тим, що мережа значною мірою залежить від типу даних, з якими вона працює. Дані можуть відрізнятися за розміром, складністю зображення, типом завдання обробки зображень тощо. Один із способів обрати параметри – це знайти правильну комбінацію, яка створює абстракції зображення у належному масштабі.

ReLU (Rectified Linear Units) шари

Після кожного згорткового шару прийнято застосовувати нелінійний шар (або шар активації). Метою цього шару є введення нелінійності в систему, яка в основному просто обраховує лінійні операції під час застосування згорткових шарів (множення і додавання елементів). В минулому застосовувалися нелінійні функції, такі як tanh і sigmoid, але дослідники з'ясували що ReLU шари працюють значно краще, то му що мережа має змогу навчатися значно швидше (через обчислювальну ефективність) без істотної різниці в точності. Це також допомагає полегшити проблему зникаючого градієнта, що є проблемою, коли нижні шари мережі навчаються дуже повільно, через градієнт що експоненційно зменшується. Шар ReLU застосовує функцію $f(x) = \max(0, x)$ до всіх вхідних значень. Коротко кажучи, ця функція просто замінює всі негативні значення на 0. ReLU шар збільшує нелінійні властивості моделі та мережі в загальному, не впливаючи на рецептивні поля згорткового шару. [5]

Шари об'єднання (Pooling layers)

Після деяких шарів ReLU розробники можуть застосувати шар об'єднання. Його також називають шаром зменшення. В цій категорії існує також кілька варіантів, найпопулярнішим є макспулінг (maxpooling, рис. 11). Такий підхід застосовує фільтр (зазвичай розміром 2x2) з кроком такого ж розміру. Фільтр застосовується до вхідних даних і залишає лише максимальне значення в кожній з областей згорнутих фільтром.

Інші підходи до об'єднання – це середнє значення та ***L2-norm***, яка визначається за формулою:

$$|x| = \sqrt{\sum_{k=1}^{n} |x_k|^2} \,. \qquad (5)$$

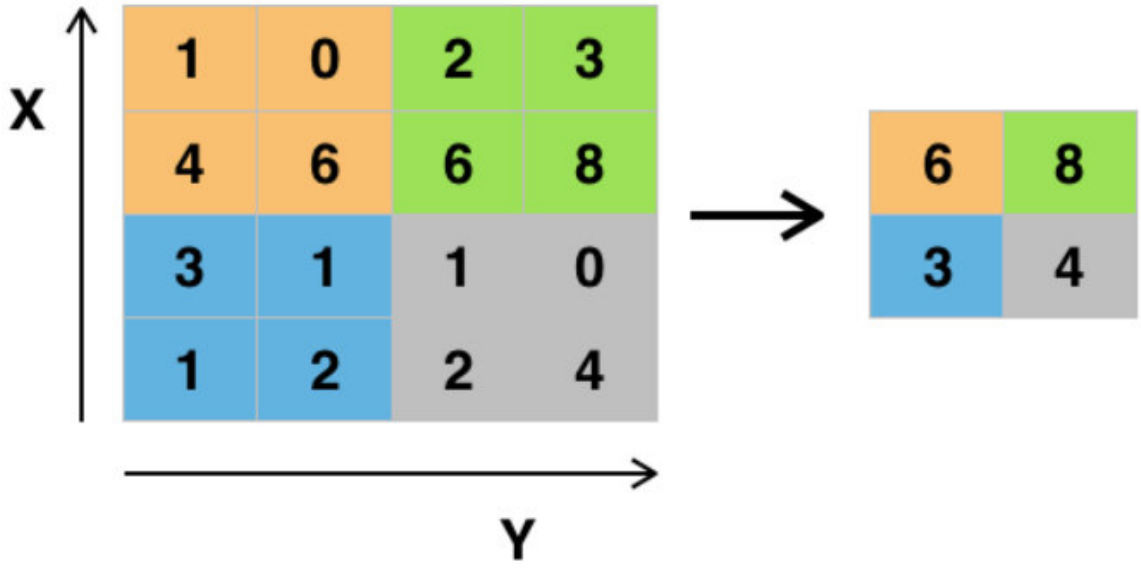

Рис. 11. Макспулінг з фільтром розміром 2x2 і кроком 2

Коли ми дізнаємося, що специфічна характеристика є у вхідних даних (в цій області буде високе значення активації), її точне розташування не так важливе як відносне розташування відносно інших характеристик. ReLU шар дуже швидко зменшує розміри (довжину і ширину, але не глибину) вхідних даних. Такі

трансформації мають 2 основні цілі. Перша – зменшення кількості параметрів або ваг на 75%, що зменшує складність обчислень. Друга – контроль ***перенавчання (overfitting).*** Це поняття описує процес при якому модель настільки налаштована на навчальні приклади, що не в змозі узагальнити дані для перевірки на тестових даних. Симптомом перенавчання є наявність моделі, яка видає 100% або 99% результат на навчальних даних, але лише 50% результат на тестових даних.

Шари відсівання (Dropout Layers)

Шари відсівання мають дуже специфічну функцію в нейронних мережах. Цей шар «відсіває» випадкові набори активацій, встановлюючи їх значення рівним нулю. Які ж переваги такого простого і, здавалося б, непотрібного і неінтуїтивного процесу? Такий підхід змушує мережу забезпечувати правильну класифікацію результату для конкретного прикладу, навіть якщо деякі з активацій відкинуті. Це гарантує, що мережа не стане занадто «пристосованою» до навчальних даних, а отже, знижує ймовірність виникнення проблеми перенавчання (перенасичення). Варто зазначити, цей шар використовується лише під час навчання, але не під час тестування. [11]

Шар мережі в мережі (Network in Network Layers)

Шар мережі в мережі посилається на згортковий шар з фільтром розміру 1x1. Виникає питання, чому цей шар є корисним, оскільки рецептивні поля зазвичай більші за простір, на який вони відображаються. Однак, згортка розміром 1x1 охоплює певну глибину, тому таку згортку необхідно розглядати як згортку розміром 1x1xN, де N – кількість фільтрів застосованих в цьому шарі. Фактично цей шар виконує N-D елементарне множення, де N – глибина вхідних даних шару. [10]

Класифікація, локалізація, виявлення, сегментація

Класифікація зображень - це процес отримання вхідного зображення і виведення номеру або назви класу з набору категорій. Але для задачі **локалізації** об’єктів потрібно не лише визначити клас, а і створити обмежувальну рамку, яка описує, де об’єкт знаходиться на зображенні.

Також є завдання **виявлення об’єктів**, де локалізація повинна здійснюватися на всіх об’єктах зображення (рис. 12, 13). Таким чином з’являється кілька обмежувальних полів і декілька класів.

Рис. 12. Класифікація – визначення що зображення це собака

Рис. 13. Локалізація – визначення класу та знаходження об’єкта на зображенні

Трансферне навчання (TransferLearning)

Поширена помилка у спільноті машинного навчання полягає в тому, що без величезних об’ємів даних (таких як в Google), ми не маємо можливості створити ефективні моделі глибокого навчання. Хоча дані є важливою частиною створення мережі, ідея трансферного навчання допомогла зменшити вимоги до даних. **Трансферне навчання** – це процес взяття попередньо навченої моделі (ваги та параметри мережі, яка була навчена на великій кількості даних кимось іншим) і «тонкого налаштування» моделі з нашим власним набором даних. Ідея полягає в тому, що ця попередньо навчена модель буде виступати в якості екстрактора ознак. Ми вилучаємо останній шар мережі і заміняємо його власним класифікатором (залежно від області задачі). Далі ми заморожуємо ваги всіх інших шарів і навчаємо мережу стандартним способом (замороження означає заборону змін ваг під час оптимізації).

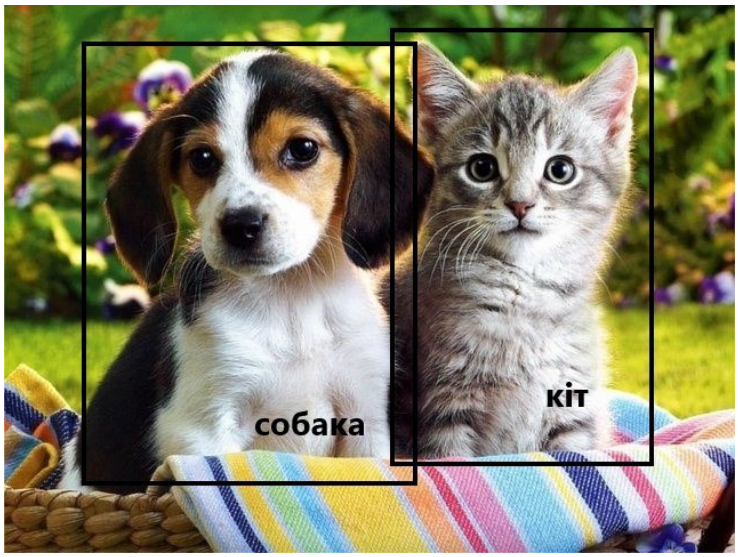

Рис. 14. Виявлення об’єктів – локалізація кількох об’єктів

Завдання **сегментації об’єктів** – це визначення класу та контуру кожного об’єкта у вхідному зображенні (рис. 15). [4]

Рис. 15. Сегментація об’єктів – визначення класу об’єкта та його виділення на вхідному зображенні

Розглянемо, чому це працює. Припустимо, попередньо підготовлена модель була навчена на ImageNet (ImageNet – набір даних, що містить 14 мільйонів зображень з більш ніж 1000 класами) [2]. Коли ми говоримо про низькорівневі шари мережі, ми знаємо, що вони будуть виявляти такі риси, як ребра та криві. Тепер, якщо в нас не дуже унікальний простір проблем і набір даних, мережі потрібно буде виявити криві і ребра. Замість того, щоб навчати мережу через ініціалізацію ваг випадковими значеннями, ми можемо використати ваги вже навченої моделі і зосередитися на більш важливих для навчання шарах. Якщо ж наш набір даних абсолютно інший ніж той що в ImageNet, тоді необхідно збільшити кількість шарів і заморозити лише кілька нижніх. [9]

Методи збільшення даних

Розглянемо, як можна збільшити існуючий набір даних лише за допомогою кількох легких перетворень. Як вже згадувалося раніше, коли комп’ютер отримує на вхід зображення, він отримує його у виді масиву чисел, що представляють пікселі. Припустимо, що все зображення зміщується вліво на один піксель. Для людини ця зміна непомітна. Однак для комп’ютера цей зсув може бути досить значним, оскільки клас зображення не змінюється, а масив пікселів – так. Підходи, що змінюють навчальні дані способами, які модифікують подання масиву, зберігаючи клас зображення, відомі як **методи збільшення даних.** Вони є способом штучного розширення набору даних. Також популярними підходами є використання чорно-білих зображень, випадкові обрізання, обертання та багато іншого. Застосувавши лише кілька цих перетворень до навчальних даних, можна легко подвоїти або потроїти кількість навчальних прикладів.

Висновки

У статті розглянуто поняття згорткових нейронних мереж, їх зв'язок з біологією та структуру. Далі більш детально розглянуто основні шари, що є в ЗНМ, описано їх роботу за допомогою математики. Кожен з шарів розглянуто окремо та у взаємодії з іншими шарами нейронної мережі.

Показано як нейронні мережі працюють, та як їх правильно навчати. Розглянуто кілька підходів до навчання нейронних мереж та прості способи вирішення поширених проблем, що виникають при навчанні: збільшення об’єму даних для навчання та

використання вже навчених мереж як бази для вирішення вузькоспеціалізованих задач. Наведено параметри, завдяки яким можна змінювати процес навчання мережі.

Розглянуто кілька найпоширеніших задач, які розв'язують за допомогою згорткових нейронних мереж: класифікація, локалізація, виявлення, сегментація.

Список використаної літератури:

1. C. Zhang, and Z. Zhang, "Improving multiview face detection with multi-task deep convolutional neural networks," IEEE Winter Conference on Applications of Computer Vision, 2014, pp. 1036-1041.
2. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09.
3. Dumitru Erhan, Christian Szegedy, Alexander Toshev, Dragomir Anguelov, Scalable Object Detection using Deep Neural Networks, arXiv preprint arXiv: arXiv:1312.4400, 2014.
4. E. Borenstein; J. Malik, "Shape Guided Object Segmentation" in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.
5. Geoffrey E. Hinton, Vinod Nair. "Rectified Linear Units Improve Restricted Boltzmann Machines", Proceedings of International Conference on Machine Learning (ICML), 2010, pp 807-814.
6. H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5325-5334.
7. Hubel, D. H., & Wiesel, T N. (1979). Brain mechanisms of vision. Scientific American, 241, 150–162.
8. J. Yan, Z. Lei, L. Wen, and S. Li, "The fastest deformable part model for object detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2497-2504.
9. Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, How transferable are features in deep neural networks?, arXiv preprint arXiv: arXiv: 1411.1792v1, 2014.
10. Min Lin, Qiang Chen, Shuicheng Yan, Network In Network, arXiv preprint arXiv: arXiv: 1312.2249, 2013.
11. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research 15, 2014, pp. 1929-1958.

Bibliography:

1. C. Zhang, and Z. Zhang, "Improving multiview face detection with multi-task deep convolutional neural networks," IEEE Winter Conference on Applications of Computer Vision, 2014, pp. 1036-1041.
2. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09.
3. Dumitru Erhan, Christian Szegedy, Alexander Toshev, Dragomir Anguelov, Scalable Object Detection using Deep Neural Networks, arXiv preprint arXiv: arXiv:1312.4400, 2014.
4. E. Borenstein; J. Malik, "Shape Guided Object Segmentation" in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.
5. Geoffrey E. Hinton, Vinod Nair. "Rectified Linear Units Improve Restricted Boltzmann Machines", Proceedings of International Conference on Machine Learning (ICML), 2010, pp 807-814.
6. H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5325-5334.
7. Hubel, D. H., & Wiesel, T N. (1979). Brain mechanisms of vision. Scientific American, 241, 150–162.
8. J. Yan, Z. Lei, L. Wen, and S. Li, "The fastest deformable part model for object detection," in IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2497-2504.
9. Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, How transferable are features in deep neural networks?, arXiv preprint arXiv: arXiv: 1411.1792v1, 2014.
10. Min Lin, Qiang Chen, Shuicheng Yan, Network In Network, arXiv preprint arXiv: arXiv: 1312.2249, 2013.
11. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research 15, 2014, pp. 1929-1958.

KUCHER Oleksandr,
PhD-student, The Bohdan Khmelnytsky National University of Cherkasy

FEATURES OF APPLICATION OF CONVERSIONAL NEURAL NETWORKS FOR IMAGE PROCESSING PROBLEMS

Summary. Introduction. *Convolutional Neural Networks (CNN) are one of the most influential innovations in the field of computer vision. For the first time neural networks attracted attention in 2012 at the ImageNet competition. With the help of CNN, a new record of classification errors was set*

- 15% (the previous value was 26%). Today, many well-known companies use deep learning. But the classic and most popular way to use neural networks is through image processing. Consider how CNNs are used to classify images.

Purpose. *The task of image classification is to process the image and determine the class to which it belongs (car, animal, etc.) or the group of classes that best characterize it.*

Results. *The article gives the basic concept and characteristics of convolutional neural networks, describes the structure and mathematical approaches to the implementation of this type of neural network. Provided description of the main layers of convolutional neural networks, mentioned parameters that allow changing the process of training the network. The process of training each of the neural network's layer is considered in detail. In addition, described techniques that allow you to increase the data set and improve the training process of the network only by several transformations.*

Conclusion. *The concept of convolutional neural networks, their connection with biology and structure has been considered in the article. The main layers that are found in CNN are discussed in detail, described their work through mathematics. Each of the layers is considered individually and in conjunction with the other layers of the neural network.*

Described how neural networks work and how to train them properly. Several approaches to learning neural networks are discussed, as well as simple ways to solve common learning problems: increasing the amount of data you need to train and using already trained networks as a base for solving specialized tasks. Here are some options for changing your network's learning process.

The article discusses some of the most common tasks that are solved using convolutional neural networks: classification, localization, detection, and segmentation.

Keywords: *deep learning, machine vision, neural networks, face recognition, convolutional neural networks, receptive fields, deep learning, neural networks, convolutional neural networks, classification, localization, detection, segmentation, rectified linear units.*

Одержано редакцією 11.04.2018 р.
Прийнято до публікації 19.09.2018 р.

УДК 519.688

DOI 10.31651/2076-5886-2019-1-75-85

PACS 02.60.-x, 02.60.Pn

КУЦЕНКО Олександр Анатолійович
студент спеціальності «Прикладна математика» Черкаського національного університету імені Богдана Хмельницького
e-mail: alexkutsenko1995@gmail.com
ORCID 0000-0002-6220-6034

СЕРДЮК Олександр Анатолійович
кандидат економічних наук, старший викладач кафедри прикладної математики та інформатики Черкаського національного університету ім. Б. Хмельницького
e-mail: serdyuk4labs@ukr.net
ORCID 0000-0002-3919-4661

ОГЛЯД АЛГОРИТМІВ ПОШУКУ ПЛАГІАТУ У ПРОГРАМНОМУ КОДІ

У статті розглянуто поняття плагіату, його класифікацію та загальновживані алгоритми пошуку плагіату, а саме: метод ідентифікаційних міток, алгоритм Хескела, метод вирівнювання рядків та метод жадібного рядкового заміщення. З розвитком технологій та можливістю використання інтернету студент, не прикладаючи багато зусиль, може видавати матеріали іншої персони за свої. Тому дані методи та алгоритми часто