УДК 004.021

DOI: 10.31673/2412-9070.2020.062025

Б. В. ШЕФКІН, студент;
І. В. КРАСЮК, студент;
В. О. ХОМЕНЧУК, аспірант;
К. П. СТОРЧАК, доктор техн. наук, професор;
А. М. ТУШИЧ, ст. викладач,
Державний університет телекомунікацій, Київ

ДОСЛІДЖЕННЯ ТА ВПРОВАДЖЕННЯ НЕЙРОННОЇ МЕРЕЖІ НА ОСНОВІ TensorFlow

TensorFlow — це механізм машинного навчання та глибокого навчання з відкритим кодом, який є зручним та гнучким для побудови поточної загальноприйнятої моделі глибинного навчання. Нейронна мережа — це класична модель глибинного навчання, перевага якої полягає у її потужних можливостях вилучення конволюційних блоків. Нейронна мережа в найпростішому випадку — математична модель, яка складається з кількох шарів елементів, що виконують паралельні обчислення. Спочатку таку архітектуру було створено за аналогією з дрібними обчислювальними елементами людського мозку — нейронами. Мінімальні обчислювальні елементи штучної нейронної мережі теж називаються нейронами. Нейронні мережі, зазвичай, складаються з трьох або більше шарів: вхідного шару, прихованого шару (або шарів) і вихідного шару. Важливою особливістю нейронної мережі є її вміння навчатися на прикладах, це називається навчанням з учителем. Нейронна мережа навчається на великій кількості прикладів, що складаються з пар вхід-вихід (відповідні один одному вхід і вихід). У задачах розпізнавання об'єктів такою парою буде вхідне зображення і відповідний йому лейбл — назва об'єкта. Навчання нейронної мережі — ітеративний процес, що зменшує відхилення виходу мережі від заданого («відповіді вчителя») — лейбла, яке відповідає даному зображенню. Цей процес охоплює кроки, названі епохами навчання (вони зазвичай обчислюються тисячами), на кожному з яких відбувається підгонка «ваг» нейронної мережі — параметрів прихованих шарів мережі. Після завершення процесу навчання якість роботи нейронної мережі переважно досить гарна для виконання завдання, під яке її було навчено, хоча оптимальний набір параметрів, котрі ідеально розпізнають усі зображення, часто підібрати неможливо. На основі платформи TensorFlow було побудовано модель нейронної мережі з двома згортковими шарами. Модель пройшла навчання та тестування за допомогою набору даних MNIST. Показник точності тесту може досягати 99,15% і порівняно з коефіцієнтом 98,69% у моделі з однією згорткою шару показує, що модель нейронної мережі з двома згортаннями має кращу здатність щодо виокремлення ознак і класифікації прийняття рішень.

Ключові слова: нейронна мережа; глибинне навчання; згорткові шари; TensorFlow.

ВСТУП

На хвилі штучного інтелекту стрімко поширилася технологічна революція, зумовлена глибинним навчанням. Як галузь машинного навчання глибинне навчання може не тільки вивчати взаємозв'язок між функціями та завданнями, а й автоматично витягувати більш складні функції з простих зразків. Машинне навчання вказує шлях до штучного інтелекту, тоді як глибинне навчання робить машинне навчання справжньою реалізацією [1]. Протягом останніх кількох років глибинне навчання поширилося на різні сфери машинного навчання, привнісши чималу зручність у життя людей.

Нині до основних framework із відкритим кодом для глибинного навчання належать: TensorFlow від Google, CNTK від Microsoft, PaddlePaddle від Baidu, Caffe в Каліфорнійському університеті, Theano з Монреальського університету, Torch від Facebook тощо. TensorFlow — офіційно відкрите джерело з листопада 2015 року. Це система глибинного навчання нового покоління. На противагу першому поколінню DistBelief він має вищу швидкість обчислення, більшу кількість підтримуваних комп'ютерних платформ та алгоритмів глибинного навчання [2]. А отже, стабільність системи також вища, тому вона дуже популярна і застосовується користувачами. TensorFlow — це відносно високий рівень серед бібліотек машинного навчання з підтримуваними мовами C++ та Python. Користувачам не потрібно писати складний код для побудови нейромережної структури. Алгоритми та функції оптимізації, які будуть використані в процесі моделювання, можуть просто викликати відповідні функції в бібліотеці TensorFlow, що значно зменшує поріг глибинного навчання. Завдяки швидкому розвитку комп'ютерних технологій та істотному вдосконаленню обчислювальних потужностей звичайні комп'ютери також можуть створювати моделі глибинного навчання на платформі TensorFlow, зменшуючи вартість розробки та полегшуючи перевірку алгоритмів [3].

ОСНОВНА ЧАСТИНА

Модель глибинного навчання

Принцип NN. Згорткова нейронна мережа (CNN) — це глибинна нейронна мережа. Основну ідею CNN можна узагальнити двома параметрами: розрідженістю зв'язку та спільними вагами. Типова згорткова нейронна мережа складається з вхідного шару, згорткового шару, шару об'єднання, повністю

© Б. В. Шефкін, І. В. Красюк, В. О. Хоменчук, К. П. Сторчак, А. М. Тушич, 2020

зв'язаного шару та вихідного шару. Переважно вона охоплює два процеси: пряме і зворотне поширення. Перший видає результат прогнозування через мережну структуру, а другий виконує коригування параметрів відповідно до різниці між результатом прогнозування та фактичним значенням. Вхідний рівень — це дані зображення, які потрібно ввести, і, як правило, це матриця $N \times r \times r$. Згортковий шар — це фільтр фіксованого розміру (ядро згортки), згорнутий із зображенням попереднього шару для вилучення власних значень на зображенні. Принцип його дії можна подати у вигляді

$$x_j^l = f(\sum_{i \in S_j} x_i^{l-1} * \omega_{ij}^l + b_j^l), \quad (1)$$

де S_j — набір вхідних зображень; x_i^{l-1} — карта функцій i-го входу $(l-1)$-го шару; ω_{ij}^l — ядро згортки з i-ї карти вхідних характеристик; $(l-1)$-й шар до j-ї карти особливостей виходу l-го шару; b_j^l — значення зміщення, що відповідає j-му виходу карти особливостей l-го шару; $f(\cdot)$ — функція активації; * — операція згортки; результат — j-та карта x_j^l особливостей l-го шару.

Об'єднувальний шар, який також називають знижувальним шаром, це спеціальний згортковий шар. Зазвичай він має дві форми: максимальне об'єднання та середнє об'єднання. Принцип подібний до рівняння (1), за винятком того, що функція активації стає функцією об'єднання. Метою трансформації є зменшення розміру карти об'єктів без зміни кількості карт об'єктів, а лише модифікуючи її розмір, тим самим зменшуючи параметри в повністю зв'язаному шарі та пришвидшуючи обчислення.

Повністю зв'язаний шар поєднано зі згортковим шаром і звичайним шаром. Проміжок від початку введення до повністю зв'язаного шару називається згортковими шарами з «блоком згортки»; а від повністю зв'язаного шару до вихідного — це всі звичайні шари кінцевого «блоку NN». Повністю зв'язаний шар витягує високовимірні дані, отримані з батьківського шару (шару згортки), як вхідні дані в плоскій формі, подібній до тієї, що зображено на рис. 1; виконує деяке нелінійне перетворення (функцію активації), а потім уводить результат у наступну систему звичайних шарів. Подальші вирішення (прогнозування та класифікація) приймаються в цих шарах. Тому «блок згортки» стає екстрактором ознак, а «блок NN» називається класифікатором прийняття рішень.

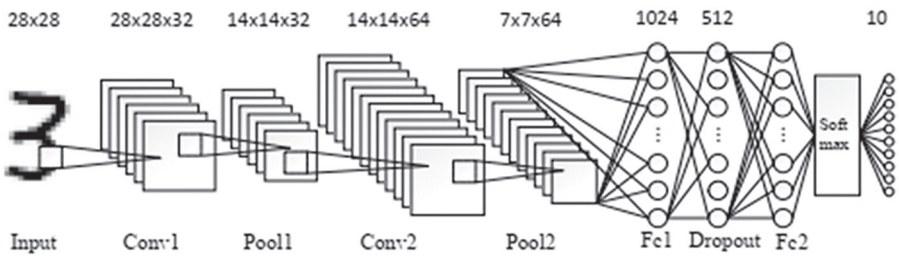

Рис. 1. Схематична діаграма згорткової нейронної мережі

Функція збитків є відображенням ступеня підгонки моделі, а функція перехресної ентропії може ідеально розв'язати проблему зникнення градієнта. Принцип їх дії можна діставати відповідно з рівнянь [4]:

$$L(y,G(x)) = -[y \ln G(x) + (1-y)\ln(1-G(x))], \quad G(x) = (v_1,\ldots,v_k)^T, \quad \sum_1^k v_k = 1. \quad (2)$$

Метод оптимізації зворотного поширення згорткової нейронної мережі приймає адаптивний алгоритм швидкості навчання Адама, який є розширенням стохастичного алгоритму градієнтного спуску і є найбільш широко використовуваним та загалом найкращим алгоритмом у програмах глибинного навчання. Помилка зворотно поширюється алгоритмом Адама, а значення параметрів шарів згорткової нейронної мережі поступово оновлюються [5]:

$$\nabla \Leftarrow \beta_1 \nabla + (1-\beta_1)\nabla w_t, \quad \nabla^2 \Leftarrow \beta_2 \nabla^2 + (1-\beta_2)\nabla^2 w_t, \quad \nabla * w_t = \nabla/(\nabla + \eta). \quad (3)$$

Модель класифікації NN. Схема структурної класифікаційної моделі згорткової нейронної мережі (див. рис. 1) загалом охоплює вхідний шар, шар із двома згортками, шар із двома пулами, повністю зв'язаний шар, шар, що випадає, Softmax та вихідний шар. Рівень згортки обертає дані верхнього шару через ядро згортки, а потім використовує функцію активації для отримання карти ознак рівня згортки; шар об'єднання встановлює максимальне значення через кожну сусідню ділянку 2×2 на карті об'єктів верхнього шару так, що розмір карти об'єктів після об'єднання становить половину верхнього шару; повністю зв'язаний шар має викласти 64 карти особливостей другого шару об'єднання у вектор; шар, що випадає, призначено для запобігання переобладнанню та покращенню здатності узагальнення даної моделі. Нарешті, дані класифікуються за допомогою регресійної моделі softmax для виведення категорії.

Дослідження та впровадження на TensorFlow із використанням набору даних MNIST

TensorFlow і TensorBoard. У TensorFlow є такі моделі обчислення: графік розрахунку та модель даних (тензор, запущена модель, сесія). Дані в TensorFlow подано структурою даних Tensor, Flow — потік і розрахунок даних. Він може використовувати (або отримувати) стрічку для призначення (або отримання) даних у тензорі.

TensorFlow — це система глибокого навчання, яка подає розрахунок у вигляді обчислювального графіка. Схему розроблення програми TensorFlow зображено на рис. 2 [6], її зазвичай можна поділити на два етапи: побудову та графік виконання.

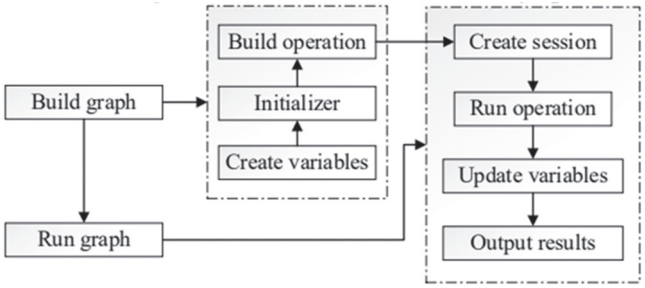

Рис. 2. Схема розроблення програми TensorFlow

MNIST набір даних. У наборі даних розпізнавання розрядів цифр MNIST є два типи зображень: перший — це 60 000 наборів навчальних зразків (55 000 навчальних зразків та ярликів, 5000 зразків та етикеток, що підтверджують навчання), а другий — 10 000 тестових зразків та позначок. Кожну вибірку подано матрицею розміром 28×28, яку викладено в 724-вимірний рядок даних. Значення позначки — це 10-вимірний вектор, який є одноразовим поданням номера категорії 0-9. Виокремлення ознак та прийняття рішень щодо класифікації є фокусом розпізнавання від руки цифр [7].

Етапи реалізації програми. Програму написано на Anaconda, найпопулярнішій платформі науки про дані Python з відкритим кодом, а середовищем розроблення Python є Spyder (наукове середовище розроблення Python). Схему структури моделі глибокого навчання унаочнює рис. 3.

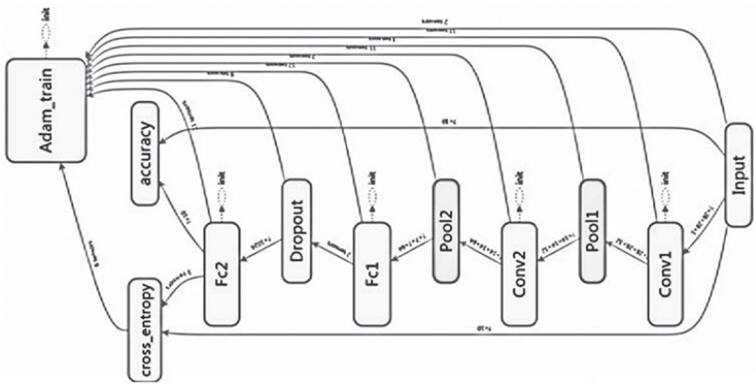

Рис. 3. Схема структури моделі глибокого навчання

Це обчислювальний графік всієї структури моделі, відображений TensorBoard, який включає в себе два згорткові шари Conv1 і Conv2, два шари об'єднання Pool1 і Pool2, один повністю зв'язаний шар Fc1, один вихідний шар Fc2 і доданий вихід між Fc1 і Fc2 шарами для запобігання перенавчання даних. Точність моделі забезпечується використанням тестового зразку, щоб передбачити результат і порівняти зі значенням етикетки; cross_entropy є функцією перехресної ентропії, яка вказує на помилку між результатом прогнозування моделі та значенням позначки; Adam_train — це алгоритм навчання оптимізації.

Вхідний шар, ініціалізування ваги і значення зміщення. Два заповнювачі x та y визначаються як вхідні дані навчального та тестових наборів, а дані 784-виміру перетворюються у матричне зображення 28×28. Вага ініціалізується за допомогою зрізаного нормального розподілу, а значення зміщення ініціалізується до константи 0,1.

Шар згортки та об'єднання. У процесі визначення операції згортки розмір кроку ядра згортки дорівнює одиниці як у напрямку x та y; заповнення = ' this'. Під час визначення операції об'єднання розмір вікна об'єднання становить 2×2. Значення ваги першого рівня згортки ініціалізуються, розмір вікна вибірки становить 5×5; 32 ядра згортки витягують елементи з однієї площини, і кожне ядро згортки має значення зміщення. Далі складається вхідне зображення *x_image* з вектором ваги, додається значення зміщення, а потім застосовується до функції активації *relu*. Нарешті максимізується об'єднання виходу, аналогічно форматуванню ваги і зміщенню значення другого шару, за винятком того, що 64 ядра згортки функції вилучено з 32 площини.

Повністю зв'язаний шар. Із попередньої процедури випливає, що початкове зображення розміром 28×28 все ще залишається 28×28 після першої згортки через одне таке подання. Карта функцій стає

14×14 після першого об'єднання через згруповане вікно 2×2 і розмір кроку 2; так само після другої згортки вона стає 14×14, а після другого об'єднання — 7×7. Нарешті, після зазначеної операції дістаємо площини $64 \times 7 \times 7$, що використовується як вхід для наступного повністю зв'язаного шару. Ініціалізуємо вагу повністю зв'язаного шару, визначаємо 1024 нейрони, 1024 значення зміщення та порівнюємо 64 карти об'єктів з $64 \times 7 \times 7$-вимірними даними, а потім помножуємо їх на вектор ваги плюс значення зміщення. Далі застосовується до функції активації *relu*, щоб знайти вихід повністю зв'язаного шару.

Випадальний шар. Виконується операція відсіву для запобігання перенапруженню даних і використовується *keep_prob*, щоб вказати вихідну ймовірність нейрона та функції *tf.nn.dropout ( )* для досягнення відсіву.

Вихідний рівень Softmax. Ініціалізується другий повністю зв'язаний шар, визначаються 10 вихідних нейронів і вихід кінцевого результату класифікації з використанням функції моделі регресії *tf.nn.softmax ( )*.

Функція втрат та навчання оптимізації. Вибираємо функцію перехресних ентропійних втрат: *tf.nn.softmax_cross_entropy_with_logits ( )*, потім зменшуємо та мінімізуємо втрати після побудови втрати з використанням алгоритму функції *tf.train.AdamOptimizer ( )* для навчання оптимізації.

Точність. Результати класифікації зберігаємо в логічному списку та застосовуємо функцію *argmax( )* для повернення позиції найбільшого значення в тензорі, а також функцію порівняння *euqal( )*, щоб визначити, чи правильний результат, і, нарешті, дістаємо точність класифікації.

Виконання сесії. Завдяки величезним навчальним даним ідеї групового навчання використовуються для створення 100 навчальних схем на партію. Після навчання 550 разів усі дані зразків можуть бути навчені. Створюються сеанси, ініціалізуються змінні, подаються навчальні дані та тренується модель 10 000 кроків, реєструються параметри та точність тестового набору на 100 кроків.

Аналіз результатів. Результат підсумкового тесту після 10 000 тренувань унаочнює рис. 4. Ліворуч зображено двошарову мережу з точністю 99,15%, що на 0,46-відсоткового пункту вище, ніж точність 98,69% одношарової. Криву залежності між часом навчання та точністю моделі CNN, яку отримано за допомогою інструменту візуалізації TensorBoard, де абсциса — це кількість тренувань, а ордината — точність розпізнавання, демонструє рис. 5. З рис. 5 видно, що зі збільшенням кількості тренувань точність поступово зростає і нарешті стабілізується. У двошаровій згортковій нейронній мережі рівень точності досягав 90% під час навчання в 200-300 разів; у разі навчання до 1100 разів даних рівень точності становив 97%; при навчанні до 6000 разів точність загалом була стабільною — майже 99%. На рис. 5 добре видно, що крива точності шарів із двома згортаннями, як правило, вища, ніж крива точності з одною згорткою, а похибка двох моделей зменшується з кількістю тренувань; також крива помилок нейронної мережі з двома згортаннями шарів значно нижча за похибку з одною згорткою. Зазначені дані показують, що модель нейронної мережі з двома згортаннями має вищу точність розпізнавання, ніж шар із однією згорткою, швидша конвергенція, сильніша здатність до виокремлення ознак та прийняття рішень щодо класифікації.

```
Step 9000, Testing Accuracy= 0.9904,
Step 9100, Testing Accuracy= 0.9909,
Step 9200, Testing Accuracy= 0.9908,
Step 9300, Testing Accuracy= 0.9909,
Step 9400, Testing Accuracy= 0.9909,
Step 9500, Testing Accuracy= 0.9907,
Step 9600, Testing Accuracy= 0.9904,
Step 9700, Testing Accuracy= 0.9911,
Step 9800, Testing Accuracy= 0.9912,
Step 9900, Testing Accuracy= 0.9912,
Step 10000, Testing Accuracy= 0.9915,
```

```
Step 9000, Testing Accuracy= 0.986,
Step 9100, Testing Accuracy= 0.9847,
Step 9200, Testing Accuracy= 0.9862,
Step 9300, Testing Accuracy= 0.9869,
Step 9400, Testing Accuracy= 0.9867,
Step 9500, Testing Accuracy= 0.9867,
Step 9600, Testing Accuracy= 0.9858,
Step 9700, Testing Accuracy= 0.9856,
Step 9800, Testing Accuracy= 0.9875,
Step 9900, Testing Accuracy= 0.9867,
Step 10000, Testing Accuracy= 0.9869,
```

Рис. 4. Результат тестів (ліворуч — два шари, праворуч — один)

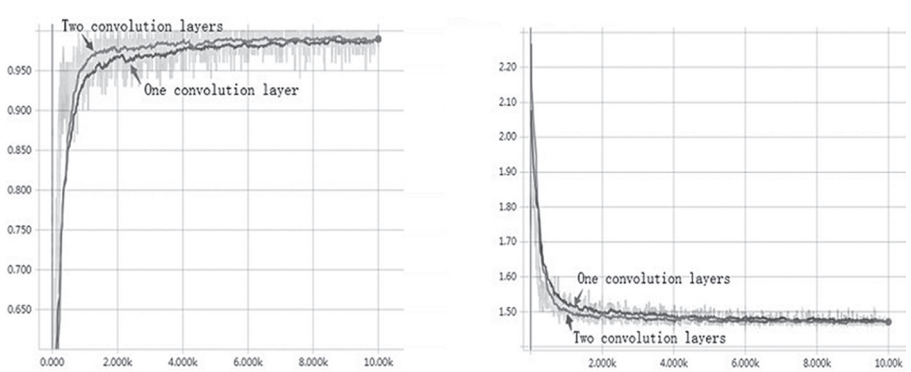

Рис. 5. Діаграма взаємозв'язку між часом навчання, точністю та помилками

Проекти на TensorFlow

TensorFlow — чудовий інструмент, який за правильного застосування має незліченні переваги. Основне використання бібліотеки включає в себе класифікацію, сприйняття, розуміння, виявлення, прогнозування та створення. Розглянемо деякі приклади використання даної бібліотеки в проектуванні та розробленні продуктів у різних сферах діяльності.

Одним із яскравих прикладів розроблення на TF є нейронна мережа для розпізнавання обличчя людини, інтегрована в мікроконтролер. Дана система має назву *DoorBellCam*. Вона ідентифікує у кадрі обличчя людини та надсилає скріншот на електронну скриньку власнику. Така система ідентифікує обличчя за допомогою нейронної мережі на базі TF, яка аналізує кадр у режимі реального часу та зіставляє картинку із зображеннями, за допомогою яких мережа навчалася.

Існує велика кількість проектів для оброблення даних різного типу в інтернеті. Далі наведемо приклади деяких систем.

RankBrain, розроблений Google — це масштабне розміщення глибоких нейронних мереж для ранжування пошуку на www.google.com. Це частина алгоритму пошуку, яка використовується для сортування мільярдів сторінок, про які вона знає, і знаходить ті, котрі вважаються найбільш релевантними. Це також найбільш обговорювана програма TensorFlow.

Inception Image Classifier, розроблений Google, є базовою моделлю та подальшим дослідженням дуже точних моделей комп'ютерного зору, починаючи з моделі, яка перемогла у класифікації зображень Imagenet 2014 року та започаткувала еру конволюційних мереж.

Massive multitask для виявлення наркотиків, Стенфордський університет, є глибинною моделлю нейронних мереж для визначення найбільш вірогідних осіб, які пов'язані з наркотиками.

Пристрій Computer Vision для OCR — це вбудована модель комп'ютерного зору для оптичного розпізнавання символів, що дає можливість здійснювати переклад у реальному часі.

Використовуючи TensorFlow, дана система може створювати алгоритми для формування зображення або візуалізації об'єктів на фотографії; алгоритм також може навчити ПК розпізнавати об'єкти на зображенні та використовувати ці дані для створення нових та цікавих форм поведінки, починаючи з розуміння подібності та відмінності великих даних; використовує їх для самоорганізації, для розуміння того, як нескінченно генерувати абсолютно новий вміст або відповідати естетиці інших зображень. Ми навіть можемо навчити комп'ютер читати та синтезувати нові фрази, які є частиною оброблення природної мови.

TensorFlow також можна використовувати з інструментами контейнеризації, такими як docker, наприклад, він може бути застосовний для розгортання моделі аналізу настроїв, яка використовує мережі ConvNet на рівні символів для класифікації тексту.

Більше того, НАСА розробляє систему з TensorFlow для класифікації орбіт та кластеризації об'єктів астероїдів, а також класифікує та передбачає близькоземні об'єкти. Отже, ця бібліотека, безумовно, пришвидшує навчання, надаючи інструменти, яких завжди бракувало.

Також TensorFlow є актуальним для проектування систем моделювання розмови в розумінні природної мови і машинного інтелекту. На жаль, сучасні віртуальні співрозмовники лише частково вирішують питання імітації розмови людини. Основу їх функціонування складає база знань. У найпростішому випадку вона містить набори можливих питань користувача і відповідних відповідей на них.

TensorFlow є глибоким вивченням системного програмного забезпечення. TensorFlow добре працює для пошуку інформації, як показано в Google в тому, як побудовано рейтинг пошуку в їх машинному навчанні системи штучного інтелекту, RankBrain. TensorFlow може виконувати розпізнавання образів, як показано в Google, Inception, а також звукового розпізнавання людської мови. Це також корисно в розв'язанні інших проблем, не характерних для машинного навчання, таких як диференційні рівняння в приватних.

ВИСНОВКИ

Запропонована стаття описує алгоритм роботи нейронної мережі на базі бібліотеки TensorFlow з відкритим кодом для створення моделі глибинного навчання CNN. Набір даних MNIST використовується як приклад для навчання та тестування моделі. За допомогою візуального інструменту TensorBoard у статті відображається структура моделі глибинного навчання, результати випробувань, а криві тенденції перевіряють валідність моделі. На противагу класичній згортковій нейронній мережі додається шар операцій згортки та об'єднання, а друга операція об'єднання згорток зручна для більш глибоких рівнів, щоб виокремити більшу кількість характеристик, що, нарешті, досягає точності розпізнавання 99,15%.

Список використаної літератури

1. ***Zheng Z.*** *TensorFlow actual Google deep learning framework // Electronic Industry Press. 2018. 287 p.*
2. ***Jing T., Zhang Y.*** *Digital recognition based on deep learning under the TensorFlow platform. 2018. 37(04). 78 p.*
3. ***Xiao J.*** *The application prospect of clustering algorithm in TensorFlow platform is discussed // Digital technology and applications. 2018. №36. P. 203.*
4. ***Goodfellow I., Bengio Y., Aaron Courville A.*** *DEEP LEARNING // Posts and Telecom Press. 2018. P. 203.*
5. ***He Y.*** *Python works with machine learning // Electronic Industry Press. 2017. №200. P. 222.*
6. ***Han S., Tan S.*** *Design and implementation of a deep learning model for stock forecasting based on TensorFlow // Computer applications and software. 2018. №35. P. 291.*
7. ***Xu Y.*** *Application of deep learning in handwritten numeral recognition // Suzhou University. 2017. №255. P. 136.*
8. ***Xu Y., Li Z.*** *Handwritten numeral recognition in convolutional neural network and TensorFlow // Shanghai electric technology. 2018. №11. P. 61.*
9. ***Handwritten*** *numeral recognition system based on TensorFlow / H. Chen, H. Guo, D. Liu, J. Zhang. // Information communication. 2018. №3. P. 110.*
10. ***Zheng P., Guo L., Ding L.*** *Research and implementation of convolutional neural network based on TensorFlow // Electronic technology and software engineering. 2018. №18. P. 22.*
11. ***Jin Z.*** *Comparison and analysis of different deep convolutional neural networks based on TensorFlow // Electronics World. 2018. №6. P. 26.*
12. ***Hou Y., Ding S., Sun T.*** *The mixed depth learning model C-RF and its application in handwritten numeral recognition // Data acquisition and processing. 2018. №33. P. 350.*
13. ***Ertam F., Aydin G.*** *Data classification with deep learning using Tensorflow // International Conference on Computer Science and Engineering (UBMK) Antalya, 2017. 758 p.*
14. ***Analogizing*** *time complexity of KNN and CNN in recognizing handwritten digits / T. Makkar, Y. Kumar, A. K. Dubey, A. Rocha // Fourth International Conference on Image Information Processing (ICIIP) Shimla, 2017. 1256 p.*
15. ***Alif M., Ahmed S., Hasan M.*** *Isolated Bangla handwritten character recognition with convolutional neural network // 20th International Conference of Computer and Information Technology (ICCIT) Dhaka, 2017. 523 p.*
16. ***Iamsa S., Horata P.*** *Handwritten Character Recognition Using Histograms of Oriented Gradient Features in Deep Learning of Artificial Neural Network // International Conference on IT Convergence and Security (ICITCS) Macao, 2013. 312 p.*

Б. В. Шефкин, И. В. Красюк, В. О. Хоменчук, К. П. Сторчак, А. Н. Тушич

ИССЛЕДОВАНИЕ И ВНЕДРЕНИЕ НЕЙРОННОЙ СЕТИ НА ОСНОВЕ TENSORFLOW

TensorFlow — это механизм машинного обучения и глубокого обучения с открытым кодом, который является удобным и гибким для построения текущей общепринятой модели глубинного обучения. Нейронная сеть — это классическая модель глубинного обучения, преимущество которой заключается в ее мощных возможностях изъятия конволюционных блоков. Нейронная сеть в простейшем случае — математическая модель, состоящая из нескольких слоев элементов, выполняющих параллельные вычисления. Сначала такая архитектура была создана по аналогии с мелкими вычислительными элементами человеческого мозга — нейронами. Минимальные вычислительные элементы искусственной нейронной сети тоже называются нейронами. Нейронные сети, как правило, состоят из трех или более слоев: входного слоя, скрытого слоя (или слоев) и выходного слоя. Важной особенностью нейронной сети является ее умение учиться на примерах, это называется обучением с учителем. Нейронная сеть обучается на большом количестве примеров, состоящих из пар вход-выход (соответствующие друг другу вход и выход). В задачах распознавания объектов такой парой будет входное изображение и соответствующий ему лейбл — название объекта. Обучение нейронной сети — итеративный процесс, который уменьшает отклонения выхода сети от заданного («ответы учителя») — лейбла, соответствующего данному изображению. Этот процесс состоит из шагов, которые называются эпохами обучения (они обычно исчисляются тысячами), на каждом из которых происходит подгонка «весов» нейронной сети — параметров скрытых слоев сети. По завершении процесса обучения качество работы нейронной сети обычно достаточно хорошее для выполнения задачи, под которую она была обучена, хотя оптимальный набор параметров, которые идеально распознают все изображения, часто подобрать невозможно. На основе платформы TensorFlow была построена модель нейронной сети с двумя сверточными слоями. Модель прошла обучение и тестирование с помощью набора данных MNIST. Показатель точности теста может достигать 99,15% и по сравнению с коэффициентом 98,69% у модели с одной сверткой слоя показывает, что модель нейронной сети с двойным свертыванием имеет лучшую способность к изъятию признаков и классификации принятия решений.

Ключевые слова: нейронные сети; глубинное обучение; сверточные слои; Tensorflow.

B. V. Shefkin, I. V. Krasiuk, V. O. Khomenchuk, K. P. Storchak, A. M. Tushych

RESEARCH AND IMPLEMENTATION OF NN BASED ON TENSORFLOW

TensorFlow is Google's open-source machine learning and deep learning framework, which is convenient and flexible to build the current mainstream deep learning model. Convolutional neural network is a classical model of deep learning, the advantage lies in its powerful feature extraction capabilities of convolutional blocks. A neural network in the simplest case is a mathematical model consisting of several layers of elements that perform parallel calculations. Initially, such an architecture was created by analogy with the small computing elements of the human brain — neurons. The minimal computing elements of an artificial neural network are also called neurons. Neural networks typically consist of three or more layers: an input layer, a hidden layer (or layers), and an output layer. An important feature of the neural network is its ability to learn by example, this is called learning with a teacher. The neural network is trained on a large number of examples consisting of input-output pairs (corresponding to each other input and output). In object recognition problems, such a pair will be the input image and the corresponding label — the name of the object. Neural network learning is an iterative process that reduces the deviation of the network output from a given «teacher response» — a label that corresponds to a given image. This process consists of steps called epochs of learning (they are usually calculated in thousands), each of which is the adjustment of the «weights» of the neural network — the parameters of the hidden layers of the network. Upon completion of the learning process, the quality of the neural network is usually good enough to perform the task for which it was trained, although the optimal set of parameters that perfectly recognizes all the images, it is often impossible to choose. Based on the TensorFlow platform, a convolutional neural network model with two-convolution-layers was built. The model was trained and tested with the MNIST data set. The test accuracy rate could reach 99,15%, and compared with the rate of 98,69% with only one-convolution-layer model, which shows that the two-convolution-layers convolutional neural network model has a better ability of feature extraction and classification decision-making.

Keywords: neural network; deep learning; convolution layers; TensorFlow.

УДК 004.432.2

DOI: 10.31673/2412-9070.2020.062628

Д. В. КРАВЕЦЬ, асистент;
А. М. ТУШИЧ, ст. викладач;
В. В. ШКАПА, доцент;
В. Р. МИКОЛАЙЧУК, ст. викладач,
Державний університет телекомунікацій, Київ

ОГЛЯД СТАНДАРТНОЇ БІБЛІОТЕКИ ДЛЯ СТВОРЕННЯ GUI МОВОЮ Python

Розглянуто стандартну бібліотеку для створення графічних інтерфейсів мовою Python. Було проаналізовано актуальність засобів створення графічних інтерфейсів, значення графічного інтерфейсу у сприйнятті користувачем програми, останні дослідження і публікації, що пов'язані з бібліотекою Tkinter. Досліджено історію розвитку бібліотеки, основні віджети та менеджери геометрії. Визначено всі функції, що відповідають за розміщення елементів графічного інтерфейсу у вікні та зроблено висновки щодо недоліків і переваг даної бібліотеки.

Ключові слова: програмування; мова програмування; графічний інтерфейс; інтерфейс; вікно; віджет; бібліотека; менеджер геометрії; клас; функція; Python; Tkinter; GUI; Tk.

ВСТУП

Постановка проблеми. На сучасному етапі розвитку інформаційних технологій пересічний користувач звик до графічних інтерфейсів і може мати складнощі у процесі роботи із суто текстовими інтерфейсами. Через це важливо знати не лише методи реалізації логіки програм, а й те, як зробити спілкування користувача з програмою максимально комфортним.

Аналіз останніх досліджень і публікацій. Стандартна бібліотека для створення графічних інтерфейсів мовою Python тісно пов'язана з бібліотекою Tk, що відома давно та багато де описана. Наприклад праця [1] досі має попит в інтернет-магазині amazon.com, незважаючи на те, що її було написано 2000 року. Проте час швидкоплинний, як і розвиток мови програмування Python. Говорячи про використання новіших версій бібліотеки Tkinter, добре згадати про підручник [2]. Однак найактуальнішим джерелом інформації завжди залишатиметься документація бібліотеки [3], що викладена на сайті документації мови програмування Python.

Мета статті. Метою статті є огляд стандартної бібліотеки для створення графічних інтерфейсів мовою Python.

© Д. В. Кравець, А. М. Тушич, В. В. Шкапа, В. Р. Миколайчук, 2020