УДК 004.896
В.М. Мельник, К.В. Мельник, Б.В. Шульга
Луцький національний технічний університет

ДОСЛІДЖЕННЯ МОДЕЛЮВАННЯ ІДЕНТИФІКАТОРА ЕМОЦІЙ ЛЮДИНИ ЗА ДОПОМОГОЮ ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ З ВИКОРИСТАННЯМ KERAS

Мельник В.М., Мельник К.В., Шульга Б.В. Ідентифікація емоцій людини за допомогою нейронної мережі на основі Keras та TensorFlow. В даній статті наведено результати досліджень визначення емоцій людини за допомогою нейронних мереж. Розробка моделі для аналізу зображень проводилась за допомогою TensorFlow, а тренування реалізовувалось з використанням Keras. Вхідні дані використано з архіву kaggle.com - FER2013. Для аналізу зображеннь використано бібліотеку OpenCV. Мова програмування – Python 3.

Даний набір інструментів вважається найпопулярнішим і найзручнішим для побудови нейронних мереж, а також систем глибинного навчання. Нейронні мережі і машинне навчання - найпопулярніші технології на даний момент. Особливо великих результатів можна досягнути поєднуючи цю технологію з іншими відомими – наприклад, з технологією об'єктно-орієнтованого програмування. Це поєднання технологій має широкий спектр застосування в різних областях, починаючи від звичайних фотосвітлин викладених в соціальних мережах, і закінчуючи контролем поведінки громадян держави або навіть планети. Аналіз емоцій дає можливість продуктовим і рекламним компаніям значно збільшити об'єм продаж, що в свою чергу збільшить прибутки [12]. Бути геніальним співбесідником, маючи можливість маніпулювати людьми знаючи що вони думають, проводити стрес-тести співробітників та оцінювати їхню реакцію, визначати реакцію людини на рекламу, оголошення, промову збирати обробляти і робити висновки. Список сфер використання обмежений лише фантазією і очікуємим результатом, тому тема цієї наукової роботи є актуальною.

Ключові слова: моделювання, ідентифікація, емоції людини, згорткова нейронна мережа, KERAS

Мельник В.М., Мельник К.В., Шульга Б.В. Идентификация эмоций человека с помощью нейронной сети на основе Keras и TensorFlow. В данной статье приведены результаты исследований определения эмоций человека с помощью нейронных сетей. Разработка модели для анализа изображений проводилась с помощью TensorFlow, а тренировки реализовывалось с использованием Keras. Входные данные использовано из архива kaggle.com - FER2013. Для анализа изображений использовано библиотеку OpenCV. Язык программирования - Python 3.

Данный набор инструментов считается самым популярным и удобным для построения нейронных сетей, а также систем глубинного обучения. Нейронные сети и машинное обучение - самые популярные технологии на данный момент. Особенно больших результатов можно достичь сочетая эту технологию с другими известными - например, с технологией объектно-ориентированного программирования. Это сочетание технологий имеет широкий спектр применения в различных областях, начиная от обычных фотосвитлин изложенных в социальных сетях, и заканчивая контролем поведения граждан государства или даже планеты. Анализ эмоций дает возможность продуктовым и рекламным компаниям значительно увеличить объем продаж, что в свою очередь увеличит доходы [12]. Быть гениальным собеседником, имея возможность манипулировать людьми зная что они думают, проводить стресс-тесты сотрудников и оценивать их реакцию, определять реакцию человека на рекламу, объявления, речь собирать обрабатывать и делать выводы. Список сфер использования ограничен только фантазией и ожидаемый результат, поэтому тема этой научной работы является актуальной.

Ключевые слова: моделирование, идентификация, эмоции человека, згортковая нейронная сеть, KERAS

Melnyk V.M., Melnyk K.V., Shulga B.V. Identification of human emotions using the Keras and TensorFlow neural Network. This article presents the results of research on the determination of human emotions through neural networks. The development of the model for image analysis was carried out using TensorFlow, and the training was implemented using Keras. The input was used from the kaggle.com - FER2013 archive. The OpenCV library was used to analyze the images. Programming language - Python 3.

This toolkit is considered the most popular and most convenient for building neural networks, as well as systems of deep learning. Neural networks and machine learning are the most popular technologies at the moment. Particularly great results can be achieved by combining this technology with other known - for example, technology object-oriented programming. This combination of technologies has a wide range of applications in various fields, ranging from ordinary photo luminaries outlined in social networks, and ending with the control of the behavior of citizens of the state or even the planet. The analysis of emotions allows grocery and advertising companies to significantly increase sales, which in turn will increase profits [12]. Being a brilliant interlocutor, having the ability to manipulate people knowing what they think, conduct stress tests of employees and evaluate their reactions, determine the human reaction to advertising, announcement, speech to collect, process and draw conclusions. The list of areas of use is limited only by fantasy and the expected result, therefore the subject of this scientific work is relevant.

Key words: modeling, identification, human emotions, convolutional neural network, KERAS

Постановка проблеми. Однією з найактуальніших технологій сьогодні виступає навчання, організоване за допомогою нейронних мереж. Практикою використання доведено, що особливо великих результатів було досягнуто в ході поєднання запропонованої технології з іншими відомими технологіями. Наприклад, отримання та документування результатів досліджень за допомогою згорткової мережі досить добре інтегрується з технологією об'єктно-орієнтованого програмування. Таке поєднання має широкий спектр застосування в різних областях, починаючи від звичайних фотосвітлин, викладених в

соціальних мережах, а також в проведенні контролю поведінки людей різного спрямування та професійного профілю в залежності від факторів, які активно впливають на емоції людини в цілому чи на конкретну емоційну характеристику.

Зі сторони ведення бізнесу аналіз емоцій дає можливість продуктовим і рекламним компаніям значно збільшити об'єм товарообороту, що, в свою чергу, веде до збільшення прибутків [12]. Бути геніальним спікером чи співбесідником, маючи можливість підсвідомо впливати та маніпулювати людьми, враховуючи наперед що вони думають, проводити стрес-тести співробітників та оцінювати їхню реакцію, визначати реакцію людини на рекламу оголошення чи промову, збирати й обробляти дані та робити відповідні висновки. Список сфер використання не обмежений фантазією та результатом очікування, тому дана наукова робота на сьогодні є досить актуальною.

Аналіз останніх досліджень. Задіювання інтелектуальних технологій для розв'язання задач класифікації нині позв'язують з застосуванням обчислювального інтелекту, який, в свою чергу, пов'язують з теорією нечітких множин та нейрокомп'ютингом. З літератури відомо, що одним з ефективних методів класифікації об'єктів є кластеризація, яка може вміщувати ієрархічні та нечіткі методи дослідження [30-32]. Наявність кластерів, які найчастіше представляються ієрархічним деревом, дає змогу створити систему правил, на основі якої з уже сформованої множини даних береться знання про певні приховані закономірності, що притаманні об'єкту дослідження. Кожен з таких методів має свої переваги та недоліки. Однак діючі алгоритми досить часто мають прогресивний творчий характер дії, але, як згадано уже у зазначеній літературі, в непоодиноких випадках задачу класифікації розв'язати все-таки вдається. Надзвичайно важливим завданням як для даного кола робіт так і для нашого завдання дослідження виступає визначення системи ознак, які до того ж можуть бути зашумленими, або нечітко визначеними, тобто у вигляді інтервалів, перервностей і т. ін.

В роботі [33], досить спорідненій з даною темою, наведено огляд існуючих підходів до можливості розпізнавання зображень засобами нейронних мереж. Концентрація думки проходить на основних методах, що використовуються в задачі розпізнавання зображень, а також визначено основні етапи самого процесу розпізнавання зображень. В роботі також розкрито особливості використання описаних підходів, переваги та недоліки зазначених методів. Зроблено висновок щодо необхідності подальшого розроблення алгоритмів розпізнавання на основі зазначених методів, що були б простими в реалізації, ефективними, мали низькі обчислювальні затрати під час навчання та високу якість класифікації в реальних практичних завданнях.

В результаті пророблених в роботі досліджень [33] було побудовано декілька різних нейронних мереж на розпізнання зображень та досліджено їхню продуктивність. Автори роботи сподіваються в майбутньому отримані дані проведеного дослідження можна використати для розробки програм машинного розпізнавання тексту, полегшеного введення тексту за допомогою сенсорних технологій. Перспективним є дослідження продуктивності різних топології нейронних мереж для виконання даного завдання. Однак, при всій спорідненості, питання розпізнання емоцій людини зовсім розкривалося.

В роботі [34] досліджувалося розпізнання акустичних коливань у вигляді звуку з використанням згорткових нейронних мереж та підходу transfer learning. Тут в деталях розглянута структура згорткових нейронних мереж та наведені їх особливості, а також викладені біологічні особливості кори головного мозку тварин, що підштовхнули до розвитку саме даний тип нейронних мереж. Також наведена інформація про підхід перенесення навчання в загальному. Окрім цього також викладена логіка вибору методики перенесення навчання в залежності від даних, яка використовується в дослідженнях подібного напрямку найчастіше. Описано особливості набору даних та загальна інформація про вибрану методику перенесення навчання.

Робота [35] демонструє дослідження системи аналізу біомедичних сігналів для контролю стану водіїв також за допомогою згорткових нейронних мереж. В даній роботі також проаналізовано та оцінено роботу побудованої системи на базі нейронних мереж. Зокрема розроблений додаток для ОС Android, який об'єднує в собі функціонал з визначення емоційних станів людини. Тут також розроблений та додатковий функціонал для використання користувачами. Основний функціонал, призначений для визначення, було створено та проведено аналіз якості роботи модуля емоційних станів відповідно до відповідно наведених критеріїв. Результати оцінювання свідчать про те, що система є досить якісною для проведення масштабних тестів, однак в перспективі вимагає удосконалення. Були також визначені недоліки поточної імплементації системи, які мають і вплив на визначення критеріїв емоцій.

Автори роботи [36] направили своє дослідження на розпізнавання мімічних мікровиразів обличчя людини також використовуючи згорткову нейронну мережу як засіб для дослідження. Метою цієї роботи послужило підвищення точності розпізнавання динамічних зображень шляхом комбінування Time Delay Neural Network та Deep Belief Networks, а також реалізація інтелектуальної системи нейромережевого розпізнавання мімічних мікровиразів обличчя людини. В основу дослідження було взято модифікований алгоритм навчання deep belief network та алгоритм time delay neural network. На їх основі в роботі створено інтелектуальну систему нейромережевого розпізнавання динамічних зображень для підвищення точності розпізнавання мімічних мікровиразів обличчя людини, а саме, розпізнавання статичних і динамічних зображень. У двній роботі запропоновано модифікацію алгоритму навчання глибоких нейронних мереж для розпізнавання статичних зображень мімічних мікровиразів обличчя людини, що дозволило покращити точність розпізнавання із 72-79%.

Відзначається, що точність розпізнавання за допомогою Time Delay Neural Network є вищою порівняно із роботою T. McLaughlin – існуючим методом розпізнавання мімічних мікровиразів на статичних зображеннях із тієї самої бази даних. Наступне покращення точності уже отримувалося у комбінації розпізнавання на динамічних зображеннях за допомогою Time Delay Neural Network із розпізнаванням на статичних зображеннях за допомогою глибоких нейромереж. В цьому випадку за допомогою комбінованого підходу точність розпізнавання складає 98 %. Отриманий результат підтверджує актуальність проведених досліджень у порівняні з найкращим із розглянутих методів нейромережевого розпізнавання зображень у статиці, тобто з застосуванням квантових нейромереж, де досягається точність до 96,5 %. Однак пряме порівняння вказаних методів є недоцільним у зв'язку із використанням різних наборів навчальних та тестових даних.

В роботі також зазначається, що використання Time Delay Neural Network для розпізнавання мімічних мікровиразів обличчя людини на динамічних зображеннях дозволяє підвищити точність розпізнавання порівняно із кадровим розпізнаванням виразів на статичних зображеннях. Використання комбінованого підходу із застосуванням Time Delay Neural Network та глибинної нерйронної мережі для покадрової обробки дозволяє ще більше підвищити точність розпізнавання порівняно із двома вищезазначеними методами. Однак в жодній із наведених робіт, особливо в останній, питання визначення ідентифікатора емоцій людини не було поставлене як завдання для дослідження, хоча остання робота досить наблизилася тематикою своїх дослідницьких завдань до теми нашого дослідження.

Аналіз найпопулярніших каркасів для побудови нейронних мереж.

TensorFlow – це бібліотека програмного забезпечення з відкритим кодом для чисельного обчислення з використанням графів потоку даних. Вузли на графі являють собою математичні операції, а грані графа представляють собою багатовимірні масиви даних (тензори), що передаються між ними. Гнучка архітектура дозволяє розгорнути обчислення для одного або декількох CPU або GPU на настільному комп'ютері, сервері або мобільному пристрої за допомогою одного API. TensorFlow був спочатку розроблений дослідниками та інженерами, що працюють у групі Google Brain Team в дослідницькій організації Google Machine Intelligence з метою проведення машинного навчання та дослідження глибинних нейронних мереж. Однак ця система є достатньо загальною, щоб бути застосованою у багатьох інших областях.

Keras – це високорівневий інтерфейс для розробки нейронних мереж, написаний на Python, і здатний працювати, використовуючи в якості бекенд – TensorFlow, CNTK або Theano. Він був розроблений з урахуванням можливості швидкого експерименту. Можливість перейти від ідеї до результату з мінімальною затримкою – це ключ до успішного дослідження.

Використовуйте Keras, якщо вам потрібна бібліотека глибокого навчання, яка:

- Дозволяє легко і швидко створювати прототипи (за рахунок зручності, модульності та розширюваності).
- Підтримує як згорткові мережі, так і періодичні мережі, а також комбінації обох.
- Працює без проблем на CPU та GPU.

Керівні принципи:

- Зручність у використанні. Keras – API, призначений для людей, а не машин, що дотримується найкращих практик зменшення когнітивного навантаження та пропонує послідовні та прості API,

мінімізує кількість дій користувача, необхідних для звичайних випадків використання, і забезпечує чіткий та дієвий відгук щодо помилок користувача.

- Модульність, що розуміється як послідовність автономних модулів, повністю готових до налаштування, які можна підключити разом із якомога меншими обмеженнями. Зокрема, нервові шари, функції витрат, оптимізатори, схеми ініціалізації, функції активації, схеми регуляції. Схеми регуляції – це автономні модулі, які можна поєднати для створення нових моделей.
- Легко розширювана. Нові модулі легко додавати як нові класи та функції, а існуючі модулі наводять достатньо прикладів. Можливість легко створювати нові модулі дозволяє тотальну виразність, що робить Keras придатним для передових досліджень.
- Робота з середовищем Python. Keras немає окремих файлів конфігурації моделей у декларативному форматі. Моделі, описані в коді Python – компактному, легко розширюваному, простому у налагодженні.

Іншим інструментом є *Caffe*. Основні його характеристики: виразна архітектура для заохочення інновації, моделі та оптимізація визначаються конфігурацією без жорсткого кодування, можливість перемикання між процесором і графічним процесором, встановивши єдиний прапорець для тренувань на машині графічного процесора, а потім розгортання на товарні кластери чи мобільні пристрої та розширюваний код для сприяння активного розвитку.

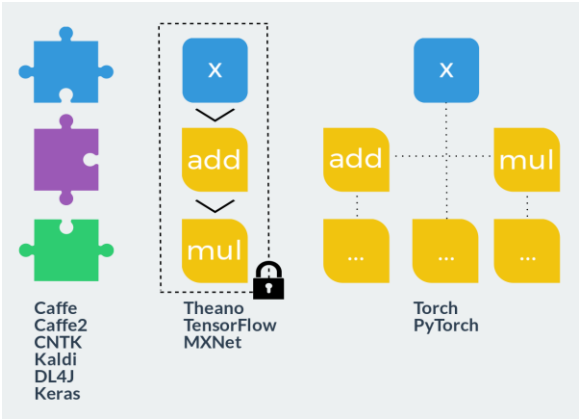
У перший рік Caffe був розширений більш ніж 1000 розробниками. Завдяки цим авторам система підтримує найсучасніші технології в коді та моделях.

Швидкість робить Caffe ідеальним інструментом для дослідницьких експериментів та розгортання різних напрямків промисловості. Caffe може обробляти понад 60M зображень на день за допомогою одного NVIDIA K40 GPU *. Це 1 мс/зображення для виводу та 4 мс/зображення для навчання, а останні версії бібліотеки та апаратні засоби ще швидшими.

Theano – це бібліотека Python, яка дозволяє ефективно визначати, оптимізувати та оцінювати математичні вирази з використанням багатомірних масивів. Особливості Theano:

- Тісна інтеграція з NumPy. Використовуйте numpy.ndarray у скомпільованих Theano-функціях.
- Прозоре використання GPU для виконання обчислення та великої кількості даних, набагато швидше, ніж на центральному процесорі.
- Ефективна символьна диференціація. Theano робить похідні для функцій з одним або багатьма входами.
- Оптимізація швидкості та стабільності. Отримайте правильну відповідь для журналу (1 + x), навіть коли x дійсно крихітний.
- Динамічне створення коду C – швидше для оцінювання виразів.
- Одиничне тестування та самопідтвердження для виявлення та діагностики багатьох видів помилок.

PyTorch являє собою бібліотеку для машинного навчання з відкритим кодом для Python, що використовується для створення таких додатків, як обробка природної мови, розпізнавання лиць, тощо. В основному розроблена вона дослідницькою групою із штучної інтелектуальної роботи Facebook, на основі якої побудовано програмне забезпечення Uber's "Pyro" для імовірнісного програмування.



Таблиця 1. Порівняння внутрішньої архітектури бібліотек.

Як уже було сказано вище, в ході розробки нейронних мереж за допомогою Python варто звернути увагу на вісім каркасів: TensorFlow, Caffe, The Microsoft Cognitive Toolkit/CNTK, PyTorch, MXNet, Chainer, Keras, Deeplearning4j та їхні характеристики. TensorFlow – найпопулярніший інструмент, на даний момент, який може застосовуватись будь-де як проект з відкрити кодом від Google. Caffe – це швидкий в роботі, але повільний в розробці. CNTK – це продукт від Microsoft. PyTorch – це другий за популярністю каркас з відкритим кодом від Facebook, за короткий період (1 рік) він набрав більшої популярності за всі інші каркаси в сумі, окрім TensorFlow та Keras. MXNet розроблявся як надефективний інструмент, щоправда таким він не став, можливо в майбутньому будуть здійснюватися над ним серйозні зміни. Chainer – це потужний динамічний інструмент, що дозволяє змінювати мережі під час виконання. Keras – мінімалістична бібліотека для побудови нейронних мереж, що найчастіше використовується в парі з TensorFlow або PyTorch в якості інструменту для навчання мережі. Deeplearning4j зазвичай використовується на великих підприємствах, для якого основною мовою є Java, але існують реалізації і на інших мовах високого рівня, в тому числі і на Python.

Отже серед даного набору, обрано найпопулярніший TensorFlow + Keras. Перевагою їх вибору є документованість і підтримка з боку інших розробників. Також можливий варіант PyTorch, замість TensorFlow, але популярність останнього більша. Інші каркаси не підходять у зв'язку зі складністю вихідного коду, а також недостатньо сформованою документацією, складністю використання або помилками при роботі з GPU. У 2013 році проходив конкурс з визначення емоцій людини за допомогою нейронних мереж. На сайті kaggle.com можна знайти більш детальну інформацію, а також скористатись архівом для тренування власної мережі.

Аналіз інструментів та підходів втілення.

TensorFlow – це одна з найкращих систем глибокого навчання, яку нині використовують такі гіганти як Airbus, Twitter, IBM, Google, головним чином завдяки високій гнучкості цієї системи [8].

Найбільш відомим випадком використання TensorFlow є Google Translate у поєднанні з такими можливостями, як обробка природної мови, класифікація тексту, узагальнення, розпізнавання мови, зображення, рукописного тексту, прогнозування та позначення [4, 8].

TensorFlow доступний як на настільних, так і на мобільних пристроях, а також він підтримує такі мови програмування, як Python, C++ і R для створення глибоких моделей навчання, бібліотек та обгорток. TensorFlow поставляється з двома інструментами, які широко використовуються на практиці застосування нейронних мереж.

TensorBoard, використовується для ефективної візуалізації даних мережевого моделювання та продуктивності TensorFlow-обслуговування для швидкого розгортання нових алгоритмів та експериментів при збереженні тієї ж архітектури сервера.

API [8] також забезпечує інтеграцію з іншими моделями TensorFlow, які відрізняються від традиційних і можуть бути розширені, щоб обслуговувати інші типи моделей і даних. Він ідеально підходить для новачків, зважаючи на можливість використання Python-підтримки від Google з документацію і покроковими поясненнями для розробників [4].

Caffe – це система глибинного навчання, яка підтримує мови програмування такі як C, C++, Python, MATLAB, а також інтерфейс командного рядка. Він добре відомий своєю швидкістю і придатністю для моделювання згорткових нейронних мереж (CNN) [10]. Найбільшою перевагою використання бібліотеки Caffe C++ (поставляється з інтерфейсом Python) є доступ до мереж з глибокого сховища «Caffe Model Zoo», які попередньо тренувалися і можуть бути використані повторно в будь-який момент часу. Незалежно від того, чи це моделювання CNN або вирішення проблем обробки зображень, ця бібліотека може стати остаточною для вирішення подібних задач [10].

Найбільша перевага Caffe – це швидкість. Він може обробляти більше шістдесяти мільйонів зображень на щоденній основі з однією відео-картою Nvidia K40. Це 1 мс/зображення для виведення і 4 мс/зображення для навчання, а новіші версії бібліотеки можуть працювати набагато швидше.

Caffe – це популярна мережа глибокого навчання для комп'ютерного бачення. Тим не менш, Caffe не підтримує точні шари мережевої деталізації, на відміну від TensorFlow або CNTK [10]. Враховуючи архітектуру, загальна підтримка повторюваних мереж та мовного моделювання є досить поганою, а створення складних типів шарів має здійснюватися на мові низького рівня.

Найвідоміший для легкої підготовки та поєднання популярних типів моделей на серверах Microsoft Cognitive Toolkit (раніше відомий як CNTK) – це фреймворк для глибокого навчання моделей з

відкритим вихідним кодом. Він виконує ефективні нейронні мережі Convolution і навчає даних на основі зображень, мови та тексту. Подібно до Caffe, він підтримується такими інтерфейсами, як Python, C++ та інтерфейс командного рядка.

Враховуючи її послідовне використання ресурсів, впровадження моделей зміцнення навчання або генеративних змагальних мереж (GAN) можна легко здійснити за допомогою інструментарію. Відомо, що вона забезпечує більш високу продуктивність і масштабованість у порівнянні з інструментами, такими як Theano або TensorFlow під час роботи на декількох машинах.

У порівнянні з Caffe, коли йдеться про винайдення нових складних типів шарів, користувачам чи розробникам не потрібно реалізовувати їх на мові низького рівня через тонку зернистість будівельних блоків. Microsoft Cognitive Toolkit підтримує типи нейронних моделей RNN та CNN і, таким чином, здатний керувати проблемами зображення, рукописного тексту та розпізнавання мови. В даний час, через відсутність підтримки архітектури ARM та можливості представлення його на мобільному телефоні досить обмежені [10].

Torch є науковою обчислювальною базою, яка пропонує широку підтримку алгоритмів машинного навчання. Це глибока основа для навчання Lua, що широко використовується серед таких гігантів, як Facebook, Twitter і Google. Він використовує CUDA разом з бібліотеками C/C++ для обробки даних. Основною метою створення Torch було масштабне виробництво будівельних моделей. Каркас PyTorch з радістю був прийнятий спільнотою глибинного навчання і вважається цілком конкурентом TensorFlow [11].

PyTorch – це версія каркаса, спрямована для використання в поєднанні з Python. Використовуються вона для побудови глибинних нейронних мереж і виконання тензорних обчислень, які є високими з точки зору складності [11].

На відміну від Torch, PyTorch працює на Python, що означає, що будь-хто з базовим розумінням Python може почати роботу над створенням власних моделей глибокого навчання. Враховуючи архітектурний стиль PyTorch, весь процес глибинного моделювання набагато простіший і прозоріший у порівнянні з Torch.

Розроблений спеціально для високої ефективності, продуктивності та гнучкості MXNet – це каркас глибинного навчання, який підтримує Python, R, C++ та Julia [9]. Суть краса MXNet полягає в тому, що він надає користувачеві можливість програмувати на різних мови програмування (Python, C++, R, Julia і Scala та ін.). Це означає, що в ньому можна тренувати свої моделі глибинного навчання, незалежно від того, якою мовою зручно програмувати без необхідності вивчати щось нове з нуля. З сервером, написаним на C++ і CUDA MXNet здатний масштабувати і працювати з безліччю GPU, що робить його незамінним для підприємств. В простому випадку Amazon використовує MXNet як свою довідкову бібліотеку для глибинного навчання [9, 11].

MXNet підтримує довгу короткочасну пам'ять (LTSM) [1] разом з RNN і CNN. Ця структура глибинного навчання відома своїми можливостями у візуалізації, розпізнаванні рукописного тексту чи мовлення, прогнозуванні, а також обробці людської мови (NLP).

Високо-потужний, динамічний і інтуїтивно зрозумілий Chainer – це глибинне навчання на основі Python. У порівнянні з іншими фреймворками, які використовують одну і ту ж стратегію, Chainer дозволяє змінювати мережі під час виконання, що дозволяє виконувати довільні операції керування потоком.

Chainer підтримує обчислення як CUDA, так і мульти-GPU. Ця система глибинного навчання використовується в основному для аналізу настроїв, машинного перекладу, розпізнавання мови і т.д., також з використанням RNN і CNN.

Keras добре відома як мінімалістична бібліотека для побудови нейронних мереж з підтримкою інтерфейсу в Python. Keras підтримує як згорткові, так і повторювані мережі, які здатні працювати на TensorFlow або Theano. Бібліотека, написана на Python, була розроблена, зберігаючи швидке експериментування як USP [2, 3].

Через те, що інтерфейс TensorFlow є трохи складним, а також тим, що це бібліотека низького рівня, яка може бути складною для нових користувачів. Keras був побудований, щоб забезпечити спрощений інтерфейс для швидкого прототипізування шляхом побудови ефективних нейронних мереж, які можуть працювати з TensorFlow [5]. Він є легкий, простий у використанні і дуже доступний, коли справа

доходить до побудови моделі глибинного навчання шляхом укладання декількох шарів – це Keras в двох словах. Саме з цих причин Keras є частиною основного API TensorFlow [2, 3].

Первинне використання Keras полягає в класифікації, генерації та узагальненні тексту, маркування, перекладі разом з розпізнаванням мови та ін. Якщо ви виявилися розробником з певним досвідом роботи на Python і бажаєте заглибитися в глибоке навчання, то Keras – це те, що ви обов'язково повинні перевірити.

Паралельне навчання через ітеративне зменшення, адаптація архітектури мікро-сервісу в поєднанні з розподіленими процесорами та графічними процесорами – це лише деякі з найбільш явних особливостей, коли мова йде про структуру глибинного навчання *Deeplearning4j*. Він розроблений на Java, а також Scala і підтримує теж інші мови JVM [14]. Це широко прийнята як комерційна, орієнтована на промисловість платформа для глибинного навчання, найбільшою перевагою цього каркасу є те, що ви можете об'єднати всю екосистему Java, щоб виконати навчання моделі і щоб її можна було керувати поверх Hadoop і Spark для оркестрування кількох потоків вузла. *DL4J* використовує карту зменшення, щоб навчити мережу в залежності від інших бібліотек для виконання великих матричних операцій.

Deeplearning4j поставляється з мережевою підтримкою через RBM, DBN, згорткові нейронні мережі (CNN), повторювані нейронні мережі (RNN), рекурсивну мережу нейронних тензорів (RNTN) і довгу короткочасну пам'ять[1] (LTSM). Оскільки ця система глибокого навчання реалізована на Java, вона набагато ефективніша в порівнянні з Python. Коли йдеться про завдання розпізнавання зображень за допомогою декількох графічних процесорів, вона працює так само швидко, як і Caffe. Ця структура показує незрівнянний потенціал для розпізнавання зображень, виявлення шахрайства, видобування тексту, частин тегування мови та обробки людської мови. Завдяки Java як основній мові програмування, слід обрати даний фреймворк обов'язково, якщо ви шукаєте надійний і ефективний метод впровадження ваших моделей у виробництво [14].

Досить очевидно, що поява глибокого навчання ініціювала багато випадків практичного використання машинного навчання та штучного інтелекту в цілому. Розділення завдань найпростішими способами для того, щоб допомогти машинам найефективнішим способом, було зроблено, ймовірно, за допомогою Deep Learning. Це, як кажуть, є ті глибокі рамки навчання з вищенаведеного списку, які найкраще підходять вашим вимогам? Відповідь на це питання лежить на ряді факторів, однак, потрібно просто почати роботу, то глибокі основи навчання на основі Python, такі як TensorFlow або Chainer, повинні бути вашим вибором. Якщо вам доведеться впоратися, потрібно враховувати швидкість, вимоги до ресурсів і використання разом з узгодженістю підготовленої моделі, перш ніж вибрати найкращу глибоку структуру навчання [14].

Оцінка актуальності використання. Порівняльна характеристика враховує документацію, вихідний код, архітектуру та тяжкість встановлення. Також в дослідженні не брав участь Deeplearning4j, але зважаючи на повільнішу розробку на Java у порівнянні з Python, можна зробити висновок, що уповільнення розробки перевищить в 10 разів

Для швидкого тренування моделі варто обрати PyTorch та Keras. При даній комбінації відбувається пришвидшення розробки на 41% порівняно з Caffe, і на 18% швидше у порівнянні з TensorFlow.

Розробка та тренування моделі для аналізу зображень за допомогою Python

В дослідженні застосовувався *стек технологій CPython 3.6.6, TensorFlow, Keras.* В якості апаратного забезпечення для тренування нейронної мережі використано сервіс Google Colaboratory, який надає відеоприскорювач Tesla K80 для навчальних цілей. Для тренування мережі також використано архів від kaggle.com – fer2013 [6].

Створення збірної нейронної мережі має змогу визначати сім різних емоцій: злість, огиду, страх, радість, сум, здивування і нейтральний стан. Тренування відбувалось для 60 епох.

Реалізація коду:

```
# -*- coding: utf-8 -*-
import sys
import os

import pandas as pd
import numpy as np
```

```
from google.colab import drive
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D, BatchNormalization
from keras.losses import categorical_crossentropy
from keras.optimizers import Adam
from keras.regularizers import l2
from keras.callbacks import ReduceLROnPlateau, TensorBoard, EarlyStopping, ModelCheckpoint

# Монтуємо GDrive
drive.mount("/content/gdrive")

WORKDIR = "/content/gdrive/My Drive/Colab Notebooks"
MODELPATH = './model.h5'    # Шлях збереження моделі

NUM_FEATURES = 64
NUM_LABELS = 7
BATCH_SIZE = 64
EPOCHS = 60
WIDTH, HEIGHT = 48, 48

sys.path.insert(0, WORKDIR)
os.chdir(WORKDIR)

data = pd.read_csv('./fer2013.csv')

pixels = data['pixels'].tolist()

faces = []
for pixel_sequence in pixels:
    face = [int(pixel) for pixel in pixel_sequence.split(' ')]
    face = np.asarray(face).reshape(WIDTH, HEIGHT)
    faces.append(face.astype('float32'))

faces = np.asarray(faces)
faces = np.expand_dims(faces, -1)

emotions = pd.get_dummies(data['emotion']).as_matrix()

X_train, X_test, y_train, y_test = train_test_split(faces, emotions, test_size=0.1, random_state=42)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.1, random_state=41)

model = Sequential()

model.add(Conv2D(NUM_FEATURES, kernel_size=(3, 3), activation='relu', input_shape=(WIDTH, HEIGHT,
        1), data_format='channels_last', kernel_regularizer=l2(0.01)))
model.add(Conv2D(NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Conv2D(2 * NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
```

```
model.add(BatchNormalization())
model.add(Conv2D(2 * NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Conv2D(2 * 2 * NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(2 * 2 * NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Conv2D(2 * 2 * 2 * NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(2 * 2 * 2 * NUM_FEATURES, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Flatten())

model.add(Dense(2 * 2 * 2 * NUM_FEATURES, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(2 * 2 * NUM_FEATURES, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(2 * NUM_FEATURES, activation='relu'))
model.add(Dropout(0.5))

model.add(Dense(NUM_LABELS, activation='softmax'))

model.summary()

model.compile(loss=categorical_crossentropy,
        optimizer=Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-7),
        metrics=['accuracy'])

lr_reducer = ReduceLROnPlateau(monitor='val_loss', factor=0.9, patience=3, verbose=1)

tensorboard = TensorBoard(log_dir='./fer.log')

early_stopper = EarlyStopping(monitor='val_loss', min_delta=0, patience=8, verbose=1, mode='auto')

checkpointer = ModelCheckpoint(MODELPATH, monitor='val_loss', verbose=1, save_best_only=True)
```

Створення додутку на основі тренованої моделі

В якості алгоритму класифікації використано каскад Хаара, а для аналізу зображень – OpenCV [7, 13].

```
import cv2
import numpy as np
from keras.models import load_model
```

```
emotion_dict = {0: "Angry", 1: "Disgust", 2: "Fear", 3: "Happy", 4: "Sad", 5: "Surprise", 6: "Neutral"}
# Завантаження моделі
model = load_model("./model.h5")

cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)

    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 1)
        roi_gray = gray[y:y + h, x:x + w]
        cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray, (48, 48)), -1), 0)
        cv2.normalize(cropped_img, cropped_img, alpha=0, beta=1, norm_type=cv2.NORM_L2,
            dtype=cv2.CV_32F)
        prediction = model.predict(cropped_img)
        cv2.putText(
            frame,
            emotion_dict[int(np.argmax(prediction))],
            (x, y),
            cv2.FONT_HERSHEY_SIMPLEX,
            0.8,
            (0, 0, 255),
            1,
            cv2.LINE_AA
        )

    cv2.imshow('frame', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Додаток стартує за допомогою інтерпретатора Python і використовує WEB-камеру в якості пристрою зчитування інформації.

Постановка наукової проблеми. Розроблений продукт повинен визначати емоції людини за фото-чи відео-матеріалами у вигляді портретних чи художніх фотографій. Для проведення дослідження подібна система повинна мати можливість використовувати камеру, підключену до пристрою на якому запущений додаток з підключеною нейронною мережею та можливістю обробляти отримані дані, а також видавати результат-висновок.

Ставилося завдання підтримки та розповсюдження результатів в трьох форматах: відкритого вихідного коду, тренованої моделі та отримання готового додатка з можливістю використання його на системах Linux, Mac OS, Windows чи в Web. Для останнього варіанту можливість документування та оновлення документації повинна бути відкрита для розробників програмного продукту та інтегрована з системами контролю версій для забезпечення підтримки і аналізу помилок розробки.

Результати моделі, їх реалізація та обговорення

Для аналізу зображень нами було обрано згорткову нейронну мережу, яка мала реревагу в ефективності для задач класифікації. В якості алгоритму для градієнтного спуску було обрано спеціальний алгоритм Adam, опублікований в 2015 для застосування його в подібних задачах. Згідно з нашими дослідженнями алгоритм показав досить невисоку зайнятість пам'яті та відзначився в швидкодії виконання задач.

Для проведення експерименту також обрано fer2013 – спеціально сформовану базу даних обличь як еталонних для порівняльних цілей. Всі наведені дані в цій базі представлені таблицею чорно-білих двовимірних масивів. Зазвичай, в ході створення нейронної мережі необхідно власноруч підготувати тренувальний сет, процес підготовки якого займає біля 70% часу дослідження.

Обирати нейронну мережу прямого поширення в нашому випадку не має сенсу, тому що не зважаючи на простоту, аналіз зображень обійдеться надто дорогим. Якщо ми вважатимемо кожен канал кожного пікселя незалежним вхідним параметром для пересептрона, тоді кожен нейрон буде додавати HxWxD нових параметрів до моделі, де: W – ширина зображення, H –висота і D – глибина. Одне з найпоширеніших для нашого дослідження рішень – це знизити роздільну здатність зображень, поки не з'явиться можливість використати персептрон. Відповідно, в ході обробки даних, що потребують високої точності, дане рішення все ж не підходить.

Функції згортки.

Припустимо, що пікселі, які розташовані близько один до одного, тісніше взаємодіють під час формування необхідної нам ознаки. В даному випадку також буде відсутній вплив розміщення при умові, якщо взяти до уваги невелику характеристику, яка є важливою в процесі класифікації.

Оператор згортки.

У нас є двовимірне зображення I і не велика матриця K, з розмірами HxW, яка являє собою ядро згортки, і побудована таким чином, щоб графічно здійснювати кодування якої небудь ознаки (рис. 1). В такому разі ми можемо обчислити згорнуте зображення I*K, наклавши ядро на нього всіма можливими способами і записати суму добутків елементів вихідного зображення і ядра.

$$(I * K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} \cdot I_{x+i-1,y+j-1};$$

Для більш точних розрахунків можна транспонувати матрицю, але в машинному навчальному варіанті це не має значення.

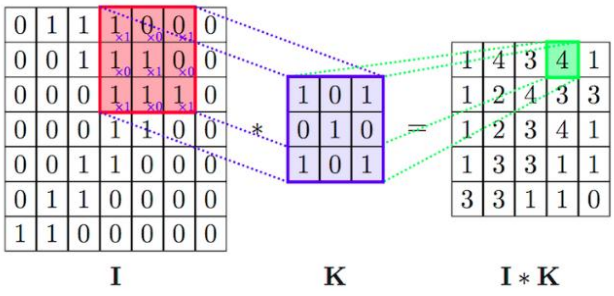


Рис. 1. – Вигляд представлення згорткових шарів.

Оператор згортки являє собою основу згорткового шару в згорткових нейронних мережах. Шар складається з певної кількості ядер K з адаптивними складовими зміщення для кожного ядра і визначає згортку вихідного зображення попереднього шару, кожен раз додаючи складову зміщення. В результаті, до всього вихідного зображення може бути застосована функція активації σ. Зазвичай вхідний потік для згорткового шару складається з каналів (d), наприклад, RGB для вхідного шару. В такому випадку ядра також розширюють таким чином, щоб вони складались з каналів. Тому для одного каналу вихідного зображення застосовується наступна формула:

$$conv(I,K)_{x,y} = \sigma\left(b + \sum_{i=1}^{h}\sum_{j=1}^{w}\sum_{k=1}^{d} K_{ijk} \cdot I_{x+i-1,y+j-1,k}\right)$$

де *K* – ядро, *a* і *b* – складові зміщення.

Ядра також можна отримати методом градієнтного спуску, подібно до визначення ваг в багатошаровому персептроні.

Операції згортки не єдині в мережах даного типу. Ми використали агрегування, а саме шар субвибірки (pooling layer), що отримує на вхід окремі частинки зображення і об'єднує кожен фрагмент в єдине значення.

В ході дослідження також застосований шаблон побудови CNN для класифікації зображень (рис. 2). Даний шаблон можна розділити на дві частини: ланцюг послідовних згорткових шарів: *Conv -> Poo*l, для якого можливі варіації з декількома шарами згортки підряд, і декілька повнозв'язних шарів, що приймають кожен піксель як незалежне значення з шаром *softmax* в якості канцевого. Також після кожного згорткового шару застосовується функція активації, а в нашому випадку ReLU.

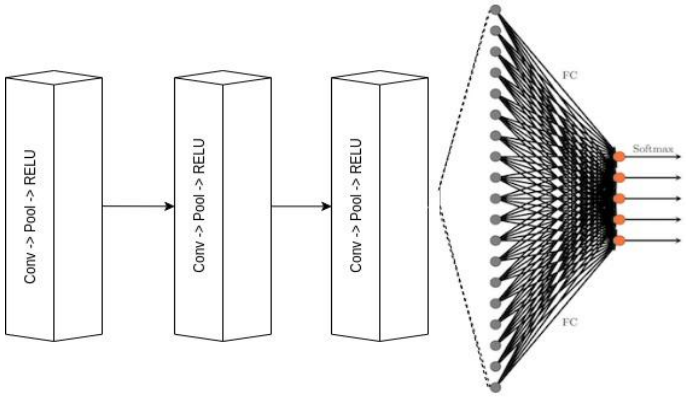


Рис. 2 – Схема шаблону побудови CNN для класифікації зображень

Один прохід Conv -> Pool впливає на зображення таким чином, що скорочує довжину і ширину певного каналу, однак при цьому збільшує його значення глибини.

Застосована функція Softmax перетворює вектор дійсних чисел у вектор імовірностей (представлення дійсними числами в інтервалі [0,1]). В нашому випадку в якості вихідних значень застосовувались імовірності потрапляння зображення в певний клас, який представляв відповідну емоцію.

Перенавчання

Перенавчання представляє собою відповідність нейронної мережі конкретному набору навчальних прикладів, для яких мережа втрачає здібність до узагальнення. В шаблоні CNN наявні багато різноманітних параметрів. Перенавчання може проявитись у наступному. Припустимо що у нас мало навчальних прикладів. В цьому випадку відповідно невелика група нейронів бере на себе більшу кількість обчислень, а інші нейрони стануть надлишковими. Або, навпаки, певні нейрони зашкодять продуктивності, проте інші нейрони з їхнього шару будуть тільки виправляти їх помилки.

Для того щоб не дозволити нейронам вивчати шуми навчальних даних, нами введено методи регулювання: замість зменшення кількості параметрів ми накладаємо умову на параметри моделі під час навчання. Одним з методів зниження є метод виключення (dropout).

В цілому виключення з параметром *p* за одну ітерацію навчання проходить до всіх нейронів певного шару I з імовірністю *p* і повністю виключає їх з мережі на час ітерації. Це змушує мережу опрацьовувати помилки і не покладатись на існування певного нейрона, або групи нейронів, а покладатись на "спільну думку" нейронів всередині одного і того шару.

Нами отримано наступні ID-емоцій, які представлені нижче відповідними рисунками. Першою отримано емоцію «Щастя» (рис. 3), представлену в робочому фреймі. На рисунку 4 визначено емоцію «Здивування», а на рисунку 5 – емоцію «Сум».

Висновки і перспективи. В даному параграфі наведено результати досліджень методів та особливостей для визначення емоцій людини за допомогою нейронних мереж. Розробка моделі для

аналізу зображень проводилась за допомогою TensorFlow, а тренування реалізовувалось з використанням Keras. Вхідні дані використано з архіву kaggle.com – FER2013. Для аналізу зображень використано бібліотеку OpenCV та мову програмування Python 3.

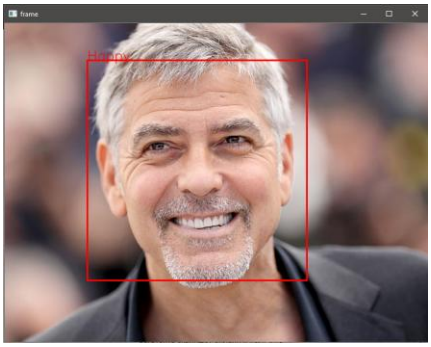


Рис. 3 – Визначення емоції щастя [21]

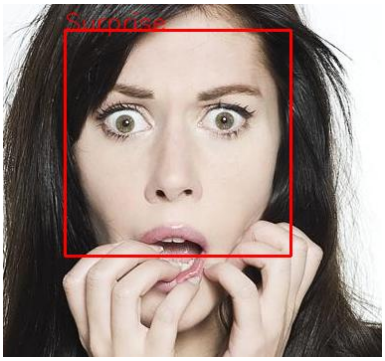


Рис. 4 – Визначення емоції здивування

Рис. 5 – Визначення емоції сум

Проведено аналіз інструментів для розробки нейронної мережі з метою розпізнавання емоцій людини з фото, відео, або зображення веб-камери. З'ясовано основні підходи та особливості дослідження алгоритмів обробки та класифікаторів зображень, що дозволило створити конфігурацію для навчання нейронної мережі і демонстрації програмного продукту. На основі проробленого дослідження можна зробити висновок, що додаток має як переваги так і недоліки. До переваг можна віднести: зрозумілість коду, можливість його переносу на іншу машину для подальших обчислень, розширюваність для використання сумісно з іншими інструментами. До недоліків можна відмітити те, що сама по собі нейронна мережа не несе високої цінності науковцям-дослідникам, яким цікаво просто розпізнати емоції людини за фото або відео-ілюстраціями. Однак, вона націлена на розробників і науковців. Для більш практичного застосування її потрібно використовувати сумісно з іншими модулями. Наприклад, у поєднанні з OpenCV [21, 27] дана мережа може визначати емоції не однієї людини, а кількох одночасно з одного фото або відео.

Слід відмітити, що особливо великих результатів можна досягнути у випадку, поєднуючи цю технологію з іншими відомими технологіями, наприклад, такими як технологія об'єктно-орієнтованого програмування. Таке поєднання технологій мало б більш широкий спектр застосування в різних областях, починаючи від звичайних фото-світлин, викладених в соціальних мережах і закінчуючи контролем поведінки громадян держави або навіть планети. Аналіз емоцій дає можливість продуктовим і рекламним компаніям значно збільшити об'єм продаж, що в свою чергу збільшить прибутки [26]. Бути геніальним співбесідником, маючи можливість маніпулювати людьми знаючи що вони думають, проводити стресс тести співробітників та оцінювати їхню реакцію, визначати реакцію людини на рекламу, оголошення, промову збирати обробляти і робити висновки. Список сфер використання обмежений лише фантазією і очікуючим результатом.

Список бібліографічних посилань

1. Вікіпедія. Довга короткочасна пам'ять // Електронний ресурс. URL: https://uk.wikipedia.org/wiki/Довга короткочасна пам'ять (дата звернення: 18.03.2018).
2. Вікіпедія. Keras – відкрита нейромережева бібліотека. URL: https://uk.wikipedia.org/wiki/Keras (дата звернення: 18.03.2018).
3. Quora Contributor. This Is What Makes Keras Different, According To Its Author/ Electron resource. URL: https://www.forbes.com/sites/quora/2016/08/25/this-is-what-makes-keras-different-according-to-its-author/ #556b58ba66cf. (дата звернення: 18.03.2018).
4. An end-to-end open source machine learning platform. Elecrton resource. URL: https://www.tensorflow.org/ (дата звернення: 18.03.2018).
5. Keras: The Python Deep Learning library. Electron resource. URL: https://keras.io/. (date of issue: 18.03.2018).
6. We use cookies on kaggle to deliver our services, analyze web traffic, and improve your experience on the site. Start with more than a blinking cursor. Electron resource. URL: https://www.kaggle.com/ (date of issue: 18.03.2018).
7. AL Cources by OpenCV. Electron resource. URL: https://opencv.org/courses/ (date of issue: 18.03.2018).
8. Вікіпедіа. TensorFlow. URL: https://ru.wikipedia.org/wiki/TensorFlow. (date of issue: 18.03.2018).
9. Вікіпедіа. Apache_MXNet. URL: https://en.wikipedia.org/wiki/Apache_MXNet. (date of issue: 18.03.2018).
10. Вікіпедіа. Згорткова нейронна мережа. Електронний ресурс. URL: https://uk.wikipedia.org/wiki/ Згорткова_нейронна_мережа. (дата звернення: 18.03.2018).
11. Вікіпедіа. Глибинне навчання. Електронний ресурс. URL: https://uk.wikipedia.org/wiki/Глибинне_ навчання. (дата звернення: 18.03.2018).
12. Вікіпедіа. Застосування штучного інтелекту. Електронний ресурс. URL: https://uk.wikipedia.org/wiki/ Застосування_штучного_інтелекту. (дата звернення: 18.03.2018).
13. Вікіпедіа. Розпізнавання рукописного введення. Електронний ресурс. URL: https://uk.wikipedia.org/ wiki/Розпізнавання_рукописного_введення. (дата звернення: 18.03.2018).
14. Вікіпедіа. Комп'ютерний зір. Електронний ресурс. URL: https://uk.wikipedia.org/wiki/Комп%27 ютерний_зір. (дата звернення: 18.03.2018).
15. Нейроні мережі і генетичні алгоритми – К.: «Корнійчук», . 2008. – 446 с.
16. Глубокое обучение на Python – СП.:«Питер», . 2018. – 400 с.
17. Нейронные сети: полный курс – М:«Диалектика-Вильямс», . 2017. – 1104 с.
18. Офіційна документація TensorFlow. URL: https://www.tensorflow.org/ (дата звернення: 18.03.2018)
19. Офіційна документація Keras. URL: https://keras.io/ (дата звернення: 18.03.2018)
20. Платформа для змагань з машинного навчання і передбачуваного моделювання. URL: https://www.kaggle.com/ (дата звернення: 18.03.2018)
21. Офіційна документація OpenCV. URL: https://opencv.org/ (дата звернення: 18.03.2018)
22. Программирование искусственного интеллекта в приложениях – ДМК пресс, «М.: Тим Джонс», . 2006. – 312 с.
23. Нейронные сети: распознавание, управление, принятие решений – М:«Финансы и статистика», . 2004. – 176 с.
24. Основные концепции нейронных сетей – М.:«Вильямс», . 2003. – 288 с.
25. Основы Data Science и Big Data. Python и наука о данных – СП : «Питер», 2016. – 336 с.
26. Python для сложных задач: наука о данных и машинное обучение – СП. : «Питер», . 2016. – 576 с.
27. Винер Н. Кибернетика, или Управление и связь в животном и машине. / Пер. с англ. И.В. Соловьева и Г.Н. Поварова; Под ред. Г.Н. Поварова. – 2-е издание. – М.: Наука; Главная редакция изданий для зарубежных стран, 1983. – 344 с.
28. Галещук С. Штучні нейронні мережі у прогнозуванні валютного ринку. URL: http://visnik.knteu.kiev.ua/files/2016/03/9.pdf (дата звернення: 18.03.2018)
29. Сєдай А.В. Використання нейронних мереж для моделювання та прогнозування фінансової діяльності транспортного підприємства. URL: http://publications.ntu.edu.ua/eut/2015-02/115-120.pdf (дата звернення: 18.03.2018).
30. Traffic from January 2014 to September 2017, by month [Електронний ресурс] — Режим доступу: https://www.statista.com/statistics/420391/spam-email-traffic-share/ (дата звернення: 10.06.2019).
31. Vikas P. Deshpand. An Evaluation of Naive Bayesian Anti-Spam Filtering Techniques / Vikas P. Deshpande, Robert F. Erbacher, Chris Harris // Proceedings of the 2007 IEEE Workshop o n Information Assurance United States Military Academy, West Point, 2007. – NY 20-22 June. – Режим доступу: http://digital.cs.usu.edu/~erbacher/publications/Bayes-Vikas2.pdf. (дата звернення: 10.06.2019).
32. Graham P. A Plan for Spam / P. Graham, 2002. [Електронний ресурс] — Режим доступу: http://www.paulgraham.com/spam.html. (дата звернення: 10.06.2019).
33. Мельник К.В., Мельник В.М., Коптюк Ю.Ю. Дослідження методів розпізнавання зображень на основі нейронних мереж. Науковий журнал "Комп'ютерно-інтегровані технології: освіта, наука, виробництво", Луцьк, 2019. Вип. № 35. – с. 161-165.
34. Гончаров Д.А. Програмні засоби розпізнавання акустичної інформації в сенсорних мережах / Магістерська дисертація на здобуття ступеня магістра зі спеціальності 123 "Комп'ютерна інженерія" спеціалізації "Спеціалізовані комп'ютерні системи". Київ, КПІ. 2018. – 122 с. URL: http://ela.kpi.ua/ bitstream/123456789/25805/1/Matviiv_magistr.pdf.
35. Саган В.Ю. Система аналізу біомедичних сігналів для контролю стану водіїв. / Магістерська дисертація на здобуття ступеня магістра зі спеціальності 122 Компьютерні науки та інформаційні технології. // Київ, КПІ. 2018. – 122 с. URL: http://ela.kpi.ua/bitstream/123456789/24036/1/ Sagan_magistr.pdf
36. А.А. Яровий, С.Г. Кашубін, О.О. Кулик. Розпізнавання мімічних мікровиразів обличчя людини. / Системи технічного зору та штучного інтелекту з обробкою та розшаруванням зображень. Вінниця. – ВНТУ, 2015. – с.76-83.