МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Фесенко М.А., Кисіль Т.М., Чичкарьов Є.А.,
Звенігородський О.С.

МЕТОДИЧНІ ВКАЗІВКИ

для виконання практичних робіт з дисципліни

«Штучні нейронні мережі»

для студентів спеціальностей:
121 «Інженерія програмного забезпечення»,
122 «Комп'ютерні науки», 123 «Комп'ютерна інженерія»,
124 «Системний аналіз», 125 «Кібербезпека та захист інформації»,
126 «Інформаційні системи та технології»
денної, заочної і дистанційної форми навчання

КИЇВ 2023

УДК 681.5 (075.8)

Рекомендовано на засіданні вченої ради Навчально-наукового інституту інформаційних технологій (Протокол № 2 від 13.09.2023 року).

Фесенко М.А., Кисіль Т.М., Чичкарьов Є.А., Звенігородський О.С. Методичні вказівки для виконання практичних робіт з дисципліни «Штучні нейронні мережі» для студентів спеціальностей: 121 «Інженерія програмного забезпечення», 122 «Комп'ютерні науки», 123 «Комп'ютерна інженерія», 124 «Системний аналіз», 125 «Кібербезпека та захист інформації», 126 «Інформаційні системи та технології» денної, заочної і дистанційної форми навчання. – К.: ДУІКТ, 2023. - 48 с.

Рецензенти: **Гайдур Г.І.**, доктор технічних наук, професор, завідувач кафедри Інформаційної та кібернетичної безпеки Державного університету інформаційно-комунікаційних технологій.

Іларіонов О.Є, кандидат технічних наук, доцент, завідувач кафедри інтелектуальних технологій Київського національного університету імені Тараса Шевченка.

Методичні вказівки для виконання практичних робіт з дисципліни «Штучні нейронні мережі» розроблено відповідно до освітніх програм спеціальностей – 121, 122, 123, 124, 125, 126. Предметом навчальної дисципліни є вивчення студентами основних принципів функціонування штучних нейронних мереж, засвоєння основних концепцій, типів архітектури та методів навчання нейронних мереж, оволодіння навичками моделювання, а також програмування нейронних мереж на персональному комп'ютері, а також вирішення інтелектуальних задач за допомогою штучних нейронних мереж. У методичних вказівках для виконання практичних робіт достатньо прикладів для легкого вивчення навчального матеріалу.

Зміст

Практична робота №1

Тема. Інтелектуальні Веб-сервіси

Мета роботи. Ознайомитися з популярними інтелектуальними веб-сервісами, дослідити їх функції та можливості. Ознайомитися з бібліотеками та продуктами на новітній платформі штучного інтелекту Google AI. Провести низку експериментів на доступних інтелектуальних сервісах і проаналізувати отримані результати.

Теоретичні відомості

Інтернет з кожним днем все більше нагадує самоорганізоване середовище, що еволюціонує з шаленою швидкістю. І хоча ця система ще не має повноцінного штучного інтелекту, цікаві застосунки вже починають користуватися популярністю (наприклад, віртуальні співрозмовники, машинний зір та аудіо інтерфейс).

Розробкою інтелектуальних застосунків займаються спільноти, що об'єднані ідеями, цілями та інтересами, які готові витрачати свій час і ресурси на втілення цих ідей. Тому, з кожним днем в Інтернеті з'являється все більше розумних програм, їх функціонал стає більш широким, а відвідувачі перетворюються зі споживачів в активних творців контенту.

Технології штучного інтелекту в IT-індустрії задіяні для чисельних проєктів в цій сфері, вражаюче за масштабами проникнення інтелекту практично у всі області сучасного життя – від медицини, експертних систем і наукових досліджень до промислової робототехніки і безпілотного транспорту. Напрямок машинного навчання і нейронних мереж активно розвивається і вдосконалюється, в ньому задіяні Intel, AMD, NVIDIA, IBM, Google, Facebook, ABBYY, а також тисячі інших компаній-розробників по всьому світу.

Ще 20 років тому штучний інтелект можна було протестувати тільки в крутих лабораторіях або великих компаніях, а сьогодні це доступно для більшості зацікавлених людей і його значне зростання передбачає багато аналітиків. Розглянемо деякі інтелектуальні веб-сервіси.

1. Інтелектуальна гра Акінатор

«Акінатор» це Інтернет-застосунок, який розроблений двома французькими програмістами в 2007 році. Користувач повинен загадати будь-якого персонажа, а Акінатор його відгадує. Такими персонажами можуть бути як реальні особи, так і вигадані персонажі з фільмів, казок, комп'ютерних ігор тощо.

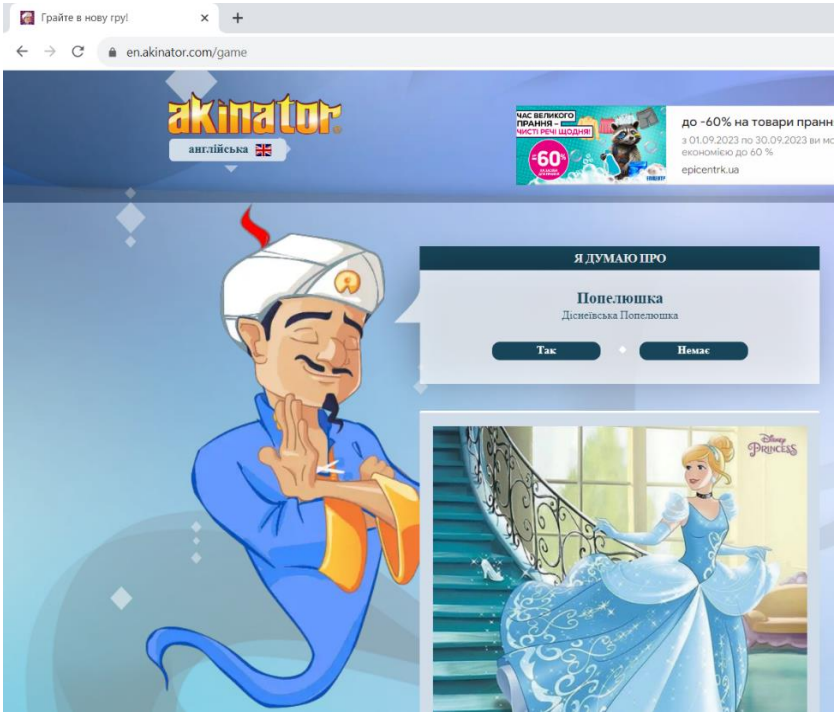


Рис. 1. Інтерфейс інтелектуальної гри Акінатор

Програма виконує наступні вимоги:

1. **Здатність до навчання.** Очевидно, що не можна навчити програму розпізнавати кілька сотень тисяч персонажів шляхом ручного введення відповідей на питання для кожного з них. Альтернатива до цього підходу – вчитися на ходу, користуючись відповідями користувачів.

2. **Програма толерантна до помилок.** Думка користувачів з приводу відповідей на деякі питання може значно різнитися. Простий приклад: палка шанувальниця і звичайний чоловік загадують Олега Винника. На питання «Ваш персонаж сексуальний?» перша, ймовірно, відповість так, в той час як більшість людей буде мати інші відповіді. Однак подібне розходження не повинно заважати.

3. **Розумний вибір питання.** Є багато стратегій, що визначають питання, які потрібно задавати. Наприклад, можна задати взагалі всі або випадкові питання, але до відповіді доведеться йти дуже довго. А можна намагатися вибирати чергове запитання так, щоб дізнатися при відповіді на нього якомога більше інформації. Саме це ми і будемо намагатися робити.

Акінатор задає до 40 запитань. На випадок, якщо він не зміг відгадати загаданого гравцем персонажа він має дві додаткові спроби (в кожній кілька додаткових питань). На кожне питання пропонується вибрати один з п'яти варіантів відповіді: «Так», «Можливо, частково», «Я не знаю», «Скоріше ні, не зовсім», «Ні». Акінатор починає з більш загальних питань, і кожне наступне питання має уточнюючий характер. Таким чином Акінатор фільтрує вибір ймовірних чи невідповідних персонажів.

Один персонаж характеризує множина параметрів, кожен з яких представляє собою вісь (вимір) в багатовимірної системі координат. Поточне задане питання прояснює значення відповідного параметра. Коли значна кількість людей загадує певного персонажа, сукупність їх відповідей (параметрів) для цього персонажа утворює множину точок, локалізованих в певній області щодо даної системи координат. У міру набору статистики, для кожного персонажа утворюється своє скупчення точок (кластер). Персонажні кластери знаходяться на різній відстані один від одного в залежності від того, чим дані персонажі відрізняються.

Наприклад, якщо обидва персонажі - актори, живуть в Америці, старше 50 років, то по цих осях кластери даних персонажів будуть збігатися. Але, якщо один з них лисий, а інший – з шевелюрою, то дані кластери можуть бути легко розділені по осі лисий-нелисий за допомогою питання: "чи задуманий персонаж є лисий?". Після кожного такого питання, Акінатор перевіряє стан точки, утвореної відповідями загадувати, в системі координат питань щодо персонажному кластеру. Якщо за десятком вимірювань видно, що ця точка належить до певного кластеру, а решта кластери досить далеко, Акінатор легко справляється з загадкою.

Однак, часто буває так, що кластери розташовані дуже близько і зливаються. Це може бути, наприклад, якщо обидва вищезазначених актора є лисими і при цьому носять вуса. В цьому випадку Акінатору потрібно шукати додаткові виміри (питання), щоб їх розрізнити. Оскільки люди часто помиляються у відповідях, то кластери, що відрізняються по одній координаті можна легко переплутати, особливо якщо набрана з даного питання статистика недостатня, а питання нерелевантні.

Вдалося Акінатору знайти релевантне питання-вимір, за яким кластери різняться надійно – він вгадав. Не вдалося – з великою ймовірністю він видає помилковий варіант. Оскільки, в процесі навчання програми винаходяться все нові питання (з'являються нові виміри), і для кожного питання набирається статистика (поліпшується точність розрізнення по осях), ефективність вгадування персонажів Акінатор буде постійно зростати.

У разі, якщо Акінатор не відгадав персонажа, то він представляє можливих персонажів, яких він припускав і пропонує ввести назву загаданого персонажа, після чого запам'ятовує всі відповіді, які були задані. Таким чином, кількість персонажів, відомих Акінатор, постійно збільшується. Для більшості користувачів кількість персонажів зазвичай є обмеженою (до 100 персонажів), тому навіть стартової бази цілком вистачає для того, щоб відгадувати більшу частину запитів.

2. Poem Portrets

Сервіс від Google Labs Arts & Culture Lab експериментує на перетині штучного та людського інтелекту – поєднання поезії, дизайну та машинного навчання. Poem Portrets – це автопортрет користувача, на який накладено унікальну поему, що створена штучним інтелектом.

Для роботи сервісу потрібно вказати ключове слово для вірша і зробити селфі. Зазначене слово буде розширено на оригінальні поетичні рядки за алгоритмом, який базується на вивчений мільйонах слів поезії XIX століття. Після виконання обчислень користувач отримує унікальний Poem Portrets свого обличчя, на який накладено оригінальні поетичні рядки. Всі створені поетичні рядки в подальшому поєднуються, щоб поповнювати колективну поему.

Сервіс не копіює і не переробляє існуючі фрази, а використовує навчальний матеріал для побудови складної статистичної моделі. В результаті алгоритм генерує оригінальні фрази, що імітують стиль того, на чому він навчався. Отримані вірші можуть бути як змістовними так й безглуздими.

Рис. 2. Результат виконання експерименту
з застосуванням сервісу Poem Portrets

3. Розпізнавання об’єктів Giorgio Cam

Сервіс призначений для розпізнавання об'єкту, що фіксується камерою. Нейромережа розпізнає об'єкт, складає вірш за підсумком результату розпізнавання і накладає відповідний музичний супровід. Іноді результат дуже смішний, особливо якщо навести камеру на незвичайні об'єкти.

В даному проекті використовуються програми MaryTTS, Tone.js і Google Cloud Vision API.

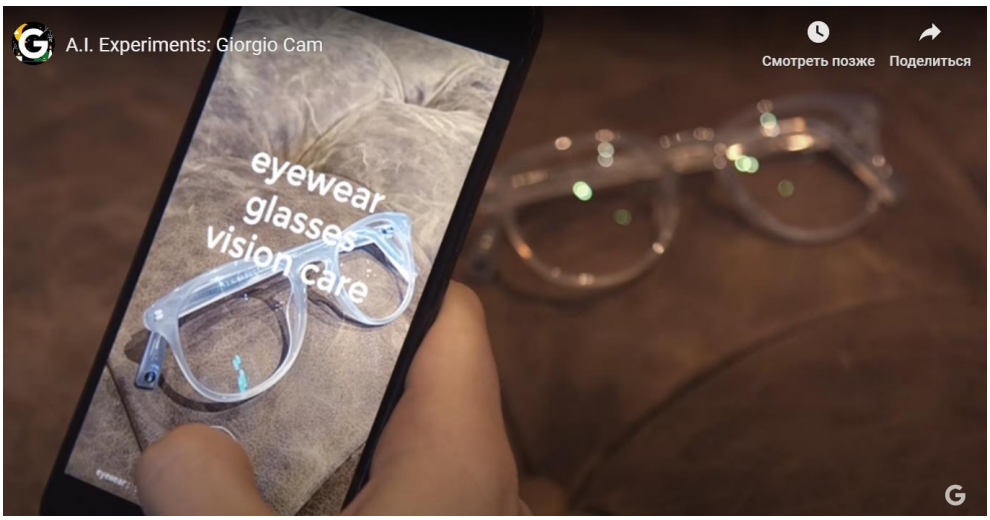


Рис. 3. Розпізнавання об’єктів Giorgio Cam

4. Реставрація старих фотографій онлайн за допомогою ШІ VanceAI

VanceAI Photo Restorer може допомогти відновити старі фотографії на 100% автоматично, використовуючи технологію відновлення фотографій AI, щоб видалити подряпини зі старих фотографій в Інтернеті, а також розриви, плями та сепію.

За допомогою VanceAI Photo Restorer можна відновити старі фотографії та покращити їх для отримання чітких і красивих ефектів. Додайти чіткість і природний колір старим фотографіям, відновлюючи їх.

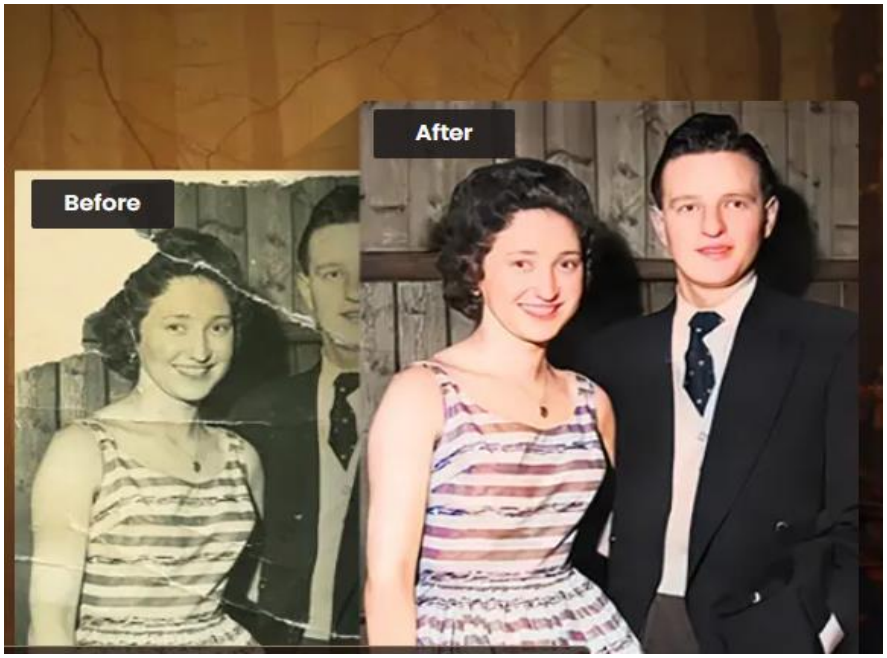


Рис. 4. Приклад результату вимконаног оексперименту
з застосуванням VanceAI Photo Restorer

5. Розфарбовування чорно-білих фотографій за допомогою ШІ Colourise

Colourise.com — це сервіс на основі штучного інтелекту, який спрощує весь процес розфарбовування. За допомогою Colourise.com можна розфарбовувати чорно-білі фотографії на 100% автоматично. Завдяки технології розфарбовування штучного інтелекту та глибокому навчанню AI Photo Colorizer дозволяє розфарбовувати фотографії за лічені секунди.

Рис. 5. Приклад результату вимконаного експерименту
з застосуванням Colourise

6. Покращення якості фотографій Let's Enhance

Let's Enhance – український стартап, сервіс по обробці зображень, який за допомогою нейромереж збільшує здатність знімків, відновлює деталі і підвищує чіткість. Let's Enhance запустили в листопаді 2017 року і на сьогодні сервіс обробив більше мільйона фотографій.

В основі сервісу покладено кілька об'єднаних нейронних мереж. Щоб навчити нейронні мережі, надають сотні тисяч знімків парами в низькій і високій роздільності. Алгоритм нейронної мережі навчений на великій базі знімків, яка завдяки знанням типових об'єктів і текстур вміє відновлювати деталі і зберігати чіткі лінії і контури оброблюваних зображень. Let's Enhance може не лише збільшувати розмір фотографії в чотири рази, але й видаляти шуми і артефакти стиснення на знімках формату JPEG, домальовувати відсутні дрібні деталі, роблячи картинку максимально реалістичною (рис. 7). Для ефективної обробки за часом і витратами, сервіс використовує потужності відеокарт - обробляти дані на CPU невигідно.

Для пересічних користувачів встановлено обмеження в 15 мегапікселів і 15 мегабайт для кожного завантаження. Користувачі з платною підпискою на

послуги сервісу мають максимальний пріоритет в обробці зображень і можливість завантажувати картинки з роздільною здатністю до 30 мегапікселів.

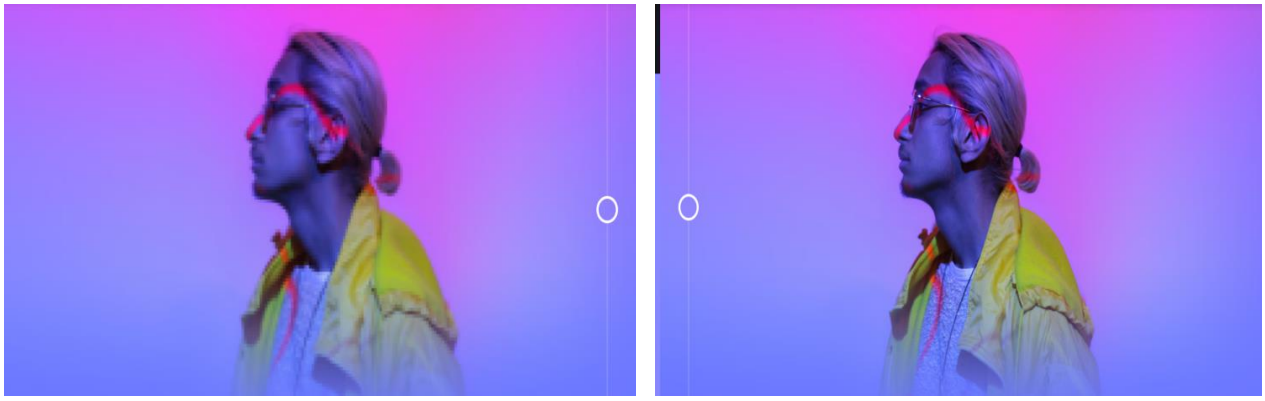


Рис. 6. Результат використання сервісу
по обробленню зображень Let's Enhance

7. Анімація фотографій Deep Nostalgia

Сервіс MyHeritage розробив функцію «Deep Nostalgia» для «оживлення» людей на фотографії. Нейромережа здатна зробити анімацію як з сучасних цифрових так й зі старих чорно-білих знімків. Щоб оживити портрет, потрібно зареєструватися на сайті та завантажити фотографію. Алгоритм оброблення робить статичний знімок, і у людини на зображенні з'являється миміка – вона почне повертати голову, моргати і посміхатися.

Рис. 7. Результат використання сервісу анімація
фотографій Deep Nostalgia

8. Накладання художніх фільтрів Dreamscope

Сервіс Dreamscope від компанії Lambda Labs дозволяє застосовувати до зображень до 19-ти різних фільтрів, перетворюючи будь-яку фотографію в психоделічну картину.

Програма заснована на розробках інженерів Google, які експериментували з новими системами розпізнавання зображень. Технологія

Deep Dream підсилює на фотографіях ті елементи, які нагадують їй знайомі об'єкти. Наприклад, якщо програма навчена розпізнавати обличчя, то на всіх зображеннях, будь то зоряне небо або будівля, вони бачитиме фрагменти людських обличь і відтворювати їх на фотографії (рис. 8).

Інженери виклали свою розробку у відкритий доступ в вигляді коду, що дозволило будь-якому користувачеві, знайомому з програмуванням, використовувати систему для реалізації своїх ідей. Додаток Dreamscope є доступним для простих користувачів, тому експериментувати з програмою можна до нескінченності, застосовуючи фільтри до зображень багато разів.

Рис. 8. Результат використання сервісу Dreamscope

9. Перетворення фотографії на картину Deepart

Німецькі вчені розробили алгоритм, який з великою точністю імітує будь-який графічний стиль і створює оригінальні картини на основі користувацьких зображень. Сервіс базується на технології нейронних мереж, що традиційно використовують для роботи з графікою (рис. 9). Сервіс працює не лише з картинами, а й з архітектурною графікою.

Щоб створити стилізоване зображення, потрібно завантажити фотографію і вибрати стиль. Готовий результат сервіс надсилає на електронну пошту, оскільки обробка займає кілька хвилин і на сервісі довга черга. Середній час відповіді - близько 10 хвилин.

Для тих, хто не бажає чекати, розробники сервісу пропонують кілька варіантів платних підписок, що дозволяють не тільки звести до мінімуму час рендерингу шедеврів цифрового мистецтва, а й забирання обмеження на розмір вихідних зображень.

10. Видалення фону Remove.bg

Безкоштовний сервіс, що дозволяє видалити фон на фотографіях без використання графічних редакторів. Після завантаження зображення система

автоматично, з використанням алгоритмів штучного інтелекту виділяє об'єкти на передньому плані і прибирає все зайве. Наразі алгоритми краще справляються з видаленням фону з фотографій, на яких зображені люди (рис. 11). Однак інструмент може працювати і іншими об'єктами на передньому плані, якщо вони чітко визначені. Також застосовують додаткові алгоритми, що покращують якість дрібних деталей.

До завантаження приймаються картинки будь-якого розміру, підсумковий варіант зображення (файл формату PNG з прозорим фоном) обмежений розміром 500 на 500 пікселів.

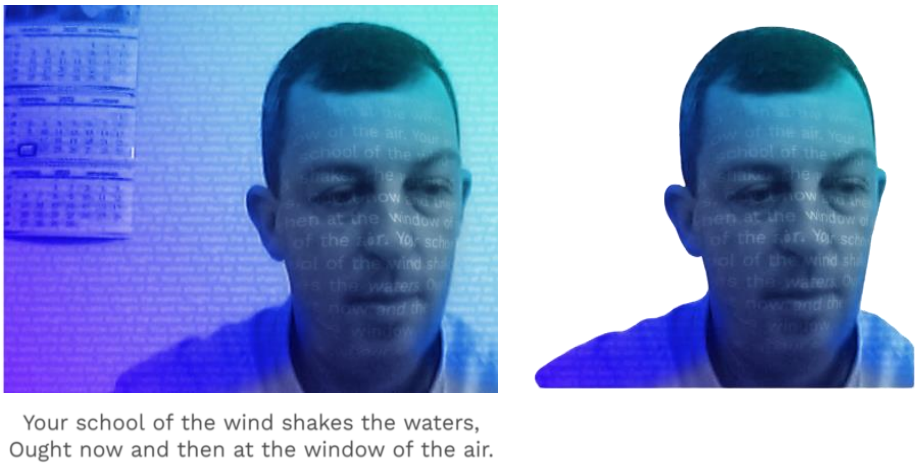


Рис. 9. Результат використання сервісу Dreamscope

11. Створення логотипів Looka Logo Maker

Сервіс використовує алгоритми TensorFlow від Google. Система опитує користувача про нову компанії, її слоган, спеціалізацію, колірну гаму, іконки. Щоб більше догодити користувачеві пропонують відмітити ті логотипи, стиль яких подобається.

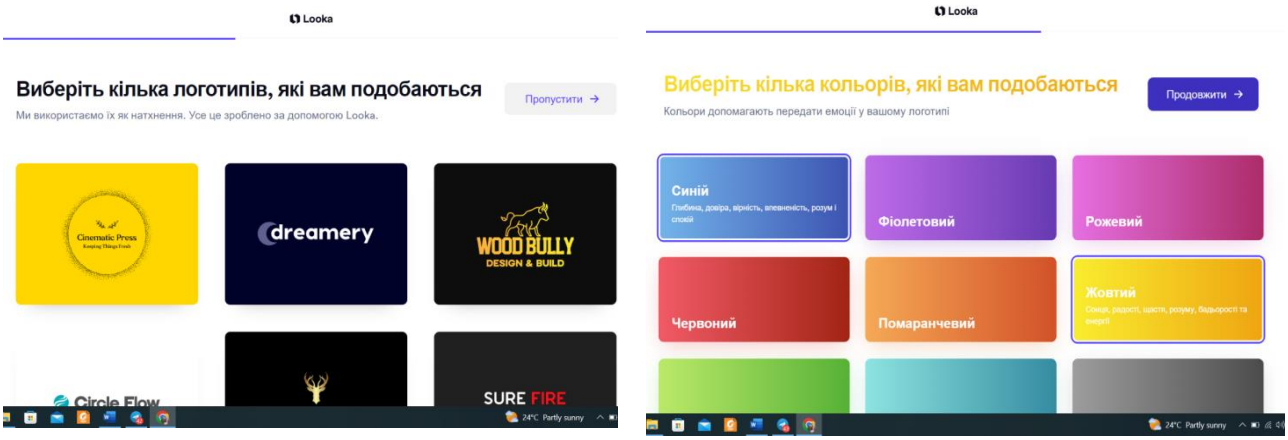


Рис. 9. Результат використання сервісу Looka Logo Maker
під час створення логотипу

12. Підбір музичного супроводу до зображення Imaginary Soundscape

Японська студія Qosmo розробила нейромережу Imaginary Soundscape, яка підбирає озвучення до завантажених фотографій чи зображень. Наприклад, до фотографії дитини в ліжечку нейромережа підбере дитячий плач, до зображення станції метро - звук потягу, до знімку пляжу - шум хвиль.

Люди при погляді на фотографію можуть уявити звуки, що в реальності супроводжують зображення: пейзаж пляжу може нагадати про звук гуркоту хвиль, жвава вулиця – з вуки автомобілів і вуличної реклами. «Imaginary Soundscape» Уявний звук) – це мережна звукова інсталяція, що сфокусована на цьому несвідомому досвіді. Користувач може пересуватися по Google Street View і занурюватися в уявні звукові ландшафти, що створені за допомогою моделей глибокого навчання.

Ця робота заснована на розробці крос-модальної методики пошуку інформації, такої як зображення-аудіо, текст-зображення, з використанням глибокого навчання. При наявності відео входів система була навчена двом моделям: одна добре налагоджена, попередньо навчена модель розпізнавання зображень обробляє кадри, а інша згорткова нейронна мережа зчитує звук як зображення спектрограми, еволюціонуючи таким чином, що розподіл вихідного сигналу стає якомога ближче до першого.

Після навчання дві мережі дозволяють отримувати найбільш доречний звуковий файл для сцени з великого набору звукових даних про навколишнє середовище.

Звукові ландшафти, які генеруються штучним інтелектом, зазвичай виправдовують очікування, але іноді ігнорують культурний і географічний контекст (наприклад, шум хвиль на крижаному полі Гренландії). Ці відмінності і помилки змушують розробників задуматися над тим, як працює уява і наскільки плідне навколишнє звукове середовище.

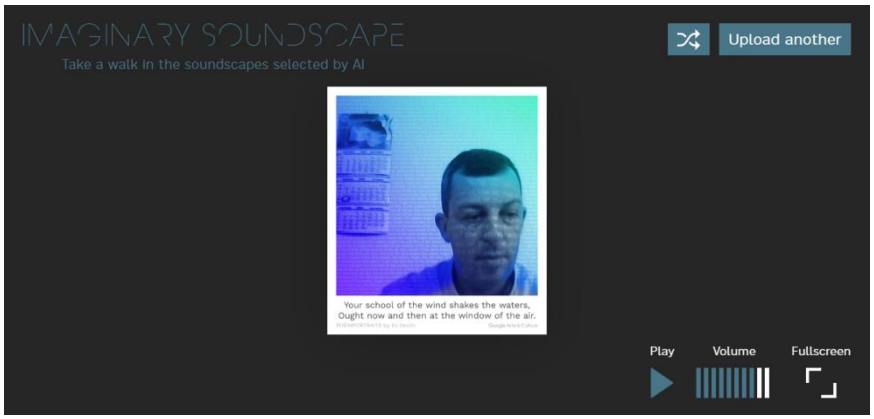
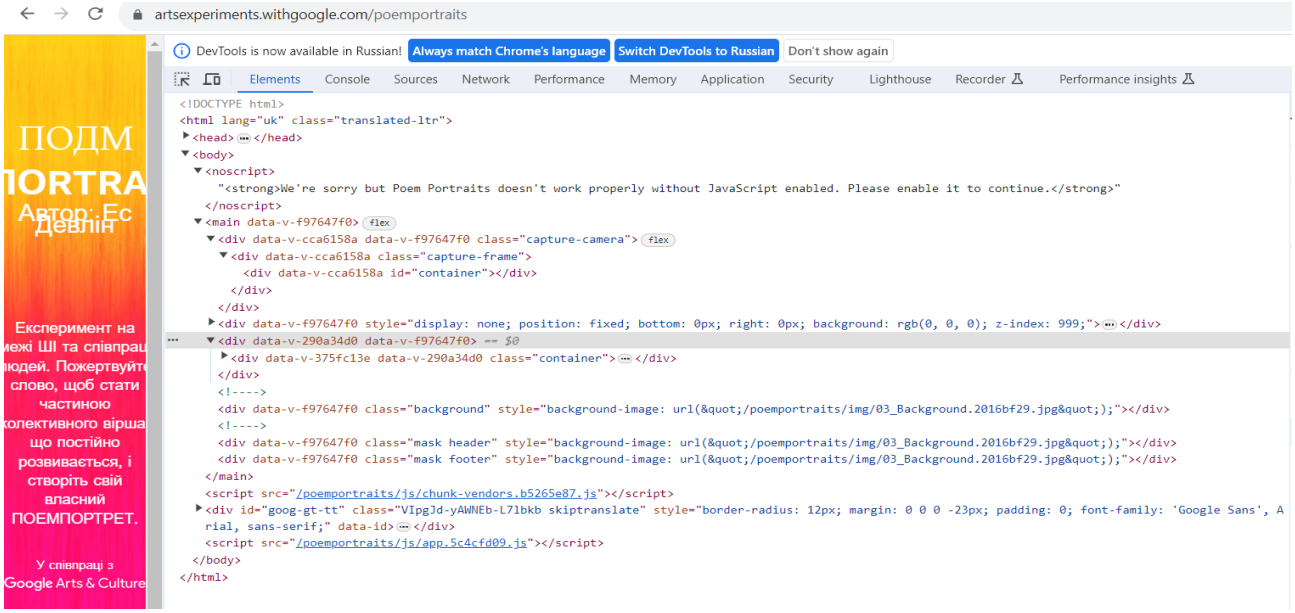


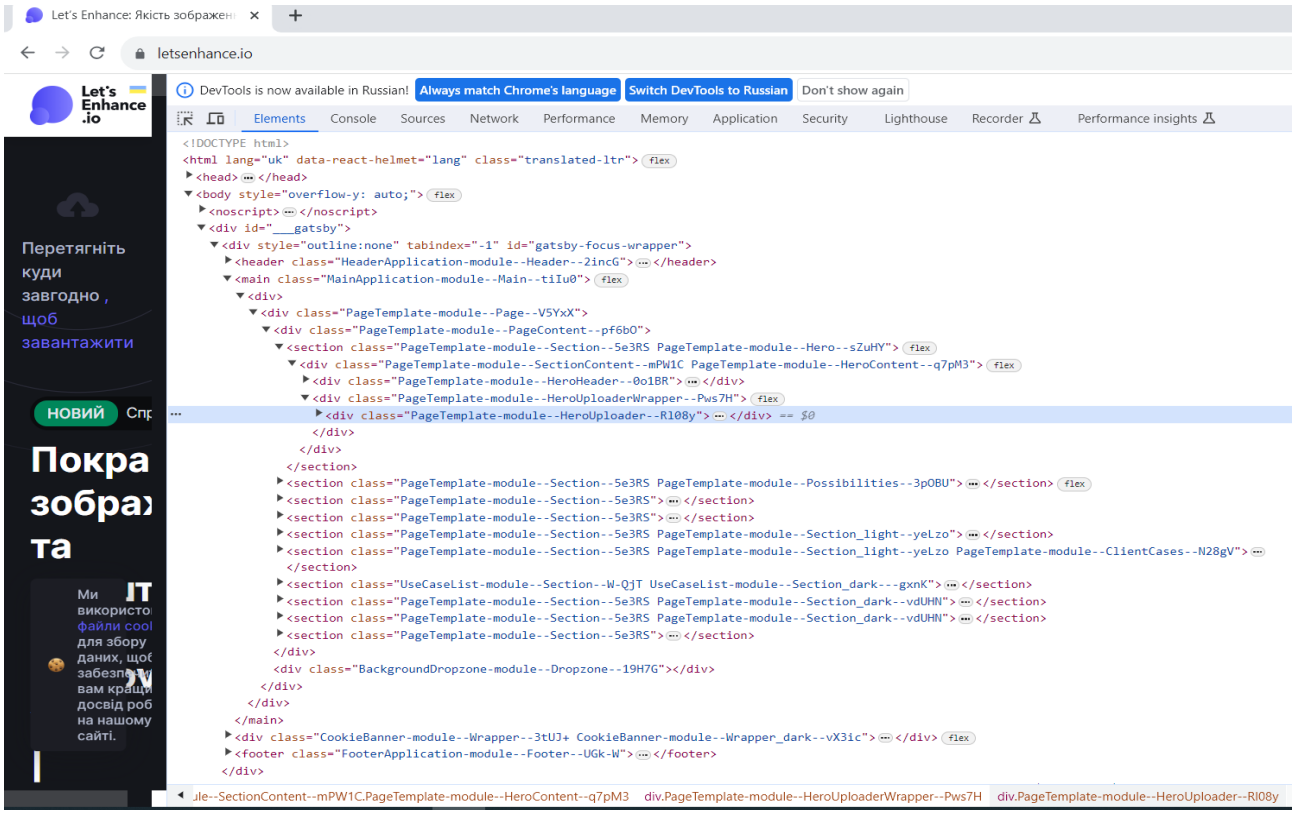
Рис. 10. Результат використання сервісу Imaginary Soundscape

Приклади програмних кодів інтелектуальних веб-сервісів

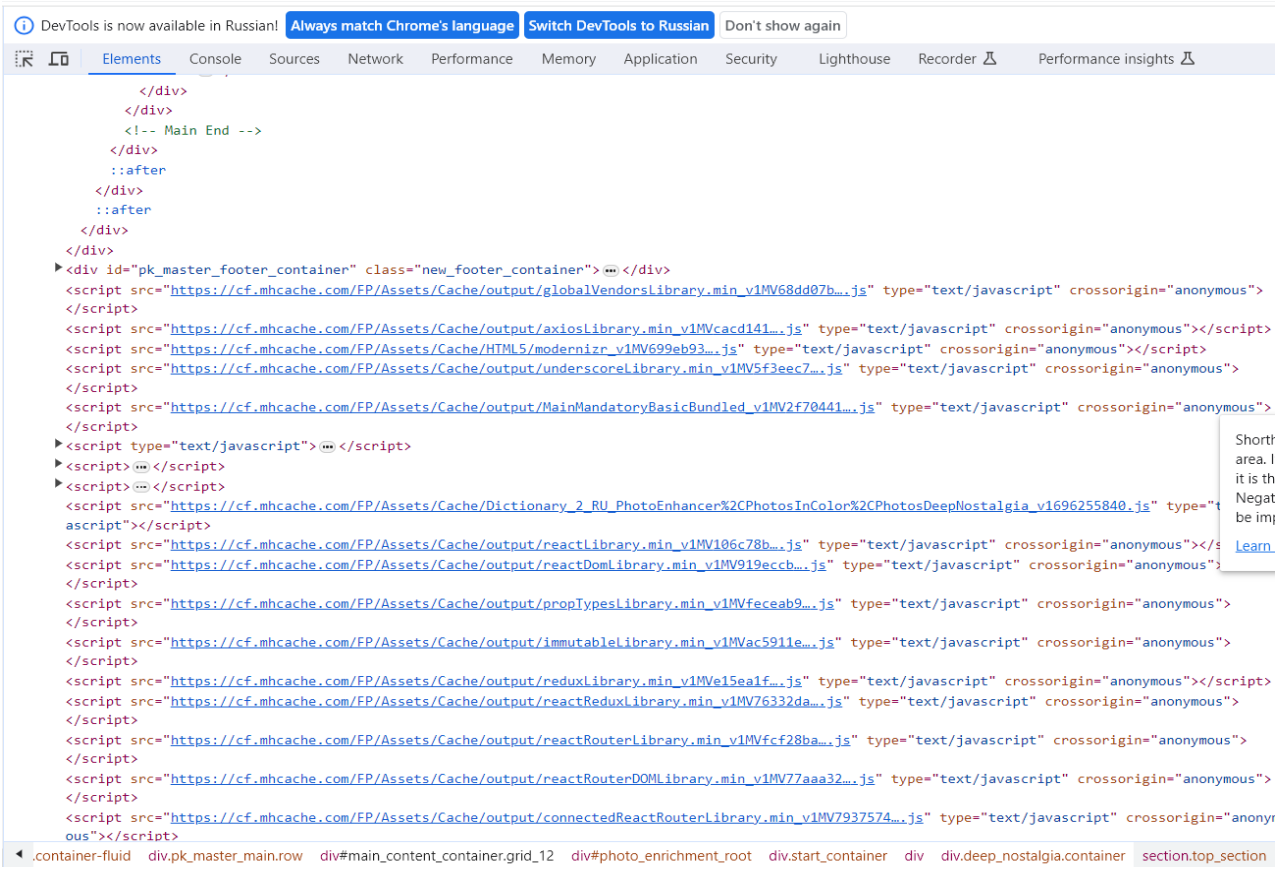
PoemPortraits



Let`s Enhance



Deep Nostalgia



Хід виконання роботи:

1. Послідовно випробувати наведені сервіси, ознайомитися з інтерфейсом, можливостями сервісу і обмеженнями для пересічних користувачів. Здійснити низку експериментів і порівняти результати.

2. Проаналізувати отримані результати і з'ясувати причини відповідних розбіжностей.

3. Віднайти інші сервіси (2-3 реалізації), де використано інтелектуальні технології. Описати суть сервісу, зазначити використані технології, здійснити низку експериментів, проаналізувати результати та зробити висновки.

4. Ознайомитися з текстами відкритих кодів наведених в роботі інтелектуальних сервісів.

5. По результатах роботи оформити звіт.

Зміст звіту практичної роботи №1

1. Назва та мета виконання лабораторної роботи.

2. Скріншоти виконання робіт з вказуванням назви сервісу та його особливостей. На скріншоті мають бути результати, що підтверджують особу студента (особисті фотографії, надписи тощо).

3. Навести типові приклади текстами відкритих кодів наведених в роботі інтелектуальних сервісів.

4. Аналітичні висновки щодо властивостей сервісів та отриманих результатів.

Практична робота №2

Тема. Голосові сервіси

Мета роботи. Ознайомитися з сервісами, що надають можливість автоматизованого розпізнавання тексту, синтаксичного та морфологічного аналізу, щоб виявити сенс. Дані сервіси застосовують на етапі попередньої обробки тексту і вони є необхідними модулями у потужних комплексах діалогових систем, що мають голосовий інтерфейс: аналіз, синтез, розуміння написаного чи сказаного. Набути практичних навичок роботи з програмами розпізнавання та синтезу мови, Провести низку експериментів для виявлення особливостей сервісів щодо оброблення людської мови.

Теоретичні відомості

Голосовий інтерфейс (VUI – Voice-User Interface) – це програмний продукт, який за допомогою голосової або мовної платформи дозволяє взаємодіяти користувачеві і комп'ютеру, запускаючи автоматизовані процеси. Завдання таких інтерфейсів - розпізнати і генерувати людський голос.

Голосові інтерфейси використовують, коли вводити текст складно або незручно. Наприклад, під час водіння автомобіля користувач може проговорити свій запит, продиктувати потрібну адресу або якщо користувач виконує дуже багато завдань і не може сконцентруватися на одній.

Базові технології голосового інтерфейсу:

Голосове введення. Запити вимовляються голосом, а не вводяться за допомогою клавіатури або графічних елементів екранного інтерфейсу.

Природна мова. Користувачі не повинні обмежуватися використанням певного, оптимізованого для комп'ютера словника або синтаксису, але можуть формувати введення будь-якими способами, подібно до розмови з людиною.

Голосове виведення. Інформація вимовляється голосом, а не виводиться на екрані.

Інтелектуальна інтерпретація. Для справжнього розуміння запитів користувача голосовий інтерфейс повинен використовувати додаткову

інформацію, таку як контекст використання або дії, які користувач здійснював раніше.

Сприяння. Голосовий інтерфейс вчиняє дії, необхідні для виконання завдання користувача, які користувач не запитував.

Голосовий інтерфейс Google

Завдяки голосовому інтерфейсу можна диктувати запити в клієнтську програму на пристрої, замість введення тексту в пошуковий рядок. Щоб транскрибувати продиктовані слова в написаний текст, Google надсилає вислови на сервери, де використовується технологія розпізнавання шаблонів. Для того, щоб навчити систему краще розпізнавати правильні пошукові запити, Google зберігає вислови, щоб покращувати служби, зокрема: дані про мову, країну, вислів і припущення системи про сказане (рис. 1). Збережені аудіо дані не містять ідентифікатор облікового запису Google, якщо користувач цього не вказав.

Рис.1. Голосовий інтерфейс Google

Для кожної мови голосовий інтерфейс Google збирає голосові фрагменти, які дозволяють створити моделі мови і забезпечити коректну роботу сервісів. Google має базу аудіо образів, що промовляються носіями мови та відрізняються за акцентами, віком і індивідуальними особливостями. Часто вживані фрази вимовляють в різних акустичних умовах, наприклад, в ресторані, на вулиці або в машині. Для кожної мови Google створює словник, що містить більше мільйона розпізнаних слів.

Сервіс функціонує на основі системи Speech Input API, завдяки якій і реалізується голосове управління. Сервіс на даний момент втілено в Google Пошук, Google Перекладач, Cmail, Google Docs, Google Калькулятор.

Голосовий пошук Google

Voice Search – це розширення для Google Chrome, що дозволяє здійснювати пошук або інші дії в Інтернеті за допомогою голосу. На сторінці Google в рядку пошуку зображено іконку мікрофону. Користувач має

натиснути на нього і вимовити голосно і чітко фразу або слово. Для отримання озвучених відповідей потрібно використовувати мову відповідно до мовного інтерфейсу Google Chrome.

У разі запиту про помітні чи загально визнані об'єкти буде озвучено інформацію, яка береться з «Графу знань» Google - бази, яка містить інформацію про різні об'єкти, події та їхні зв'язки між собою. Відомості з «Графу знань», зазвичай, виводиться праворуч від результатів пошуку і надає стислу інформацію за запитом, який ввів користувач.

Це може бути, наприклад, інформація про актора, включаючи фільми, в яких він знявся, і дату народження. Озвученню буде підлягати, наприклад, відповідь на прості запитання «Скільки доларів буде в 100 гривнях», «яка столиця Франції», «хто такий Коельо».

Голосовий перекладач Google

Google реалізував універсальний перекладач принципово нового типу, який дозволяє користувачам, що спілкуються на різних мовах, говорити один з одним в режимі реального часу, причому саме «говорити», а не «листуватися». Іншими словами перекладач розпізнає мову, перекладає отриманий в результаті цього текст і відтворює його іншою мовою (рис. 2).

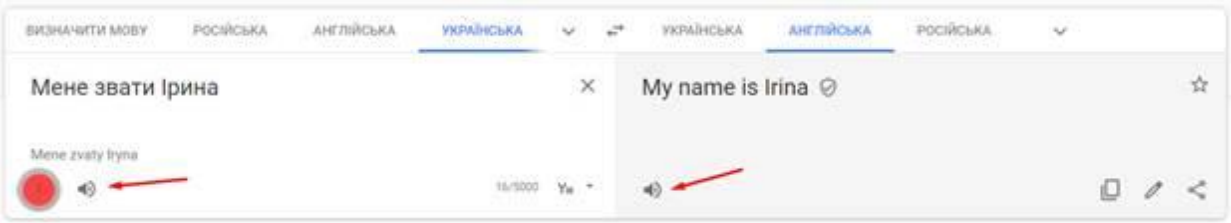


Рис. 2. Інтерфейс голосового перекладача Google

Для перекладу певної фрази достатньо натиснути на зображення мікрофона в програмі, сказати в мікрофон потрібні слова і програма автоматично надсилає записану мову на сервери Google, де відбувається розбір звукового файлу і переклад фрази. Після текстового перекладу можна прослухати вимову перекладу і оригінального тексту (синтезований жіночий голос). Правильному перекладу можуть перешкодити такі фактори як акцент, чіткість вимови і сторонні шуми.

Онлайн-програми розпізнавання мови

Розпізнавання мови розвивається значними темпами. Якщо ще кілька років тому впізнавання людської мови не перевищувала кількох відсотків, то сьогодні комп'ютер спокійно, навіть в досить шумній атмосфері правильно розпізнає більшу частину промови. Користуючись API від провідних платформ

розпізнавання голосу створюється багато сервісів, що пропонують послуги переведення голосової промови у друкований текст: онлайн-блокноти, додатки в телефонах тощо.

Speechlogger

Speechlogger – програма для розпізнавання мови і миттєвого голосового перекладу. Використовує технологію Google "голос в текст" для отримання кращих результатів. Виконана як веб-додаток з автоматичної розстановкою розділових знаків, автоматичним збереженням позначки часу, можливістю редагування тексту, транскрипції аудіо файлів, опцією експорту (в текст і записи) і багатьма іншими функціями (рис. 3).

Рис. 3. Інтерфейс програми Speechlogger

Speechnotes

Speechnotes – це онлайн блокнот з функцією мовного введення з залученням передової технології розпізнавання мови. Сучасні технології разом з підключенням вбудованих інструментів (автоматичних або ручних) забезпечують точні результати, ефективність, продуктивність і комфорт. Працює онлайн у браузері Chrome, не потребує завантаження, встановлення та реєстрації (рис. 4).

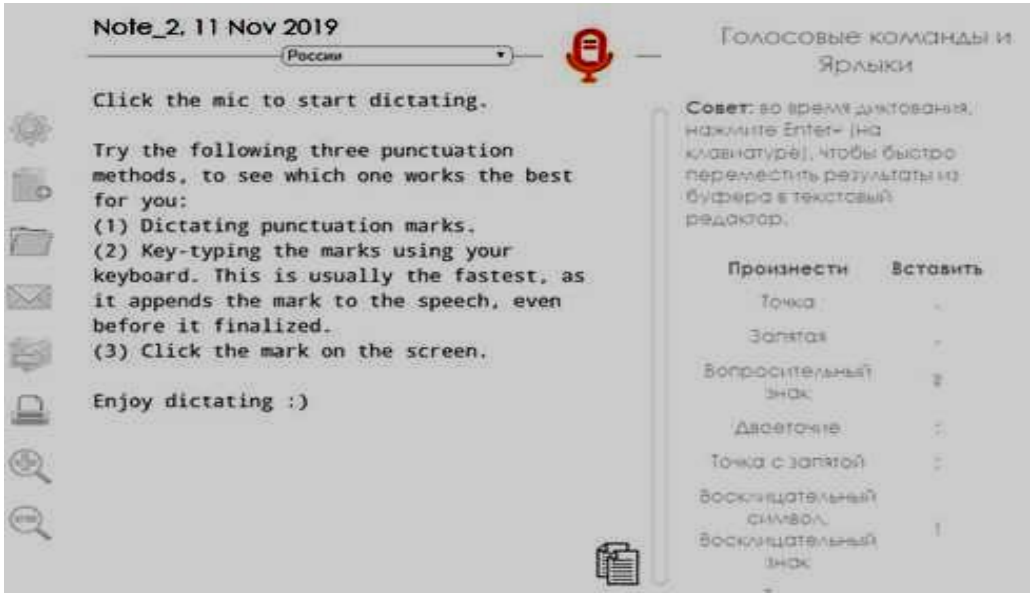


Рис. 4. Інтерфейс програми Speechnotes

Онлайн-програми синтезу мови

Acapela

Один з найвідоміших синтезаторів мови, що розмовляє на 30 мовах (рис. 6). Тексти можна зачитувати чоловічим або жіночим голосом. На безкоштовне використання є обмеження: в браузерній версії можна відтворити не більше 300 символів. Для використання повного функціоналу необхідно завантажити платну програму - вона доступна на Windows, Linux, Mac, а також на мобільних ОС Android і IOS.

Рис. 6. Інтерфейс програми Acapela

Oddcast

Багатомовний синтезатор, тексти зачитують чоловічим або жіночим голосом анімовані диктори (рис. 7). На безкоштовне використання є обмеження: в браузерній версії можна відтворити не більше 300 символів.

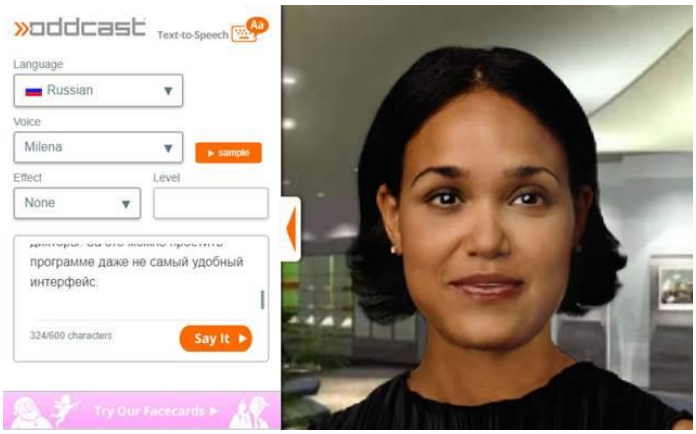


Рис. 7. Інтерфейс програми Oddcast

Linguatec

Багатомовний синтезатор, тексти зачитують чоловічим або жіночим голосом анімовані диктори (рис. 8). На безкоштовне використання є обмеження 250 символів.

Рис. 8. Інтерфейс програми Linguatec

Text-to-Speech

Синтезатор мови з широкими налаштуваннями: можна задавати швидкість мовлення, розмір шрифту і машинний переклад. Доступна навіть екранна клавіатура для людей з обмеженими можливостями (рис. 9).

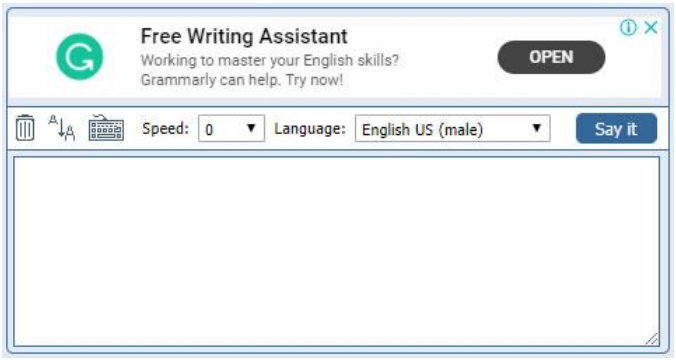


Рис. 9. Інтерфейс програми Text-to-Speech

Ispeech

iSpeech.org – програмне забезпечення перетворення тексту в промову, що доступне для популярних платформ. Сервіс надає безкоштовний онлайн-інструмент, що швидко перетворює друкований текст в мовний фрагмент в цифровому аудіоформаті.

Сервіс має легкий, дружній інтерфейс користувача, озвучення відбувається як чоловічими голосами на більш ніж 30 мовах з акцентом для конкретної країни (рис. 6). В безкоштовному додатку є обмеження на кількість слів, іноді голос може звучати як роботизований.

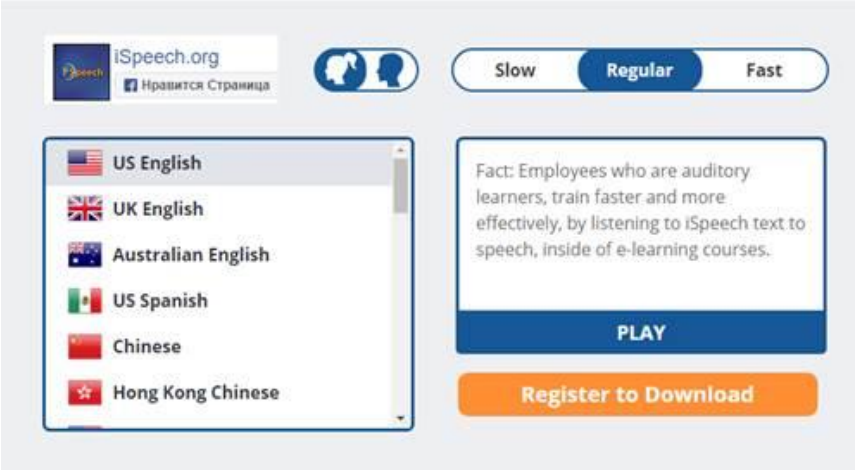


Рис. 10. Інтерфейс програми iSpeech

Діалогові системи та чат-боти

Одним з перспективних напрямків в області глибокого навчання є діалогові системи, які наразі представлено в 2 видах:

- **Примітивні – чат боти**. Функціонально обмежені системи, які відповідають тільки на певні команди, що занесено в їх пам'ять . Якщо ввести не ту фразу, на яку запрограмовано чат-бот, то потрібної відповіді не буде.

- **Просунуті** – голосові асистенти. Системи з штучним інтелектом, їх робота заснована на машинному навчанні. Для асистентів не потрібно спеціально підбирати фрази для того, щоб бути зрозумілим. Вони розуміють живу мову, постійно навчаються, аналізують всі запити і слова та отримують нові знання з діалогів. Такий віртуальний асистент не використовує заготовлені відповіді, а будує їх з речень, що дотичні до теми питання. Вважається, що поки віртуальні асистенти зі штучним інтелектом знаходяться на початковому етапі розвитку, але вже за кілька років спроможні замінити реальних фахівців в багатьох сферах, наприклад, програмістів, HR-фахівців, офіс-менеджерів, маркетологів-аналітиків тощо.

Віртуальний асистент: основні функції та вміння

Сьогодні голосові помічники стали невід'ємною частиною життя. З кожним днем все більше людей вибирає віртуальних асистентів, замінюючи мишку та клавіатуру. Штучний інтелект допомагає вирішувати прості завдання за допомогою голосового діалогу. Після введення інформації, помічник розпізнає сказану мову і починає функціонувати. Для того, щоб асистент зрозумів і виконав запит, слід говорити чітко та повільно. Асистент може підказати маршрут, новини дня, знайти музику, показати погоду, відповісти на просте запитання.

Розумні асистенти покликані спростити взаємодію користувачів з високотехнологічними складними пристроями і мають 5 основних функцій:

- **Голосове введення.** Дозволяє людині ставити команди машині без використання додаткових інструментів
- **Інтерпретація природної мови.** Користувачеві не треба вчити команди, підбирати правильні слова і вирази для команд. Асистенти навчені на величезних вибірках діалогів розуміти природну повсякденну мову людини.
- **Голосова відповідь.** Замість тексту на дисплеї асистенти озвучують відповідь, що ще більше спрощує взаємодію.
- **Вміння враховувати контекст.** Розумні асистенти враховують ймовірний зміст та контент діалогу за значенням слів, поведінку користувача, історію минулих запитів, ситуацію, в якій відбувається запит, щоб точніше зрозуміти намір користувача.
- **Самостійні дії.** Система самостійно виконує певні завдання, ґрунтуючись на попередній поведінці користувача.

Вміння враховувати контекст і виконувати самостійні дії вимагають від асистента постійного вивчення користувача, його поведінки і дій в різних ситуаціях. Саме так асистент вчиться змінювати свою поведінку і оптимізувати його під потреби конкретної людини. Асистенти не завжди використовують всі

5 функцій. Так, якщо екран пристрою вільний, то віртуальний помічник може вивести текст відповіді на екран замість голосової озвучення.

Впровадження віртуальних асистентів має дві незаперечні переваги:

- Можливість створювати складні інтерфейси без шкоди для зручності використання. Достатньо навчити помічника швидко знаходити потрібну інформацію для користувача за голосовою командою. Тоді сам користувач не зіткнеться з проблемою «довгого шляху» до потрібної функції або даними.
- Підвищити точність взаємодії. Завдяки здатності прогнозувати дії користувача розумні помічники знаходять точну відповідь.

Розумні асистенти перебувають на початковій стадії розвитку. Вони здатні допомогти людині в ситуаціях, коли зайняті руки або коли голосовий запит набагато швидше відправити, ніж друкувати текст. Асистенти справляються лише з простими завданнями, використовують вбудовані функції не в повному обсязі, часто не взаємодіють зі сторонніми додатками.

Найпоширеніші голосові помічники

Cortana для Windows

Голосовий помічник створено компанією Microsoft та інтегровано в операційну систему. Призначений перш за все для Windows, але в якості застосунків працює на платформах iOS, Android, Xbox One, Microsoft Phone, Microsoft Band. «Cortana» допоможе систематизувати і розпланувати завдання на певний період, нагадає про виконання будь-яких дій, за запитом надає інформацію. Має вбудований функціонал для відповідей на загальні питання, використовуючи пошук Bing. У функціонал входить прокладання маршруту, інформація про стан доріг, нагадування про зустрічі. Вводити інформацію можна за допомогою голосу та клавіатури в текстовій формі. Асистент підтримує розмову: співає пісні, надсилає анекдоти.

До особливостей можна віднести таку функцію, як передбачення бажань користувача. Якщо надати доступ до особистих даних, віртуальний помічник від Microsoft буде «підлаштовуватися» під власника, постійно аналізуючи його дії: місця, в яких подобається перебувати, уподобання в різних сферах, інтереси, хобі та багато іншого.

Віртуальний асистент Cortana тісно пов'язаний з операційною системою і може керувати Windows 10 та окремими додатками під час роботи: допоможе прочитати електронні листи, відстежити місце розташування, перевірити список контактів, стежити за календарем, керувати музикою, охоплюючи

численні музичні програми та контролюючи звук відповідно до власних уподобань.

Присутня можливість синхронізації асистента на кількох пристроях. Cortana буде підтримувати актуальність на декількох комп'ютерах одночасно.

Ok Google

Ok Google – голосовий помічник і одночасно частина пошукової системи. У програми є багато функцій: планування подій (встановлення нагадувань), відстеження поштового листування, перехід на певний сайт, пошук музичних композицій, знаходження адрес тощо. Особливість програми: після виконання команди, програма самостійно доповнює інформацію. Програма є безкоштовною і стабільно працює, її можна налаштувати під конкретного користувача. Асистент вбудований в браузер Google Chrome, доступний для ПК, Android, iOS.

Хід виконання роботи:

1. Послідовно випробувати наведені сервіси (https://www.victoria.lviv.ua/library/students/sss/theme8.html), ознайомитися з інтерфейсом, можливостями сервісу і обмеженнями для пересічних користувачів. Здійснити низку експериментів і порівняти результати.
2. Проаналізувати отримані результати і з'ясувати причини відповідних розбіжностей.
3. Ознайомитися з текстами відкритих кодів наведених сервісів в роботі.
4. По результатам роботи оформити звіт.

Зміст звіту практичної роботи №2

1. Назва та мета виконання практичної роботи.
2. Скріншоти виконання робіт (проведенняекспериментів) з вказуванням назви сервісу та його особливостей. На скріншоті мають бути результати, що підтверджують особу студента (особисті фотографії, надписи тощо).
3. Типові приклади відкритих кодів програм інтелектуальних сервісів.
4. Аналітичні висновки щодо властивостей сервісів та отриманих результатів.

Практична робот №3

Тема роботи. Нейромережа для класифікації Sharky Neural Networ. Застосування нейронної мережі “Teachable Machine”

Мета роботи. Ознайомитися з демонстраційним нейроемулятором Sharky Neural Network для дослідження роботи нейронної мережі. Набути практичних навичок з користування програмою, дослідити параметри, що впливають на якість класифікації, застосувати різні можливості для класифікації складних фігур.

Нейромережа для класифікації Sharky Neural Network

Sharky Neural Network – це комп'ютерна програма фірми SharkTime Software для демонстрації можливостей нейромережного класифікатора. Програма freeware, працює під ОС Windows різних версій.

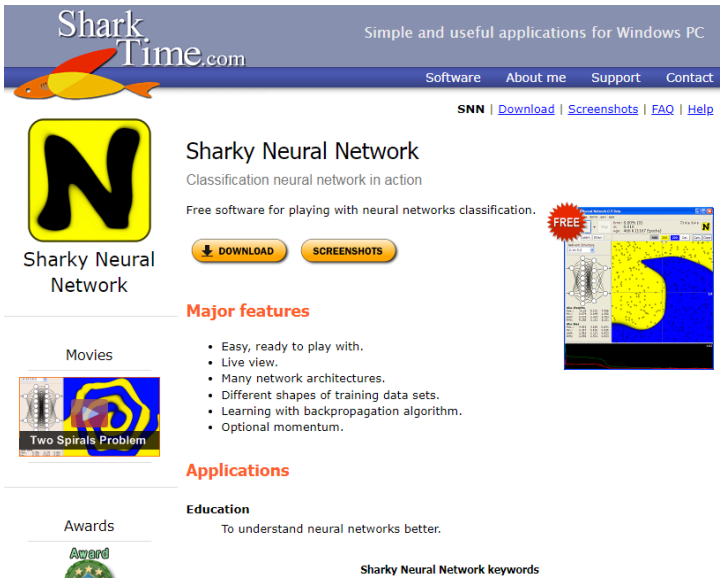


Рис. 1. Інтерфейс програми Sharky Neural Network

Програма реалізує нейронну мережу типу багатошарового перцептрона, що призначена для класифікації 2D-точок в два різних класи: жовтий і синій. Кожна множина 2D-точок представляє геометричну фігуру (форму) – коло, квадрат, діамант, хвилю, місяць або іншу фігуру. Програма при класифікації не визначає форму, вона просто розподіляє всі крапки на дві групи: сині та жовті. Геометрична форма розпізнаваних фігур при цьому проявляється при візуалізації результату класифікації.

Вихідні дані можна завантажити тільки у вигляді визначеного образу з кількох наявних заготовок (xor, circle, square, diamond, ring, moon, wave тощо).

На сайті можна завантажити додаткові файли «AI.points», «cn.points», «N.points», «Two_Spirals_Cartesian.points» і Two_Spirals_Radial.points».

Програма дозволяє вносити зміни у вихідні дані: додавати, видаляти, завантажувати або зберігати точки. Комбінація клавіш Ctrl + Left Click дозволяє працювати в режимі spray (додавати групу розсіяних навколо кліку точок).

Хід виконання роботи

1. Ознайомитися з теоретичними матеріалами щодо нейронних мереж та їх застосування в задачах класифікації за посиланням (https://www.victoria.lviv.ua/library/students/sss/theme4.html).

2. На сайті SharkTime Software ознайомитися з документацією щодо нейроемулятора, завантажити та встановити додаток. Ознайомитися з інтерфейсом додатку і здійснити низку досліджень з параметрами, що встановлено за замовченням.

3. Відповідно до інструкції сформувати власний набір точок і здійснити експерименти по навчанню та використанню мережі. Змінити параметри та порівняти якість класифікації. Проаналізувати отримані результати і зробити висновки.

4. Провести експеримент з застосуванням нейронної мережі “Teachable Machine”: Навчіть модель класифікувати положення тіла. Для цього понадобляться готові зображення або веб-камера.

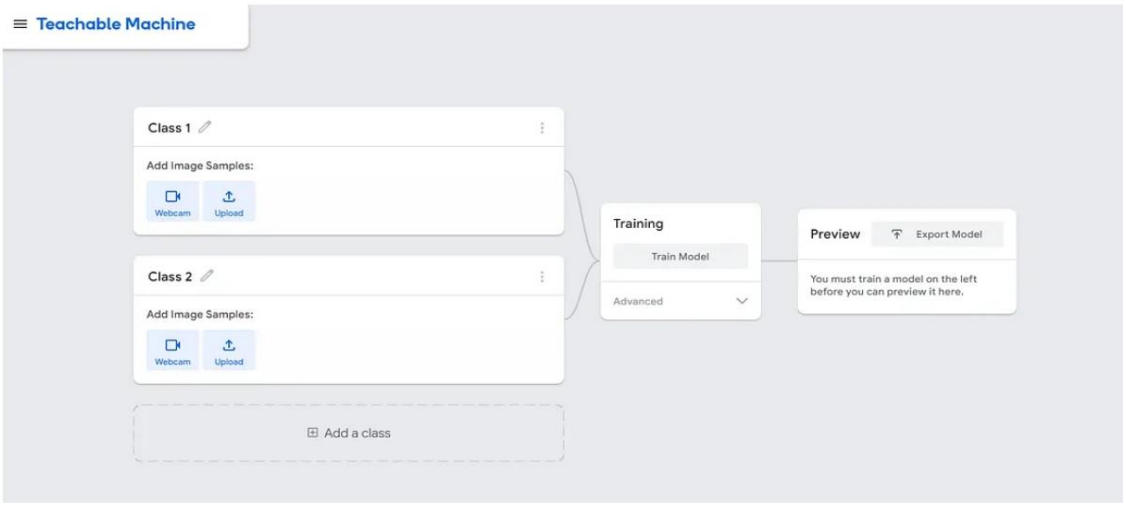
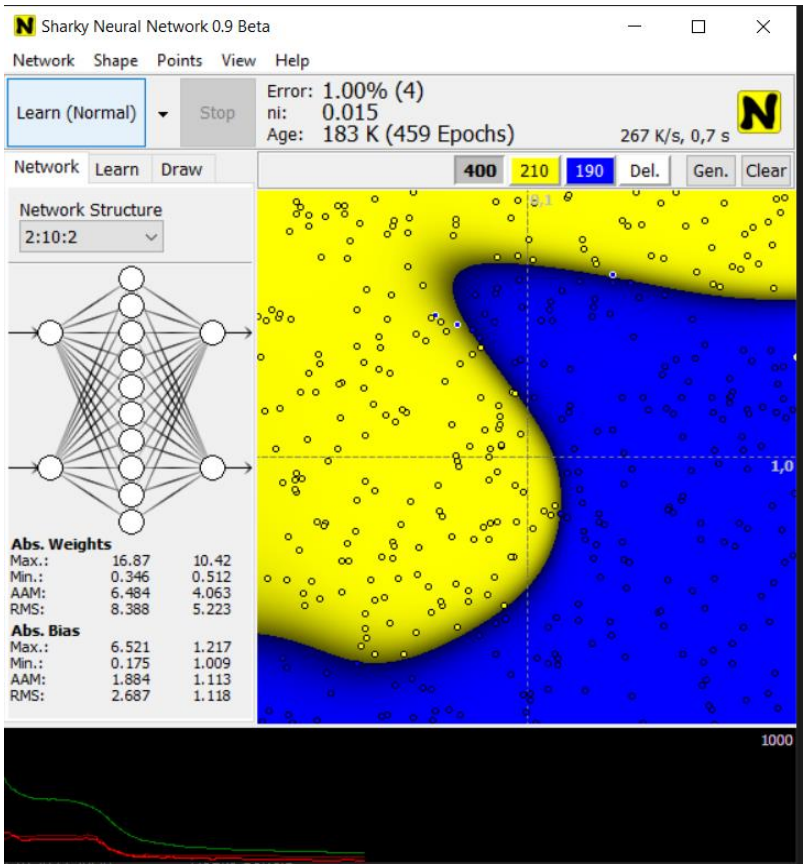


Рис. 2. Інтерфейс програми «Teachable Machine».

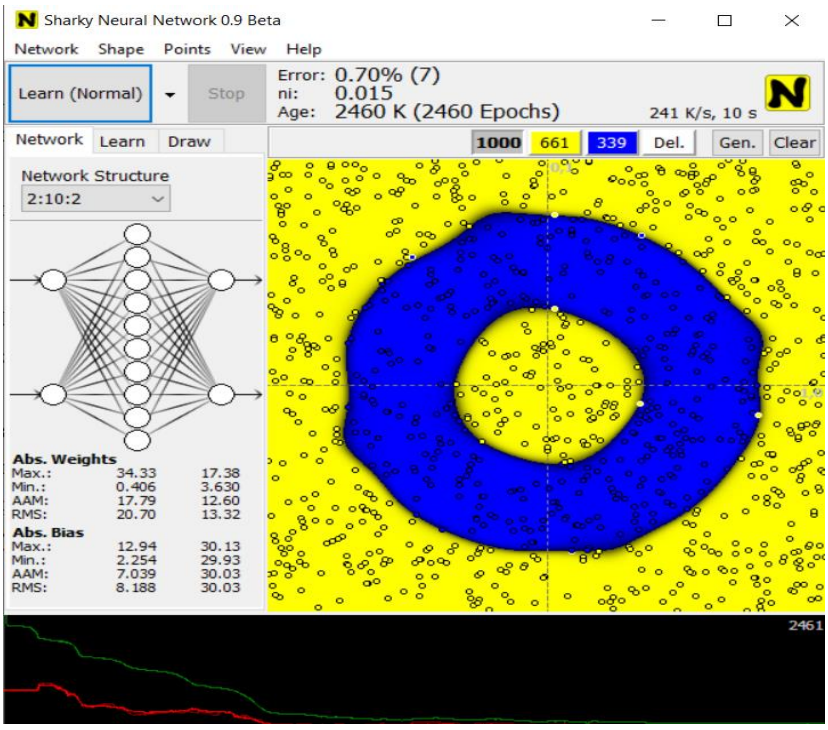
Зміст звіту

1. Назва та мета виконання лабораторної роботи.

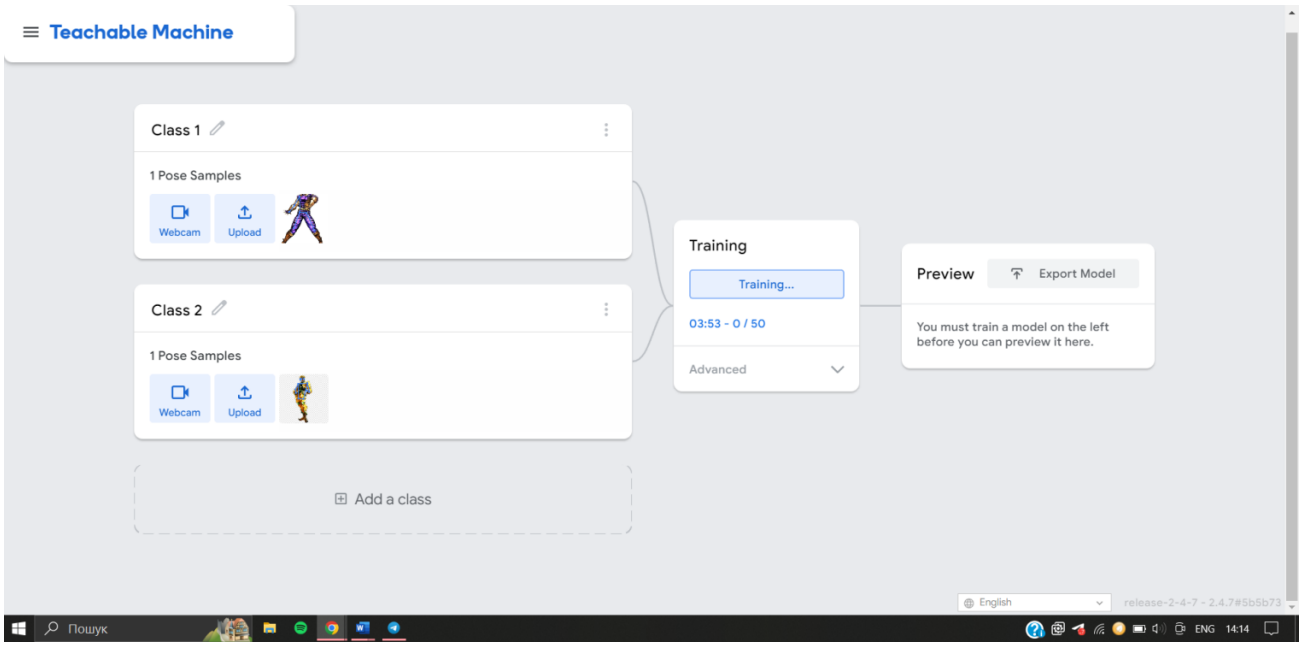
2. Скріни етапів виконання завдання з коротким описом. Характеристика основних параметрів налаштування.



Скрін програми з зміненою формою то більшою кількістю точок



Скрін програми Teachable Machine»



3. Аналітичні висновки щодо властивостей нейромережі Sharky та отриманих результатів.

Практична робота №4

Тема. Ознайомлення з хмарною платформою Google Colaboratory

Мета роботи: Вивчити структуру, засвоїти основні функції в online-середовищі хмарної платформи Google Colaboratory.

Загальні відомості

Google Colaboratory або Colab**,** це один хмарний сервіс від Google Research. Це IDE (Integrated Development Environment – Інтегроване середовище розробки), яка дозволяє будь-якому користувачеві писати вихідний код у своєму редакторі та запускати його з браузера. Зокрема, він підтримує мову програмування Python і орієнтований на завдання машинного навчання, аналіз даних, навчальні проєкти тощо.

Ця послуга, заснована на Jupyter Notebook**,** розміщено абсолютно безкоштовно за допомогою облікового запису Gmail, і він не вимагає налаштування, а також вам не доведеться завантажувати чи встановлювати Jupyter. Він запропонує вам обчислювальні ресурси для редагування та тестування вашого коду, наприклад GPU його серверів тощо. Очевидно, що як щось безкоштовне, Google Colaboratory не має необмежених ресурсів і не

гарантує їх, але вони змінюються залежно від використання, яке надається системі.

Важливо зазначити, що коли ви отримуєте доступ до Colab за допомогою свого облікового запису, ви отримуєте віртуальну машину, на якій ви можете запускати свій код, ізольований від інших користувачів і ресурсів. Тому ви можете відновити віртуальну машину до початкового стану, якщо у вас виникнуть проблеми. Це також означає, що якщо ви виконуєте деякий код у своїй віртуальній машині та закриваєте браузер, машини будуть видалені після певного періоду бездіяльності, щоб звільнити ресурси. Однак ви матимете свої блокноти в GDrive, якщо ви їх зберегли, або ви зможете завантажити їх локально (формат Jupyter з відкритим кодом .ipynb).

Можливості Google Colaboratory:

- Запуск кодів Python.
- Зберігання своїх проєктів на Google Drive (GDrive), щоб не втратити їх.
- Завантаження кодів з GitHub.
- Обмін блокнотами (текстом, кодом, результатами та коментарями).
- Імпортування блокнотів Jupyter або IPython.
- Завантаження будь-якого блокноту Colab локально з GDrive.

У записниках Colab можна поєднувати виконуваний код **і** форматований текст в одному документі, а також додавати зображення**,** HTML**,** LaTeX тощо. Коли ви створюєте власні записники Colab, вони зберігаються в обліковому записі Google Диска. Ви можете надавати доступ до записників Colab співробітникам або друзям – так вони зможуть коментувати або навіть редагувати їх.

З Colab ви отримуєте доступ до всіх популярних бібліотек Python для аналізу й візуалізації даних. У наведеній нижче клітинці з кодом використовуються бібліотеки numpy (дає змогу генерувати довільні дані) і matplotlib (для їх візуалізації). Щоб змінити код, просто натисніть клітинку й почніть редагувати.

У Colab можна імпортувати набір даних зображень, навчати на їх основі класифікатор і оцінювати модель.

Colab широко застосовується у сфері машинного навчання для таких завдань:

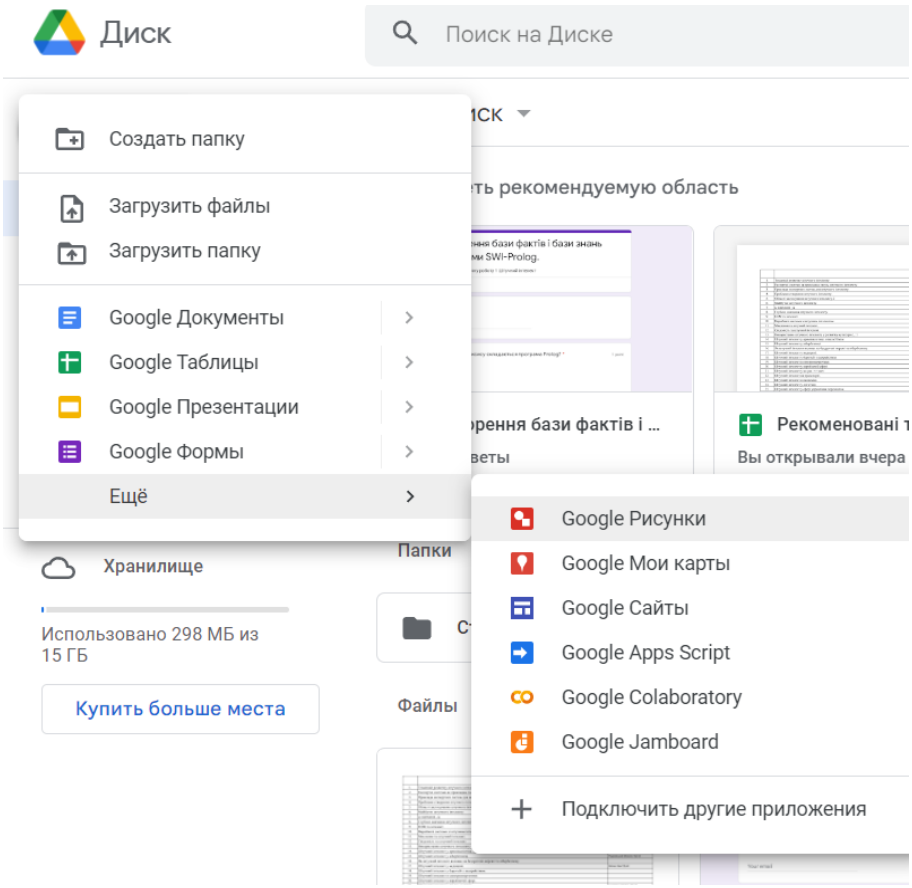
- початок роботи з TensorFlow;
- розробка й навчання нейронних мереж;
- експерименти з тензорними процесорами;
- поширення досліджень у сфері ШІ;
- створення навчальних посібників.

Хід виконання роботи:

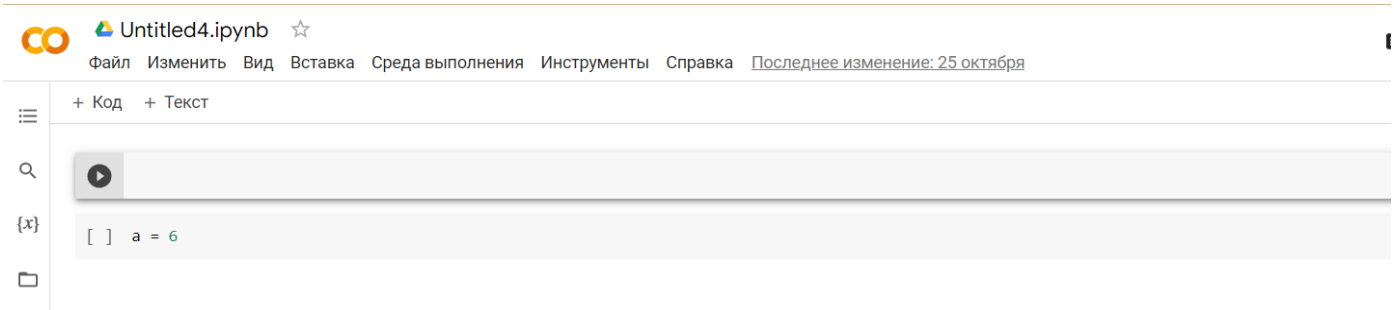
1. Ознайомитись з загальною інформацію про хмарну платформу Google Colaboratory.

2. Виконати всі вправи, які наглядно показано у відео, за наступним посиланням: https://www.youtube.com/watch?v=rt4806DzfUY

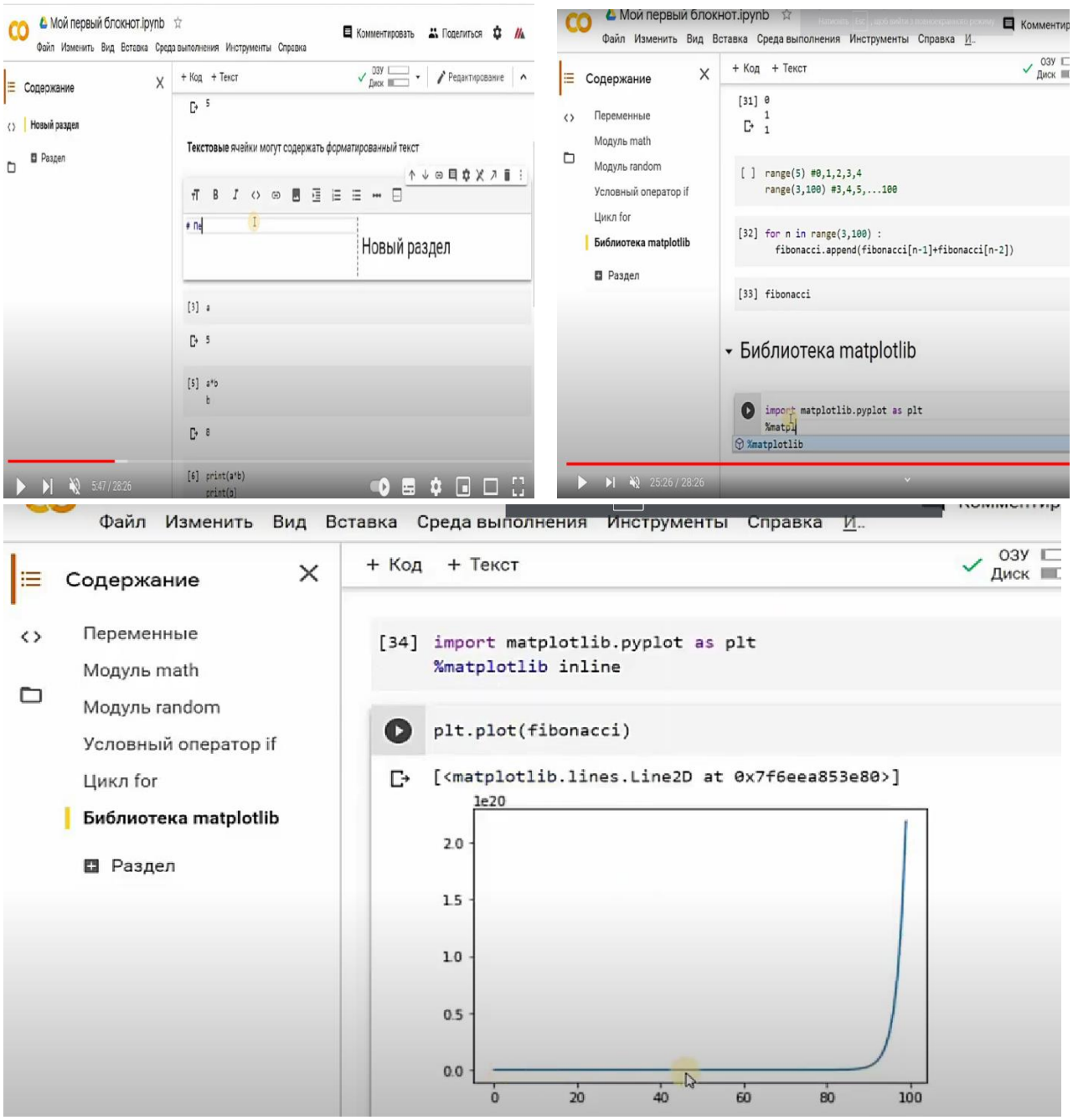
За послідовністю: Гугл Диск – Створити – Ще – Google Colaboratory



Завантажити середовище записника в якому будете виконувати вправи:



3. Зробити скрини екранів виконаних вправ на платформі Google Colaboratory (починаючи від змінних, математичних модулів, операторів, циклів до побудови залежності у вигляді функції)



Типові приклади скринів екрану з виконанням вправ

4. Виконати всі вправи, які наглядно показано у відео, за наступним посиланням: https://www.youtube.com/watch?v=Ve5oW1qqbZg

5. Виконати вправити, які показані в даному відео застосування бібліотеки Keras https://www.youtube.com/watch?v=GquJSJ4KU2E

6. Зробити скрини екранів виконаних вправ на платформі Google Colaboratory (завантаження даних різними способами).

Результати роботи

Завантаження даних в інтернеті::

Дані, які перебувають у відкритому доступі в інтернеті, можна завантажити, використовуючи `wget`:

```
[ ]  !wget http://vis-www.cs.umass.edu/lfw/lfw-a.zip

     --2023-10-27 07:24:44--  http://vis-www.cs.umass.edu/lfw/lfw-a.zip
     Resolving vis-www.cs.umass.edu (vis-www.cs.umass.edu)... 128.119.244.95
     Connecting to vis-www.cs.umass.edu (vis-www.cs.umass.edu)|128.119.244.95|:80... connected.
     HTTP request sent, awaiting response... 200 OK
     Length: 15167852 (14M) [application/zip]
     Saving to: ‘lfw-a.zip.1’

     lfw-a.zip.1         100%[===================>]  14.46M  2.73MB/s    in 5.4s

     2023-10-27 07:24:49 (2.68 MB/s) - ‘lfw-a.zip.1’ saved [15167852/15167852]
```

Завантаження файлів з локального ком’ютера:

Завантажити дані з локального комп'ютера на Colaboratory, або навпаки, можна за допомогою `python`, модуль `google.colab`.

```
[ ]  from google.colab import files
```

Завантаження файлів з локального комп'ютера

```
files.upload()
```

```
Вибрати файли  Файл не вибрано    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Romanok_Viacheslav.jpg to Romanok_Viacheslav (1).jpg
{'Romanok_Viacheslav (1).jpg': b'\xff\xd8\xff\xe0\x00\x10JFIF\x00\x01\x01\x01\x00H\x00H\x00\x00\xff\xe2\x02(ICC_PROFILE\x00\x01\x01\x00\x
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00acsp\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
\x00\x00\x00\x00\x00\x00o\xa2\x00\x008\xf5\x00\x00\x03\x90XYZ \x00\x00\x00\x00\x00\x00b\x99\x00\x00\xb7\x85\x00\x00\x18\xdaXYZ
\x00\x00\x00\x00\x00\x00$\xa0\x00\x00\x0f\x84\x00\x00\xb6\xcfpara\x00\x00\x00\x00\x00\x04\x00\x00\x00\x02ff\x00\x00\xf2\xa7\x00\x00\rY\x0
\x00\x00\x00\x00\x00\x00\xf6\xd6\x00\x01\x00\x00\x00\x00\xd3-mluc\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x0cenUS\x00\x00\x00 \x00\x0
\x002\x000\x001\x006\xff\xdb\x00C\x00\x04\x03\x03\x04\x03\x03\x04\x04\x03\x04\x05\x04\x04\x05\x06\n\x07\x06\x06\x06\x06\r\t\n\x08\n\x0f\r
\x1a\x12BI\x00\x0c\x15]"\x82\tU\xf9\xf4U\x85L\xb0\xc3\xa02\x02\x10\x11!!P\xc9\x05 \x82\x18\x8c\xa4\x8b\n\xd1\x84\x108A\x08\x04"\x84\x84\x
\xa4\xc2+\xa8U\xc1\x00*\xa6\x12\x02E\x90\x92EV`d=\x0el\xd6=\r\x9e{v\xb8\xf5Ul\xd6\x16IRFU\x84@\x90\xc2\x83*#\xac\xb2\xbbAX"\xa0\x90\x01\x
2BH\t$\x04`\x00\xca\x90\x98HE\x03$A\x08\x15\xd6\xc0D!\x06\xb8\x17\xe5lv\xd6\xd9\xdf\x1a\xb8\xd6%\xb2\x02@\xe02\x002\xb4\xb5\x98T\x95h1Z\x
\x15\xd2Y!\x15L\x14\x11@\x11*\xab\xc0+\x00\x02\xc0\x84\x04\x02\x10 H\x88\tP\x88I!\x14\x86\xa5V\x82\xa3W\x14\x19\n\xee\xa3U\xb6C\x15L\x89\
\x0c\x80\xc0y\xa9\xaa\xbc\xf5G\xa5nu\xdb\x89\xb3\xads=\xad[fy\x9b\xa2Tb\xc2\x8c\xb0C(0\xca\xb1\xc1\x04\x01(\xc1\x8a\xca\x18\x00\x83\x16I\
\x90\x82\x03\x00\x18P\x86\x02\x18s\x96\xd9\xc3\xd1\x9a\xad\x82\xcem]Ju\x9co\x17Y\xb5\xa9\x92\xdeiin|\xe5\xadG=\x99\xd5\xb2\xa7\xcdhB\xc0\
D*!\x00\x8c\t\x04\x08\x04\x80\xc02\x96Z\x03k\xf4yCX\xf4\xad\xe7\xfa\x17\x97A\x03\\EaJ\xac\xb0\x15\x95@a*B\x08\n\xd4*FP\x06\x90\xa4\x92\x0
\x8aH\x03\na\x14\x804\x84\x10\xc0I,\x81\xa1\x15\xd4\x02\xc0$d\x08 \x82\x1a\x90\x83\xcc_\x9e]\xf4\xef\xe1[\xcf]\x85\xc5\xaf\x9e\x954Ky\xf5
(d#\x08@\xad\x00A\x14\x182\x90\x18 \t\x83(\x94\xc2+6\xef\xe5I\x9fJ\xdew\xa9\xd3\x8e\xd4\x0fb\xa3\xac(!@*\x15\x80\x840\x01\x80\x8d\x00CPed
\xa6\x8aI!\x15l\x02+\x05\x84\x14\x12\x1a\xf3i\xb6\x8b\xbc\xe2\xd0 dK\xaf\xc5%\xeb\xbf\x12\xeco\xa8\x99\xf5cuS\xacW:\x9e\xa5z\xce\x17\xb2\
\x83,\x89\n\xca"2\x00U4\xee\xe46\xb3\xdf\x9c}\xb7\x96\xb5R\x91\x1d@\tP$\x1a)@D\t\x06\x8c\x85$\x040\x18 \x11L\x95\x04T\x88\xea\xb4\x98c6\x
\x902\x1a\x00\xc4\n\xc0S\x02\xc0H\x90\x9a\xc0\xd9\xdb\x87s\x9e\xf6\xaezt\xe9\xd6y\xe3Zk9\x85\xc9\xa5,\xea\xbao\xe6.]\xb9\xc9\xd1\x9dmIfw\
`\x03\x15\x8a\xe9!\x18@\xab)!\x96\t!\x04\x01V[$\x82\xc3\x16\x1a7\xf2\x0b\x1d\xd9\xc8\xdd\xaezP\xa1h\xa9\x87\n\x11\xda\xb3\x0eU\xf5H\x92I$
\x84\x10\xa9\x0c\x92\xd8T\xa1\x82\x0c\x8e\xb4a\x84V@\xa9R\t\x02"\x8e\x91k\xcb\xdf\x94nt\xb4\xf1,\xe7\xbe\xc3r\xf5\xf3\xe9\xa8\xd4\xd8\xdd
```

Завантажуємо фото в Google Colab:

```
!ls
```

```
 gdrive      lfw-a.zip.1                  Romanok_Viacheslav.jpg
 lfw-a.zip  'Romanok_Viacheslav (1).jpg'  sample_data
```

```
from IPython.display import Image
Image('Romanok_Viacheslav.jpg')
```

Збережемо дані на диску:

```
files.download("sample_data/california_housing_train.csv")
```

Перевіримо чи збереглись дані:

- фото збережено на диску

Підключення Google Drive до віртуальної машини:

Переглядаємо підключені диски:

```
!df -h
```

```
Filesystem      Size  Used Avail Use% Mounted on
overlay         108G   28G   81G  26% /
tmpfs            64M     0   64M   0% /dev
shm             5.8G     0  5.8G   0% /dev/shm
/dev/root       2.0G  1.1G  885M  55% /usr/sbin/docker-init
tmpfs           6.4G  268K  6.4G   1% /var/colab
/dev/sda1        44G   28G   16G  65% /etc/hosts
tmpfs           6.4G     0  6.4G   0% /proc/acpi
tmpfs           6.4G     0  6.4G   0% /proc/scsi
tmpfs           6.4G     0  6.4G   0% /sys/firmware
drive            15G  995M   15G   7% /content/gdrive
```

Переглядаємо вміст диска:

- **Можемо бачити наші файли, які зберігаються на диску**

```
!ls /content/gdrive/
```

```
MyDrive
```

```
!ls /content/gdrive/"My Drive"
```

```
 Certificate
 Classroom
'Colab Notebooks'
 DUT
 Dut2022Before
 images
 T-Smart_ПЗ2_Романок_ШІД_21.gdoc
 T-Smart_ПР1_Романок_ШІД_21.docx
'Звіт - Контрольна робота 1.docx'
'Лабараторна робота №3 - Романок В’ячеслав Володимирович - ШІД-21.gdoc'
'Питання1 щодо ОПЕРАЦІЙНА СИСТЕМА.docx'
 ПР2_СБЖ_Романок_ШІД-31.docx
'ПР9_T-Big_Романок В.В..gdoc'
'Романок В’ячеслав Володимирович (ШІД-21) v2.0.gdoc'
'СОС_Linux_ПР1_Романок В.В..gdoc'
```

Копіювання даних з Google Drive на локальний диск віртуальної машини:

```
!cp /content/gdrive/'My Drive'/datasets/tuberculesis/base_dir.zip .
```

```
cp: cannot stat '/content/gdrive/My Drive/datasets/tuberculesis/base_dir.zip': No such file or directory
```

```
!ls
```

```
 gdrive      lfw-a.zip.1                  Romanok_Viacheslav.jpg
 lfw-a.zip  'Romanok_Viacheslav (1).jpg'  sample_data
```

Копіюємо дані з диска віртуальної машини на Google Drive:

```
!cp 'Romanok_Viacheslav.jpg' /content/gdrive/'My Drive'/images
```

Створюємо Keras Callback для збереження нейронної мережі на Google Drive:

```
from tensorflow.keras.callbacks import ModelCheckpoint
filepath="/content/gdrive/'My Drive'/deep_nets/mynet_best.h5"
checkpoint = ModelCheckpoint(filepath, monitor='val_acc', verbose=1,
                             save_best_only=True, mode='max')
```

Аналіз якості роботи нейронної мережі для розпізнавання моделей одягу в Keras:

```
from tensorflow.keras.datasets import fashion_mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras import utils
from tensorflow.keras.preprocessing import image
from google.colab import files
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
%matplotlib inline
```

Підготовка даних для навчання мережі:

Завантажуємо набір даних:

Keras має вбудовані засоби для роботи з відомими наборами даних, які зазвичай представлені у вигляді пар даних для навчання (x_train, y_train) і даних для перевірки (x_test, y_test).

```
(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
```

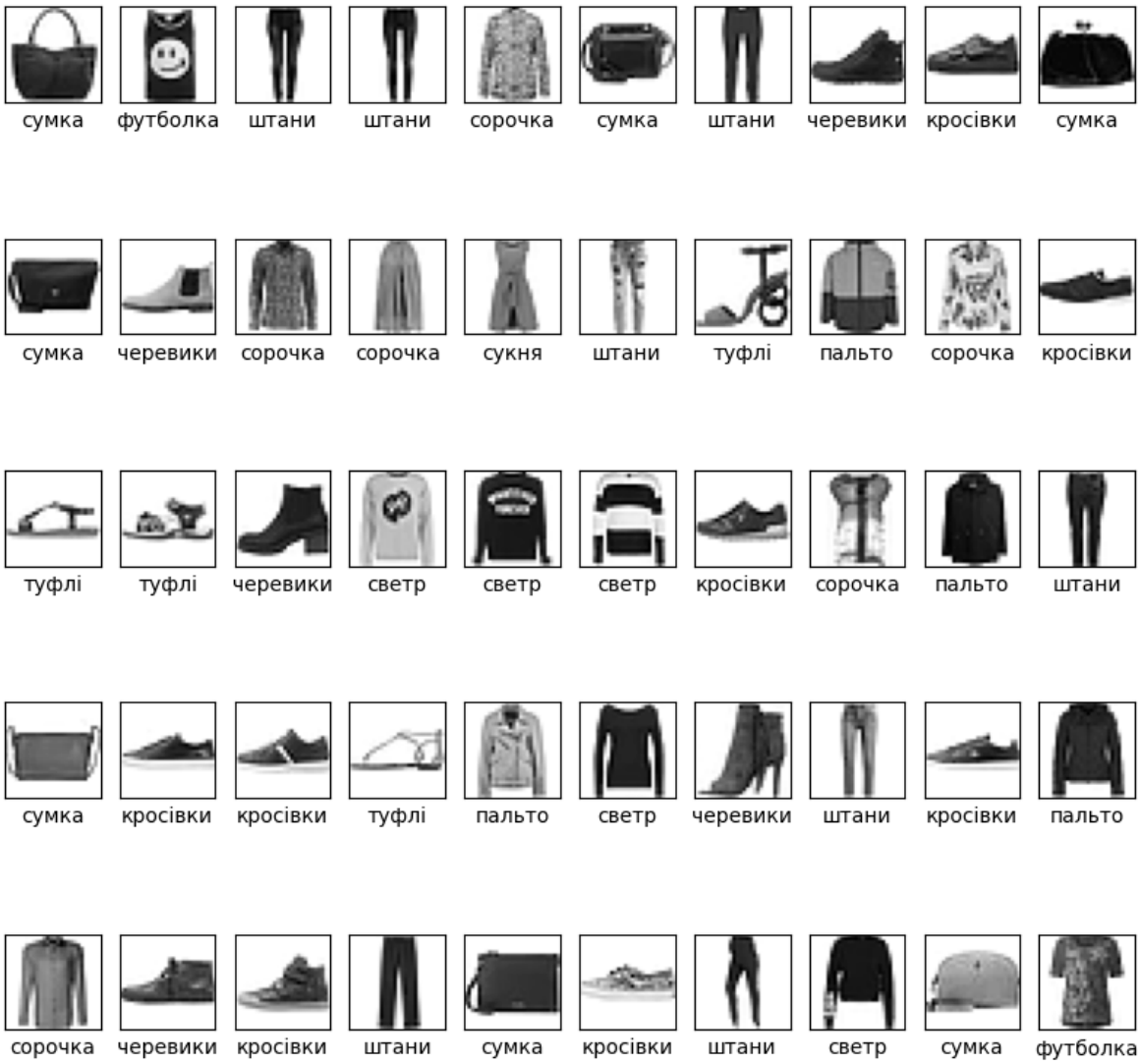
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26421880/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4422102/4422102 [==============================] - 0s 0us/step
```

Список із назвами класів:

```
classes = ['футболка', 'штани', 'светр', 'сукня', 'пальто', 'туфлі', 'сорочка', 'кросівки', 'сумка', 'черевики']
```

Переглянемо приклади зображень:

```
plt.figure(figsize=(10,10))
for i in range(100,150):
    plt.subplot(5,10,i-100+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(x_train[i], cmap=plt.cm.binary)
    plt.xlabel(classes[y_train[i]])
```



Перетворення розмірності даних у наборі:

```
x_train = x_train.reshape(60000, 784)
x_test = x_test.reshape(10000, 784)
```

Нормалізація даних:

```
x_train = x_train / 255
x_test = x_test / 255
```

Робота з правильними відповідями:

```
n = 0

print(y_train[n])
```

```
9
```

Перетворюємо мітки у формат one hot encoding:

```
y_train = utils.to_categorical(y_train, 10)

y_test = utils.to_categorical(y_test, 10)
```

Правильна відповідь у форматі one hot encoding:

```
print(y_train[n])
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
```

Створюємо нейронну мережу:

```
# Створюємо послідовну модель
model = Sequential()
# Вхідний повнозв'язний шар, 800 нейронів, 784 входи в кожен нейрон
model.add(Dense(800, input_dim=784, activation="relu"))
# Вихідний повнозв'язний шар, 10 нейронів (за кількістю рукописних цифр)
model.add(Dense(10, activation="softmax"))
```

Компілюємо мережу:

```
model.compile(loss="categorical_crossentropy", optimizer="SGD", metrics=["accuracy"])
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 800)               628000

 dense_1 (Dense)             (None, 10)                8010

=================================================================
Total params: 636010 (2.43 MB)
Trainable params: 636010 (2.43 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

Навчаємо нейронну мережу:

```
history = model.fit(x_train, y_train,
                    batch_size=200,
                    epochs=100,
                    validation_split=0.2,
                    verbose=1)
```

```
history = model.fit(x_train, y_train,
                    batch_size=200,
                    epochs=100,
                    validation_split=0.2,
                    verbose=1)
```

```
Epoch 1/100
240/240 [==============================] - 3s 10ms/step - loss: 1.1891 - accuracy: 0.6534 - val_loss: 0.8471 - val_accuracy: 0.7383
Epoch 2/100
240/240 [==============================] - 2s 10ms/step - loss: 0.7713 - accuracy: 0.7595 - val_loss: 0.7030 - val_accuracy: 0.7748
Epoch 3/100
240/240 [==============================] - 2s 10ms/step - loss: 0.6732 - accuracy: 0.7878 - val_loss: 0.6378 - val_accuracy: 0.7958
Epoch 4/100
240/240 [==============================] - 3s 12ms/step - loss: 0.6196 - accuracy: 0.8028 - val_loss: 0.5967 - val_accuracy: 0.8076
Epoch 5/100
240/240 [==============================] - 2s 10ms/step - loss: 0.5841 - accuracy: 0.8134 - val_loss: 0.5692 - val_accuracy: 0.8145
Epoch 6/100
240/240 [==============================] - 2s 10ms/step - loss: 0.5584 - accuracy: 0.8189 - val_loss: 0.5486 - val_accuracy: 0.8211
Epoch 7/100
240/240 [==============================] - 2s 10ms/step - loss: 0.5390 - accuracy: 0.8245 - val_loss: 0.5324 - val_accuracy: 0.8242
Epoch 8/100
240/240 [==============================] - 2s 10ms/step - loss: 0.5233 - accuracy: 0.8292 - val_loss: 0.5223 - val_accuracy: 0.8257
```

Зберігаємо нейронну мережу для подальшого використання:

```
model.save('fashion_mnist_dense.h5')
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079:
  saving_api.save_model(
```

Оцінювання якості навчання:

```
scores = model.evaluate(x_test, y_test, verbose=1)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.3753 - accuracy: 0.8696
```

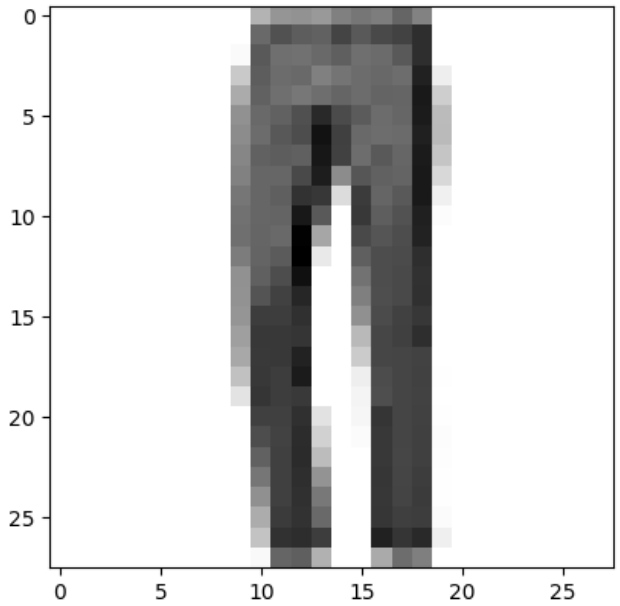
```
print("Частка правильних відповідей на тестових даних, у відсотках:", round(scores[1] * 100, 4))
```

```
Частка правильних відповідей на тестових даних, у відсотках: 86.96
```

Використовуємо мережу для розпізнавання предметів одягу:

```
n_rec = 496
```

```
plt.imshow(x_test[n_rec].reshape(28, 28), cmap=plt.cm.binary)
plt.show()
```



Змінюємо розмірність зображення і нормалізуємо його:

```
x = x_test[n_rec]
x = np.expand_dims(x, axis=0)
```

Запускаємо розпізнавання та друкуємо результати розпізнавання:

```
prediction = model.predict(x)
```

```
1/1 [==============================] - 0s 69ms/step
```

```
prediction
```

```
array([[4.4771281e-05, 9.9984646e-01, 9.2326291e-06, 9.0994079e-05,
        5.1855145e-06, 3.5282011e-09, 3.2761741e-07, 1.8136321e-07,
        2.8723289e-06, 1.0203737e-08]], dtype=float32)
```

Переобразуємо результати з формату one hot encoding:

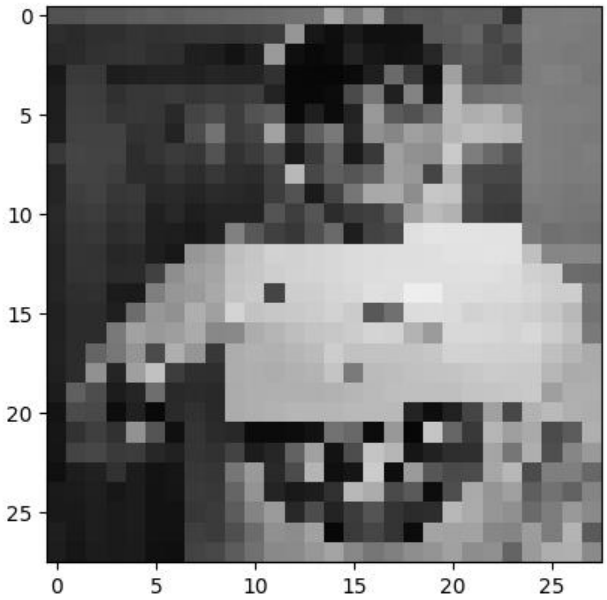
```
prediction = np.argmax(prediction[0])
print("Номер класу:", prediction)
print("Назва класу:", classes[prediction])
```

```
Номер класу: 1
Назва класу: штани
```

Завантажимо своє фото:

```
img_path = 'Romanok_Viacheslav2.jpg'
img = image.load_img(img_path, target_size=(28, 28), color_mode = "grayscale")
```

```
plt.imshow(img.convert('RGBA'))
plt.show()
```



Перетворюємо картинку для обробки нейронною мережею:

```
# Перетворимо картинку в масив
x = image.img_to_array(img)
# Змінюємо форму масиву на плоский вектор
x = x.reshape(1, 784)
# Інвертуємо зображення
x = 255 - x
# Нормалізуємо зображення
x /= 255
```

Запускаємо розпізнавання та друкуємо результати розпізнавання:

```
prediction = model.predict(x)
```

```
1/1 [==============================] - 0s 35ms/step
```

```
prediction
```

```
array([[3.0099806e-01, 2.7128555e-02, 2.4983667e-02, 3.5408430e-04,
        1.2637763e-03, 2.0112070e-11, 1.0805447e-03, 2.3629195e-12,
        6.4419132e-01, 1.0628409e-09]], dtype=float32)
```

```
prediction = np.argmax(prediction)
print("Номер класу:", prediction)
print("Назва класу:", classes[prediction])
```

```
Номер класу: 0
Назва класу: футболка
```

Практична робота №5

Тема. Ознайомлення з платформою по дослідженню даних Kaggle

Мета роботи: Ознайомитись з принципом розпізнавання рукописних цифр за допомогою даних Kaggle в online-середовищі хмарної платформи Google Colaboratory

Загальні відомості

Kaggle – середа організована як публічна веб-платформа, на якій користувачі та організації можуть оприлюднювати набори даних, досліджувати та створювати моделі, взаємодіяти з іншими спеціалістами за даними та інженерами з машинного навчання, організовувати конкурси з дослідження даних та брати участь у них. В системі розміщені набори відкритих даних, надані зовнішні інструменти для оброблення даних і машинного навчання.

В даній практичній роботі обрана задача розпізнавання рукописних цифр із вибору MHIST.

Декілька відомостей про MNIST (Mixed National Institute of Standards and Technology database) є основною базою для тестування системи розпізнавання образів, а також широко використовується для навчання та тестування

алгоритмів машинного навчання. Вона була створена перегрупуванням образів з оригінальної бази NIST, яка є достатньо складною для розпізнавання. Крім цього, були виконані певні перетворення.

База MNIST складається з 60000 образів для навчання та 10000 образів для тестування. Написано велику кількість статей, присвячених задачам розпізнавання MNIST, наприклад (в даному випадку автори використовували ієрархічну систему зі зворотних нейронних сетей).

На Kaggle представлена повна вибірка MNIST, організована трохи по-другому. Тут навчальна вибірка включає в себе 42000 зразків, а вибірка тестування – 28000. Тем не менше, за вмістом вони еквівалентні. Кожен образ MNIST представлений картинкою 28×28 пікселів із 256 градаціями сірого кольору. Приклади кількох неоднозначних в ідентифікації цифр представлені на рисунку нижче.

Рис. 1 Набір цифр для розпізнавання

Відповідно вказівок, які даються у відео, необхідно виконати свій багатошаровий персептрон, навчити його на завантажених і оброблених даних, а потім провести тестування.

Посилання на відео:

https://www.google.com/search?tbm=vid&sxsrf=ALiCzsao27fsejIv9uL4CPd23CIipWuw-Q:1669060931850&q=mnist+%D0%B3%D1%83%D0%B3%D0%BB+%D0%BA%D0%BE%D0%BB%D0%BB%D0%B0%D0%B1&spell=1&sa=X&ved=2ahUKEwj72c2_iMD7AhXuCBAIHdk7CUEQBSgAegQIERAB&biw=1536&bih=746&dpr=1.25#fpstate=ive&vld=cid:b4602fe9,vid:zO0RAtZRkpc.

Хід виконання.

1. Ознайомитись з загальною інформацію та сайтом Kaggle (посилання: https://www.kaggle.com/).

2. Виконати всі вправи, які наглядно показано у відео, за наведеним вище посиланням.

3. Зробити скріни екранів виконаних вправ.

4. Ознайомитись з текстом коду і навести його в практичній роботі

Ошибка! Недопустимый объект гиперссылки..

5. Дати характеристику функцій кожної строки коду.

6. Написати висновок щодо виконання завдань даної практичної роботи

Результати виконання роботи

Змагання з розпізнавання рукописних цифр

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Dropout, Flatten
from tensorflow.keras import utils
from tensorflow.keras.preprocessing import image
from tensorflow.python.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint
import tensorflow as tf
from sklearn.model_selection import train_test_split
from google.colab import files
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-1-a56b12673300> in <cell line: 5>()
      3 from tensorflow.keras import utils
      4 from tensorflow.keras.preprocessing import image
----> 5 from tensorflow.python.keras.preprocessing.image import ImageDataGenerator
      6 from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint
      7 import tensorflow as tf

ModuleNotFoundError: No module named 'tensorflow.python.keras.preprocessing'

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------
```

OPEN EXAMPLES ПОШУК У STACK OVERFLOW

Перевірка формату даних

Данні для навчання

```
[ ] !head train.csv
```

```
label,pixel0,pixel1,pixel2,pixel3,pixel4,pixel5,pixel6,pixel7,pixel8,pixel9,pixel10,pixel11,
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
7,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
```

Данні для тестування

```
[ ] !head test.csv
```

```
pixel0,pixel1,pixel2,pixel3,pixel4,pixel5,pixel6,pixel7,pixel8,pixel9,pixel10,pixel11,pixel1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
```

```
[ ] !head sample_submission.csv
```

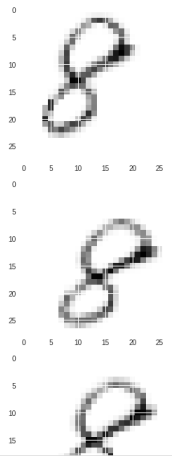
```
ImageId,Label
1,0
2,0
3,0
4,0
5,0
6,0
7,0
8,0
9,0
```

```
train_dataset = np.loadtxt('train.csv', skiprows=1, delimiter=',')
```

```
[ ] train_dataset[0:5]
```

```
array([[1., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [4., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

```
data = X_train[0]
data = np.expand_dims(data, axis=0)
for batch in datagen.flow(data, batch_size=1):
    plt.figure(i)
    imgplot = plt.imshow(batch[0][:,:,0])
    i += 1
    if i % 6 == 0:
        break
plt.show()
```

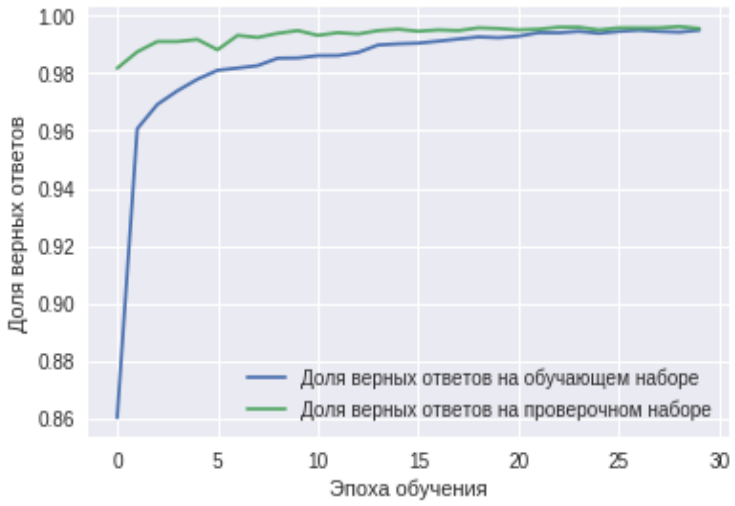


```
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])

print(model.summary())
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 32)        832
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 28, 28, 32)        25632
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 14, 64)        18496
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 14, 14, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 3136)              0
_________________________________________________________________
dense (Dense)                (None, 256)               803072
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2570
=================================================================
Total params: 887,530
Trainable params: 887,530
Non-trainable params: 0
_________________________________________________________________
None
```

```
plt.plot(history.history['acc'],
         label='Доля верных ответов на обучающем наборе')
plt.plot(history.history['val_acc'],
         label='Доля верных ответов на проверочном наборе')
plt.xlabel('Эпоха обучения')
plt.ylabel('Доля верных ответов')
plt.legend()
plt.show()
```




```
[ ] predictions = model.predict(x_test)
```

```
[ ] predictions[:5]
```

```
array([[7.0808330e-19, 1.2293642e-15, 1.0000000e+00, 1.6526332e-13,
        4.8959526e-16, 1.6048716e-18, 1.2838233e-17, 1.1961627e-12,
        7.7564989e-14, 2.3213994e-16],
       [9.9999893e-01, 1.6688695e-12, 6.3178089e-09, 6.3089062e-10,
        7.1604861e-10, 2.8566465e-09, 2.1731910e-07, 2.6650182e-10,
        3.7282078e-07, 4.8097058e-07],
       [2.9786121e-10, 2.6178534e-10, 1.4524261e-07, 1.2859771e-09,
        5.9936187e-06, 1.6657364e-10, 6.1487855e-12, 1.1224843e-08,
        2.0155933e-06, 9.9999177e-01],
       [9.9914634e-01, 1.0904064e-07, 7.0121109e-06, 1.3721834e-05,
        4.1037286e-08, 5.3695007e-06, 3.6620080e-05, 4.0169263e-07,
        2.4500233e-04, 5.4535794e-04],
       [2.5390169e-13, 2.4379212e-11, 6.3516737e-08, 9.9996173e-01,
        3.9227069e-17, 2.6025369e-09, 1.9312415e-11, 3.6186339e-11,
        3.8091617e-05, 5.9398592e-10]], dtype=float32)
```

Преобразуем результаты распознавания из формата one hot encoding в цифры

```
[ ] predictions = np.argmax(predictions, axis=1)
```

```
[ ] predictions[:5]
```

```
array([2, 0, 9, 0, 3])
```

Перелік посилань

1. https://www.victoria.lviv.ua/library/students/sss2021/theme1.html
2. https://www.victoria.lviv.ua/library/students/nn/help/nn-sharky.pdf
3. https://medium.com/@meqdad.dev/building-simple-and-customizable-image-classifier-with-teachable-machine-and-python-30d50169d638#:~:text=Teachable%20Machine%20is%20a%20web,images%2C%20sounds%2C%20and%20poses.
4. https://www.youtube.com/watch?v=rt4806DzfUY
5. https://www.youtube.com/watch?v=GquJSJ4KU2E
6. https://www.youtube.com/watch?v=Ve5oW1qqbZg
7. https://www.google.com/search?tbm=vid&sxsrf=ALiCzsao27fsejIv9uL4CPd23CIipWuw-Q:1669060931850&q=mnist+%D0%B3%D1%83%D0%B3%D0%BB+%D0%BA%D0%BE%D0%BB%D0%BB%D0%B0%D0%B1&spell=1&sa=X&ved=2ahUKEwj72c2_iMD7AhXuCBAIHdk7CUEQBSgAegQIERAB&biw=1536&bih=746&dpr=1.25#fpstate=ive&vld=cid:b4602fe9,vid:zO0RAtZRkpc.

Надруковано в РВЦ
Державного університету інформаційно-комунікаційних технологій
Формат 60х90/16. Папір друкарський.
Наклад 100 прим. Зам. 530.

Свідоцтво суб’єкта видавничої справи ДК №7917 від 16.08.2023 р.

03110, м. Київ, вул. Солом’янська, 7.
Тел. (044) 249-25-76