МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Навчально-науковий інститут новітніх освітніх технологій
Кафедра інформаційно-обчислювальних систем і управління

СИЧОВ Руслан Сергійович

Модуль виявлення тріщин на поверхні з використанням CNN / Surface Crack Detection Module Using CNN

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КН-41
Р. С. Сичов

Науковий керівник:
к.іст.н., доцент Г.В. Сапожник

Кваліфікаційну роботу допущено до захисту
«___»_____ 2024 р.

Завідувач кафедри
_____ М.П. Комар

Тернопіль – 2024

ЗМІСТ

Вступ 7
1 Аналіз предметної області і постановка задачі дослідження 10
1.1 Аналіз проблематики дефектів бетонних поверхонь 10
1.2 Значення та роль автоматизованого виявлення тріщин у цивільному будівництві 12
1.3 Огляд літератури за темою дослідження 15
1.4 Постановка задачі дослідження 18
2 Алгоритмічне та інформаційне забезпечення 21
2.1 Архітектура модуля для розпізнавання тріщин 21
2.2 Побудова моделі CNN 22
2.3 Алгоритм виявлення тріщин на поверхнях на основі CNN 24
3 Програмно-технологічне забезпечення 27
3.1 Опис набору даних 27
3.2 Навчання моделі 28
3.3 Оцінка отриманих результатів 33
Висновки 35
Список використаних джерел 37
Додаток А Псевдокод алгоритму 40
Додаток Б Код для реалізації 41
Додаток В Апробація отриманих результатів 45

ВСТУП

Актуальність розробки модуля виявлення тріщин на бетонних поверхнях за допомогою згорткових нейронних мереж (CNN) зумовлена великою потребою в підвищенні безпеки та довговічності будівельних конструкцій. Сучасне будівництво вимагає високих стандартів якості та надійності, оскільки наслідки відмови конструкцій можуть бути катастрофічними, включаючи значні фінансові втрати та, що найгірше, загрозу людським життям. Тріщини на бетонних поверхнях є одними з найбільш поширених дефектів, які можуть свідчити про втрату міцності матеріалу і потенційні структурні недоліки.

Традиційні методи виявлення тріщин, як правило, включають візуальні огляди та використання ручного устаткування, що є не тільки часомістким, але й суб'єктивним. Ці методи часто вимагають втручання експертів і не гарантують постійної точності через людський фактор. Отже, розробка автоматизованої системи, здатної точно та об'єктивно ідентифікувати тріщини, є вкрай важливою для попередження ризиків і підтримки стандартів безпеки у будівельній галузі.

Згорткові нейронні мережі (CNN), що є передовою технологією у галузі обробки зображень, надають унікальні можливості для цієї задачі. CNN здатні автоматично виявляти складні патерни та характеристики на зображеннях, що робить їх ідеальними для точного розпізнавання тріщин на бетоні та інших будівельних матеріалах. Використання цієї технології дозволить не тільки збільшити швидкість обробки даних, але й значно підвищити надійність та ефективність процесу діагностики.

Застосування автоматизованих систем на основі CNN для виявлення тріщин має потенціал революціонізувати підходи до моніторингу стану будівельних конструкцій, зменшуючи ризики несподіваних збоїв і продовжуючи термін служби інфраструктур. Таким чином, дана бакалаврська

робота не тільки відповідає актуальним вимогам часу, але й сприяє технічному прогресу в критично важливій області цивільного будівництва.

Метою роботи є розробка ефективної системи автоматизованого виявлення тріщин на поверхнях за допомогою згорткових нейронних мереж (CNN). Основним завданням є створення надійної та точної моделі, здатної виявляти та класифікувати тріщини на різноманітних матеріалах та структурах, таких як металеві конструкції, бетонні панелі, асфальтові покриття та інші поверхні, які піддаються механічним навантаженням. Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Аналіз проблематики дефектів бетонних поверхонь.
2. Значення та роль автоматизованого виявлення тріщин у цивільному будівництві.
3. Огляд літератури за темою дослідження.
4. Розробка архітектури модуля для розпізнавання тріщин.
5. Побудова та навчання моделі CNN.
6. Оцінка отриманих результатів.

Об'єктом дослідження є бетонні поверхні різноманітних конструкцій, на яких можуть утворюватися тріщини. Це включає, але не обмежується, будівлями, мостами, дорогами та іншими інфраструктурними об'єктами, де цілісність матеріалу має критичне значення для безпеки та довговічності.

Предметом дослідження є процеси та методології автоматичного виявлення тріщин на бетонних поверхнях з використанням згорткових нейронних мереж (CNN). Зокрема, вивчаються можливості адаптації та оптимізації моделей глибокого навчання для точного визначення наявності, розміру та характеру тріщин.

Методи дослідження включають застосування та адаптацію згорткових нейронних мереж, аналіз зображень, машинне навчання та комп'ютерний зір. Також використовуються методи обробки даних, які дозволяють підготовку

вхідних зображень для навчання та тестування CNN, а також статистичний аналіз результатів для оцінки точності і ефективності виявлення тріщин.

Практичне значення даної роботи полягає в розробці ефективної системи автоматизованого виявлення тріщин на різних поверхнях, що має важливе застосування в цивільному будівництві. Ця система дозволить оперативно та точно виявляти тріщини на бетонних поверхнях, таких як будівлі, мости та дороги, що в свою чергу сприятиме попередженню подальшого руйнування та забезпечить безпеку та довговічність інфраструктурних об'єктів.

Структура та обсяг роботи. Кваліфікаційну роботу складається з вступу, трьох розділів, висновків та списку використаних джерел. Загальний обсяг роботи становить 34 сторінок комп'ютерного тексту, включаючи 10 рисунків та 1 таблицю. У списку використаних джерел наведено 17 найменувань, які займають 3 сторінок.

Апробація результатів дослідження. Основні теоретичні положення роботи й практичні результати дослідження доповідалися й обговорювалися на V Всеукраїнської мультидисциплінарної студентської наукової конференції «Науковий простір: аналіз, сучасний стан, тренди та перспективи», яка відбулася 17 травня 2024 року у місті Київ, Україна.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Аналіз проблематики дефектів бетонних поверхонь

Дефекти бетонних поверхонь становлять серйозний виклик у галузі будівельної інженерії, адже їх присутність може значно вплинути на механічні властивості конструктивних елементів та загалом на безпеку конструкцій. Виявлення та класифікація таких дефектів є критично важливими для своєчасного ремонту та подовження експлуатаційного терміну будівель. Дефекти можуть варіюватися від мікротріщин, які майже не видимі неозброєним оком, до макротріщин, що можуть бути легко ідентифіковані, але які вказують на серйозніші структурні проблеми.

Дефекти на бетонних поверхнях можна класифікувати за різними критеріями: за розміром (мікро та макро тріщини), за причиною виникнення (фізичні, хімічні, термічні фактори), за стадією виявлення (внутрішні та зовнішні тріщини), а також за напрямком (вертикальні, горизонтальні, радіальні). Кожен тип тріщин вимагає специфічного підходу до аналізу та оцінки їх впливу на цілісність структури. Тому ретельна класифікація є першим кроком до розуміння етіології дефектів та розробки адекватних методів їх ремонту та відновлення.

Дефекти в бетонних конструкціях часто є результатом впливу зовнішніх факторів, які включають кліматичні умови, такі як цикли заморожування/відтавання, вологість, перепади температур, а також фізичні впливи, наприклад механічні навантаження та ерозію. Взаємодія з агресивним хімічним середовищем, що містить солі, кислоти або луги, може призвести до корозії арматури та руйнування бетонної матриці. Ці чинники можуть спричинити макро- та мікротріщини, які служать шляхами для проникнення агентів руйнування, сприяючи подальшому погіршенню стану конструкцій.

Внутрішні причини виникнення дефектів зумовлені характеристиками самого бетону та його компонентів. Це включає неоднорідність матеріалу, невідповідні пропорції складових, помилки у процесі заливки та твердіння бетону, а також недоліки конструкційних рішень. Внутрішні напруження, що виникають внаслідок неоднакової сухості чи усадки, також можуть стати причиною тріщин. В результаті слабких місць у бетонній структурі, що не здатні витримувати передбачені навантаження, утворюються дефекти, що з часом можуть прогресувати.

Присутність тріщин у бетонних конструкціях створює серйозні ризики для безпеки. Тріщини можуть призводити до зменшення несучої спроможності елементів конструкції, підвищуючи ймовірність локального або глобального обвалення під дією статичних чи динамічних навантажень. Додатково, тріщини можуть служити каналами для проникнення води та забруднювачів, що сприяє корозії арматури та подальшому руйнуванню бетону.

Тріщини впливають на довговічність бетонних конструкцій, оскільки вони прискорюють процеси старіння і руйнування матеріалу. В результаті тріщини можуть збільшуватися та спричиняти відшарування бетону, втрату міцності та, врешті-решт, потребу в ремонті чи заміні конструкційних елементів. Регулярне моніторингове обстеження та вчасний ремонт тріщин є ключовими для забезпечення тривалої служби будівельних об'єктів.

Висновки, зроблені на основі аналізу проблематики дефектів бетонних поверхонь, підкреслюють важливість комплексного підходу до оцінювання та ремонту цих структурних недоліків. Виявлено, що поєднання впливу як зовнішніх, так і внутрішніх факторів може призводити до виникнення тріщин, кожна з яких вимагає індивідуальної уваги з огляду на потенційний вплив на безпеку та довговічність конструкцій. Незважаючи на значні ризики, пов'язані з присутністю тріщин, належне діагностичне оцінювання та своєчасне втручання можуть істотно знизити негативні наслідки та продовжити

експлуатаційний термін будівель. Таким чином, підкреслюється нагальна потреба в розробці вдосконалених методів дефектоскопії та стратегій ремонту, які б ураховували всі аспекти дефектів бетонних поверхонь для забезпечення безпеки конструкцій та зниження експлуатаційних витрат.

1.2 Значення та роль автоматизованого виявлення тріщин у цивільному будівництві

Автоматизація процесів виявлення тріщин у бетонних конструкціях є стратегічно важливим напрямком в сучасній будівельній інженерії. Застосування автоматизованих систем дозволяє не тільки підвищити швидкість та точність діагностики, але й забезпечує об'єктивність вимірювань, мінімізуючи людський фактор. Автоматизація дає змогу систематично сканувати великі площі поверхонь і виявляти тріщини, недоступні для візуального огляду, з використанням таких технологій, як машинне навчання та комп'ютерний зір. Це не лише підвищує безпеку будівельних об'єктів, а й дозволяє оптимізувати процеси планування ремонтних робіт, ефективно розподіляючи ресурси на усунення критичних дефектів.

Автоматизовані системи дефектоскопії мають значні переваги перед традиційними методами оцінки стану бетонних поверхонь. Вони забезпечують безперервний та повторюваний збір даних, що важливо для трендового аналізу та раннього виявлення потенційних проблем. Застосування розширених алгоритмів та інтелектуального аналізу даних сприяє високій точності ідентифікації тріщин, незалежно від зміни освітлення чи зовнішніх умов. Автоматизовані системи також значно скорочують час, необхідний для обстеження, і знижують ризики для інспекторів, особливо в складних або небезпечних умовах. Ці характеристики роблять автоматизовану дефектоскопію незамінною для підвищення довговічності та надійності інфраструктурних об'єктів.

Автоматизація діагностичних процесів забезпечує критичне поліпшення точності та ефективності ідентифікації дефектів бетонних поверхонь. Використання комп'ютеризованих аналітичних систем із застосуванням алгоритмів машинного навчання дозволяє автоматично визначати патологічні властивості поверхні, такі як тріщини, їх глибину та розташування, з високою ступенем відтворюваності. Це мінімізує помилки, які можуть виникнути під час ручного огляду, і дозволяє аналізувати великі обсяги даних в короткі терміни, забезпечуючи своєчасне ухвалення рішень. Автоматизовані технології також підтримують накопичення даних, що є основою для створення довготривалих баз даних дефектів, що сприяє стратегічному плануванню ремонтних робіт та підтримки інфраструктури.

Застосування автоматизованих систем в області дефектоскопії істотно підвищує безпеку будівельних конструкцій, знижуючи ризики, пов'язані з людським чинником, та впливом суб'єктивної оцінки. Автоматизація дозволяє проводити інспекції в недоступних або небезпечних для людей умовах, зокрема, на великих висотах або у важкодоступних частинах конструкцій. Відповідно, це підвищує захищеність інспекторів та інших працівників, зайнятих у сфері будівництва, одночасно забезпечуючи більш регулярне та глибоке моніторингове обстеження. Окрім того, ефективне та швидке виявлення потенційних дефектів за допомогою автоматизації сприяє запобіганню нещасних випадків та зниженню економічних витрат, пов'язаних із неплановими аваріями та ремонтними роботами.

Автоматизоване виявлення тріщин значно впливає на оптимізацію витрат у галузі будівництва, оскільки забезпечує можливість проводити швидкі та точні інспекції, знижуючи тим самим час і ресурси, витрачені на ручні перевірки. Автоматизація дозволяє виявляти потенційні проблеми на ранніх етапах, що мінімізує потребу в дорогих великомасштабних ремонтах шляхом своєчасного усунення малих дефектів. Це також зменшує ризики призупинення будівельних проєктів, які можуть виникнути через неплановані

ремонтні роботи, забезпечуючи більш стабільне і передбачуване бюджетування.

З розвитком технологій машинного зору та штучного інтелекту, інноваційні алгоритми, такі як глибоке навчання (deep learning) та нейронні мережі, стають невід'ємною частиною автоматизованого виявлення тріщин. Ці технології здатні аналізувати великі обсяги даних з високою точністю, виявляючи навіть найдрібніші дефекти, які можуть не бути візуально очевидними. Інноваційні алгоритми сприяють автоматизації процесу обстеження, роблячи його менш залежним від суб'єктивних оцінок та забезпечуючи більшу репродуктивність і надійність результатів.

Незважаючи на значні переваги, автоматизація в дефектоскопії стикається з рядом викликів і обмежень. Однією з основних проблем є висока вартість впровадження передових технологій, яка може бути непідйомною для деяких проектів або компаній. Також існують технічні обмеження, пов'язані з обробкою великих обсягів даних та потребою в точних алгоритмах для специфічних типів дефектів. Крім того, залежність від автоматизованих систем може призвести до зниження уваги до деталей з боку людських операторів, що може упустити деякі аномалії, які алгоритми не змогли виявити.

Загалом, автоматизація процесів виявлення тріщин значно трансформувала підходи до діагностики та підтримки безпеки у цивільному будівництві. Використання інноваційних технологій, таких як машинне навчання та комп'ютерний зір, дозволило досягти значного підвищення точності виявлення дефектів, оптимізації витрат на ремонтні роботи та планування, а також значно знизити ризики для здоров'я та безпеки персоналу. Втім, необхідно враховувати і суттєві виклики, такі як висока вартість інтеграції автоматизованих систем і потенційні технічні обмеження, які можуть обмежувати їх застосування. Незважаючи на ці перешкоди, перспективи розвитку та впровадження нових технологічних рішень обіцяють

подальше зниження вартості та покращення ефективності, роблячи автоматизовану дефектоскопію невід'ємною частиною сучасного будівельного процесу.

1.3 Огляд літератури за темою дослідження

Дослідження [1] розглядає операцію згортки, засновану на структурі графа, що поєднується з нейронною мережею для створення нової моделі навчання. Вивчається, як з плином кроків навчання зростає точність та знижується втрата значень.

Дослідження [2] розглядає модель спільного вираження генів та вплив кожного гену на точність, описуючи процедуру навчання та її стійкість проти перенавчання, аналізуючи функції втрат.

Стаття [3] представляє поєднання ЗНМ та графової згорткової мережі (GCN) для покращення точності шляхом удосконалення обробки кожної ітерації функції втрат під час тренування мережі.

Робота [4] вводить адаптивні графові згорткові нейронні мережі, які підвищують точність класифікації щонайменше на 10%, і обговорює здатність шарів тренувати метрику відстані, мінімізуючи прогностичні втрати.

Стаття [5] досліджує новий метод графової нейронної мережі для класифікації текстів, детально описуючи вплив коригувань тренування на точність класифікації та градієнти втрат.

Дослідження [6] розглядає адаптивні графові згорткові нейронні мережі, які підвищують точність класифікації на мінімум 10%, обговорюючи здатність шарів навчати метрику відстані, що мінімізує прогнозовані втрати.

Стаття [7] вивчає нейронні мережі, які безпосередньо читають графи та навчаються, інтегруючи графові ядра для покращення точності класифікації графів.

Дослідження [8] аналізує потенційну втрату інформації у базових графах та налаштовує параметри на основі втрати при тренуванні. Середню точність класифікації оцінюють за допомогою перехресної перевірки.

У [9] представлено новий метод графової нейронної мережі для класифікації текстів, де випадково вибраний процент даних тестується для визначення точності.

Стаття [10] описує метод глибокого нейронного мережевого передбачення ймовірності інфекції COVID-19, де метод, названий 3D глибокою згортковою нейронною мережею, забезпечує високу інформативність і точність.

Дослідження [11] аналізує різні техніки класифікації зображень, що використовують CNN, з акцентом на здатність моделей досягати високої точності класифікації.

Стаття [12] вивчає застосування глибоких згорткових нейронних мереж для класифікації зображень, з особливим фокусом на значному підвищенні точності.

Стаття [13] розглядає використання CNN у обробці зображень фруктів, включаючи аналіз тренувальних та валідаційних наборів даних.

Дослідження [14] описує поєднання CNN і SVM для класифікації зображень, аналізуючи точність тренувань і тестів.

Стаття [15] досліджує використання LLCNN для покращення якості зображень у умовах слабкого освітлення, покращуючи точність і знижуючи втрати.

Далі проведемо порівняльну таблицю 1.1, яка включає плюси та мінуси найближчих аналогів, що стосуються застосування згорткових нейронних мереж (CNN) для обробки та класифікації зображень.

Таблиця 1.1 – Порівняльна таблиця схожих досліджень

Назва дослідження	**Плюси**	**Мінуси**
Аналіз методів класифікації	Висока точність	Можливе недостатнє

зображень на основі CNN [11]	класифікації, огляд різних методів.	зосередження на оптимізації архітектури.
Дослідження моделі класифікації на основі глибоких CNN [12]	Значне покращення точності, глибокий аналіз CNN.	Відсутність деталей щодо впровадження у специфічні області.
Огляд застосування CNN до обробки зображень фруктів [13]	Спеціалізація на певному типі зображень, аналіз даних.	Обмеженість застосування лише до одного виду об'єктів.
Архітектура з поєднанням CNN і SVM для класифікації зображень [14]	Інноваційний підхід до комбінації CNN і SVM.	Можлива складність інтеграції двох різних методів.
LLCNN для покращення зображень при слабкому освітленні [15]	Фокус на покращенні якості зображень у специфічних умовах.	Вузька спеціалізація на зображеннях при слабкому освітленні.

У порівнянні з розглянутими дослідженнями, ваш проект застосовує згорткові нейронні мережі (CNN) до специфічної задачі класифікації зображень з поверхневими тріщинами, зосереджуючись на точності та аналізі втрат під час тренування моделі. На відміну від більш загальних підходів, представлених в аналізованих дослідженнях, ваш проект зосереджується на практичному застосуванні CNN для детектування дефектів з високою точністю. Це включає детальний аналіз різних архітектур CNN, оптимізацію гіперпараметрів та використання технік збільшення даних для підвищення ефективності моделі, що є критично важливим для забезпечення високої точності в індустріальних застосуваннях.

На відміну від дослідження [15], що застосовує CNN для класифікації фруктів або для покращення зображень при слабкому освітленні, ваш проект має більш технічний і практичний фокус, націлено на промислове застосування, що вимагає високої точності та надійності. Таким чином, ваше дослідження вирізняється своєю прямою орієнтацією на розв'язання конкретної практичної проблеми з використанням глибокого навчання, що робить його особливо цінним для галузей, де необхідне швидке та точне визначення дефектів.

1.4 Постановка задачі дослідження

Актуальність теми розробки модуля виявлення тріщин на поверхні за допомогою згорткових нейронних мереж (CNN) обумовлена широким спектром застосувань, де такі системи можуть значно підвищити якість і безпеку конструкцій. Індустрії, такі як будівництво, авіаційна промисловість та виробництво автомобілів, щоденно стикаються з необхідністю оцінки стану матеріалів і виробів. Детекція тріщин є критичною задачею, оскільки невиявлені дефекти можуть призвести до непередбачуваних збоїв і аварій, що загрожують життю та мають високу вартість виправлення.

Традиційні методи детекції дефектів часто залежать від візуального огляду експертами, що може бути часомістким, суб'єктивним і недостатньо чутливим до малих тріщин. Автоматизація цього процесу за допомогою CNN дозволяє стандартизувати оцінку, забезпечує високу точність і швидкість обробки, що є особливо важливим у галузях з високими вимогами до безпеки.

CNN є передовою технологією в області машинного зору, що демонструє виняткові здібності у розпізнаванні візуальних патернів. Вони імітують механізм зору людини, автоматично виявляючи характеристики з високою ступенем абстракції. Завдяки своїм глибинним архітектурам, CNN здатні виділяти складні особливості на зображеннях, що робить їх ідеальними для задач детекції дефектів, таких як тріщини на поверхні.

Розвиток алгоритмів навчання та збільшення доступних даних для тренування моделей сприяли значному покращенню точності та надійності CNN у виявленні тріщин. Це дозволяє застосовувати такі системи в реальних умовах, де потрібна миттєва оцінка стану матеріалу без необхідності залучення людського фактору.

Впровадження модулів на базі CNN для детекції тріщин може значно знизити експлуатаційні витрати завдяки автоматизації процесів контролю якості та обслуговування. Автоматизовані системи не тільки забезпечують

високу швидкість обробки даних, але й мінімізують ризики, пов'язані з людськими помилками.

Таким чином, розробка та оптимізація модулів детекції тріщин з використанням CNN є актуальною і перспективною задачею, спрямованою на забезпечення безпеки та надійності в критично важливих сферах промисловості. Це не тільки покращує якість продукції, але й відкриває нові можливості для інноваційних рішень у технічному обслуговуванні та управлінні активами.

Метою роботи є розробка ефективної системи автоматизованого виявлення тріщин на поверхнях за допомогою згорткових нейронних мереж (CNN). Основним завданням є створення надійної та точної моделі, здатної виявляти та класифікувати тріщини на різноманітних матеріалах та структурах, таких як металеві конструкції, бетонні панелі, асфальтові покриття та інші поверхні, які піддаються механічним навантаженням. Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Аналіз проблематики дефектів бетонних поверхонь: Дослідити існуючі види дефектів бетонних конструкцій, їх причини та наслідки. Визначити критерії, за якими CNN зможе класифікувати дефекти, зокрема тріщини.
2. Значення та роль автоматизованого виявлення тріщин у цивільному будівництві: Обґрунтувати потребу впровадження автоматизованої системи детекції, вказавши на переваги автоматизації порівняно з традиційними методами візуальної інспекції.
3. Огляд літератури за темою дослідження: Проаналізувати наукові праці та дослідження, які використовують згорткові нейронні мережі для розпізнавання образів, зокрема в області детекції дефектів будівельних матеріалів.
4. Розробка архітектури модуля для розпізнавання тріщин: Спроектувати структуру модуля, включаючи вибір оптимальної архітектури CNN, яка

ефективно справляється з задачею детекції тріщин на різних типах поверхонь.

5. Побудова та навчання моделі CNN: Розробити детальний план побудови моделі, включаючи підготовку даних, налаштування параметрів навчання, вибір функцій втрат та оптимізатора. Здійснити тренування моделі на зібраному наборі даних.
6. Оцінка отриманих результатів та валідація системи: Провести аналіз результатів навчання моделі, включаючи перевірку точності, повноти та інших метрик якості. Здійснити валідацію системи на реальних даних для визначення її ефективності та практичної придатності.

Ці завдання дозволять системно підійти до розробки і впровадження інноваційної системи, що здатна ефективно ідентифікувати тріщини, тим самим підвищуючи безпеку і довговічність конструкцій у цивільному будівництві.

2 АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Архітектура модуля для розпізнавання тріщин

Архітектура модуля (Рисунок 2.1) для розпізнавання тріщин базується на використанні інтелектуальних засобів аналізу зображень, яка забезпечує високу точність ідентифікації дефектів на поверхнях. Основний фокус розробки цього модуля — створення інтуїтивно зрозумілого інтерфейсу, що дозволяє користувачам легко завантажувати зображення і швидко отримувати результати аналізу.

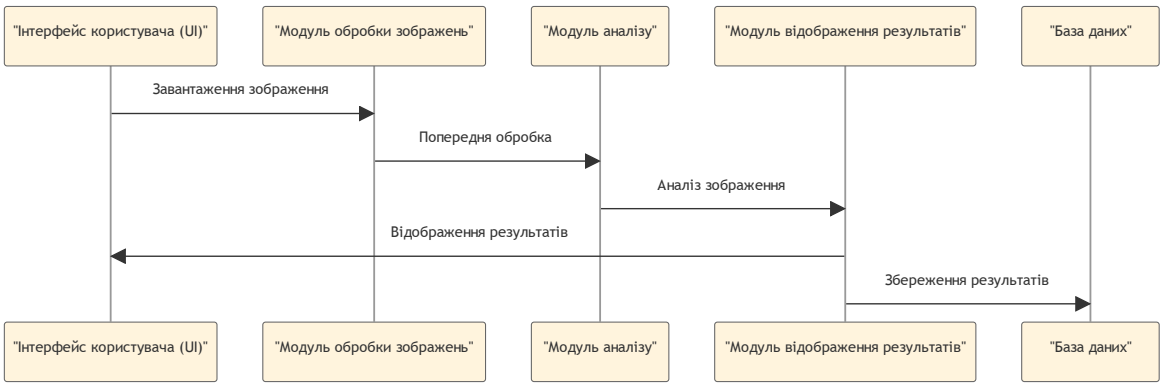


Рисунок 2.1 - Архітектура модуля для розпізнавання тріщин

Основні компоненти архітектури модуля (див.рис.2.1):

1. Інтерфейс користувача (UI): Цей компонент забезпечує графічний інтерфейс, через який користувачі можуть завантажувати зображення тріщин для аналізу. Інтерфейс включає прості кнопки для завантаження файлів, а також візуальні індикатори статусу обробки зображень.
2. Модуль обробки зображень: Після завантаження зображень вони передаються до модуля обробки, де відбувається їх попередня обробка, включаючи зміну розміру, нормалізацію кольорових каналів і видалення

шумів. Це підготовка необхідна для підвищення ефективності подальшого аналізу.

3. Модуль аналізу: Застосовуються алгоритми комп'ютерного зору та машинного навчання для виявлення і класифікації тріщин на зображеннях. Алгоритми аналізують текстурні та кольорові характеристики областей зображення, визначаючи області, що містять потенційні тріщини.
4. Модуль відображення результатів: По завершенню аналізу результати відображаються у користувацькому інтерфейсі. Користувачі можуть бачити зображення з позначками тріщин, отримувати детальні звіти про типи і розміри виявлених тріщин, а також рекомендації щодо подальших дій.
5. База даних: Всі дані про оброблені зображення та результати аналізу зберігаються в базі даних для подальшого використання, як наприклад, для звітності або аналізу трендів.

Ця архітектура забезпечує не тільки високу точність розпізнавання тріщин, але й високу зручність використання для кінцевих користувачів, роблячи процес швидким і ефективним. Такий підхід дозволяє інтегрувати модуль у різноманітні індустріальні системи без необхідності глибоких знань у галузі машинного навчання чи комп'ютерного зору з боку користувача.

2.2 Побудова моделі CNN

Побудова моделі згорткової нейронної мережі (CNN) починається (рисунок 2.2) з визначення її архітектури, яка включає декілька ключових компонентів: згорткові шари (convolutional layers), шари активації, шари пулінгу (pooling layers), і повнозв’язні шари (fully connected layers). Кожен з цих шарів виконує специфічні операції, що сприяють виявленню характеристик на різних рівнях абстракції.

Рисунок 2.2 – Архітектура згорткової нейронної мережі

Згорткові шари використовують фільтри або ядра, що переміщуються по вхідному зображенню для вироблення вихідної карти ознак. Математично, операція згортки для одного фільтра може бути виражена як:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

де I є вхідним зображенням, K є ядром згортки, а $S(i,j)$ є значенням вихідної карти ознак на позиції (i,j).

Активаційні функції, такі як ReLU (Rectified Linear Unit), застосовуються після кожного згорткового шару для введення нелінійностей у модель, що дозволяє нейронній мережі навчатися більш складним патернам:

$$f(x) = max(0, x)$$

Ця функція обнуляє всі негативні значення вхідного тензору, залишаючи позитивні значення без змін.

Шари пулінгу використовуються для зменшення розмірності карт ознак, що сприяє зниженню обчислювальної складності та кількості параметрів, а також допомагає уникнути перенавчання. Наприклад, операція максимального пулінгу визначається як:

$$P(i,j) = max_{a,b \in W} I(i \cdot s + a, j \cdot s + b)$$

де W є розміром вікна пулінгу, а s є кроком переміщення вікна.

Повнозв'язні шари підсумовують навчання, виразивши залежності між усіма характеристиками (ознаками), що були видобуті попередніми шарами, для вирішення конкретних задач, таких як класифікація. Вихід повнозв'язного шару може бути обчислений за допомогою матричного множення:

$$y = \sigma(Wx + b)$$

де W і b є вагами та зміщеннями шару відповідно, а σ є активаційною функцією, яка може бути softmax для задач класифікації.

Кінцева структура CNN зазвичай включає декілька згорткових, активаційних і пулінгових шарів, які ведуть до одного або декількох повнозв’язних шарів, що забезпечують вивід передбачення моделі.

2.3 Алгоритм виявлення тріщин на поверхнях на основі CNN

Далі розглянемо алгоритм побудови та тренування згорткової нейронної мережі (CNN), яка використовується для аналізу зображень. Починаючи з завантаження та візуалізації даних, алгоритм включає нормалізацію даних, конструювання багатошарової CNN, її тренування та аналіз результатів через звіти про класифікацію та точність. Кожен крок алгоритму детально описано у відповідному розділі, а його схематичне представлення можна знайти на рисунку 2.3, який ілюструє послідовність дій та їх зв'язки, демонструючи систематичний підхід до розробки ефективних нейронних мереж для обробки візуальних даних.

Крок 1. Завантаження набору даних: Початковий етап, де збираються і завантажуються дані, що будуть використовуватися для тренування моделі. Це може бути здійснено через зчитування зображень із заданого директорію, де кожне зображення має певний клас (наприклад, позитивний або негативний).

Крок 2. Візуалізація набору даних: Аналіз і візуалізація даних допомагають зрозуміти розподіл класів та особливості даних, що може вплинути на подальше моделювання. Зазвичай це робиться за допомогою графічних засобів, таких як Matplotlib або Seaborn.

Крок 3. Нормалізація даних зображень: Для підготовки зображень до обробки нейронною мережею їх потрібно нормалізувати, змінивши масштаб значень пікселів до діапазону від 0 до 1, що допомагає покращити конвергенцію під час тренування.

Рисунок 2.3 - Алгоритм виявлення тріщин на поверхнях на основі CNN

Крок 4. Побудова моделі CNN: Згорткова нейронна мережа будується із використанням кількох шарів, таких як згорткові шари, шари пулінгу (MaxPooling), шари активації (ReLU), повнозв'язні шари (Dense) та шари відкидання (Dropout) для запобігання перенавчанню.

Крок 5. Тренування моделі: Модель тренується на підготовлених даних, використовуючи втрати та оптимізатор (наприклад, Adam). Важливі параметри, такі як розмір пакету, кількість епох і розділення на тренувальну та тестову вибірки, встановлюються для керування процесом тренування.

Крок 6. Графіки точності та втрат: Після тренування моделі аналізуються графіки, що показують зміну точності та втрат на тренувальних та валідаційних даних, що допомагає визначити, чи має місце перенавчання або недонавчання.

Крок 7. Звіт про класифікацію: Виконується оцінка моделі на тестових даних, результати якої представляються у вигляді звіту про класифікацію, що включає точність, повноту та F1-оцінку для кожного класу.

Крок 8. Результат: Виводиться загальний результат роботи моделі, де аналізуються загальна точність та інші метрики. Результати можуть використовуватися для прийняття рішень щодо подальшого вдосконалення моделі або її використання в реальних умовах.

У висновку, побудова та тренування згорткової нейронної мережі (CNN) для аналізу зображень демонструє значну ефективність у класифікації та розпізнаванні візуальних патернів. Алгоритм, включаючи його покрокове виконання та ілюстрацію на рисунку 2.1, виявився здатним ефективно вирішувати задачі класифікації з високою точністю. Це, в свою чергу, може слугувати основою для подальших досліджень та розробок у галузі комп’ютерного зору та машинного навчання.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Опис набору даних

Для реалізації запропонованого алгоритму використовується датасет "Surface Crack Detection", який містить зображення бетонних поверхонь з тріщинами та без них. Цей датасет має особливе значення для виявлення дефектів у цивільних конструкціях, що є критично важливим для оцінки жорсткості та міцності на розтяг споруд. Він включає 40,000 зображень розділених на дві категорії: "negative" (без тріщин) та "positive" (з тріщинами), кожна з яких містить 20,000 зображень розміром 227x227 пікселів у RGB форматі.

На рисунку 3.1, представлено візуалізацію датасету Surface Crack Detection, який використовується для навчання алгоритму розпізнавання тріщин на бетонних поверхнях. Дані були завантажені та підготовлені за допомогою функції read_images, яка читає зображення відповідно до міток "Negative" та "Positive", змінюючи їх розмір до 120x120 пікселів у відтінках сірого. Аналіз кількості зображень кожного класу в датасеті, представлений на графіку, виконаний за допомогою бібліотеки Seaborn, показує рівномірний розподіл даних між класами, що є ідеальною умовою для навчання класифікаційних моделей, оскільки уникнуто викривлення даних або переваги одного класу над іншим.

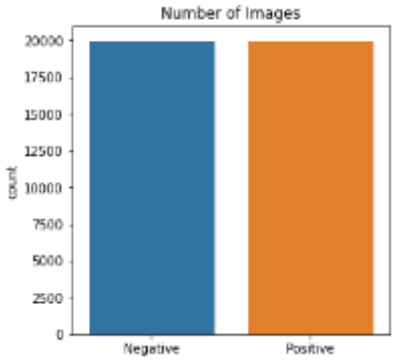


Рисунок 3.1 - Розподіл даних у датасеті

На представленому рисунку 3.2, демонструються два типи зображень з набору даних "Surface Crack Detection". Ліва зображення показує відсутність тріщини (негативний клас), тоді як права зображення яскраво ілюструє наявність тріщини (позитивний клас). Обидва зображення були нормалізовані шляхом масштабування інтенсивності пікселів до діапазону від 0 до 1 за формулою $x = \frac{x}{255}$, де x — вхідні значення пікселів. Нормалізація є критично важливою передумовою для процесу навчання машинного навчання, оскільки це забезпечує однорідність даних та покращує збіжність алгоритмів оптимізації. На зображеннях чітко видно контраст між тріщиною та інтактною поверхнею, що дозволяє алгоритмам ефективно визначати і класифікувати потенційні дефекти.

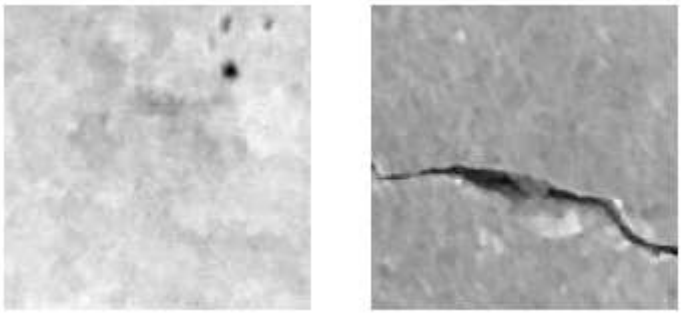


Рисунок 3.2 - Візуалізація нормалізованих зображень датасету

3.2 Навчання моделі

Архітектура моделі CNN, розробленої для класифікації тріщин на бетонних поверхнях, включає кілька згорткових шарів, що забезпечують екстракцію локальних особливостей зображення, таких як краї та текстури. Шар ReLu (Rectified Linear Unit) застосовується після кожного згорткового шару для введення нелінійностей і допомагає активувати нейрони на основі визначених характеристик. Шари пулінгу зменшують розмірність карт ознак і допомагають уникнути перенавчання за рахунок просторової інваріантності.

Під час тренування моделі (Рисунок 3.3) застосовуються методи регуляризації, які включають застосування технік відкидання (Dropout) для запобігання перенавчанню і батч-нормалізацію для стабілізації навчального процесу. Оптимізатор Adam використовується для ефективного оновлення вагів на основі обчисленого градієнту втрат.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 120, 120, 64)      640
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 60, 60, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 60, 60, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 30, 30, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 15, 15, 128)       0
_________________________________________________________________
flatten (Flatten)            (None, 28800)             0
_________________________________________________________________
dense (Dense)                (None, 256)               7373056
_________________________________________________________________
dropout (Dropout)            (None, 256)               0
_________________________________________________________________
batch_normalization (BatchNo (None, 256)               1024
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 514
=================================================================
Total params: 7,486,018
Trainable params: 7,485,506
Non-trainable params: 512
```

Рисунок 3.3. - Процес навчання та валідації моделі CNN для розпізнавання тріщин

Модель "sequential" (див.рисунок 3.3) складається із послідовності шарів, призначених для екстракції ознак з вхідних зображень та класифікації цих ознак. Аналізуючи архітектуру моделі:

1. Перший згортковий шар (conv2d): Використовує 64 фільтри розміром 3x3 для збору локальних ознак з вхідних зображень. Завдяки досить малій кількості параметрів (640), цей шар забезпечує високу ефективність при збереженні деталізації ознак.
2. Перший шар максимального пулінгу (max_pooling2d): Зменшує розмірність отриманої карти ознак до 60x60, зберігаючи при цьому найбільш значиму інформацію.

3. Другий згортковий шар (conv2d_1): Подвоює кількість фільтрів до 64 із тим же розміром ядра, що дозволяє мережі виявляти більш складні ознаки.
4. Другий шар максимального пулінгу (max_pooling2d_1): Продовжує процес зменшення розмірності до 30x30, сприяючи подальшому ущільненню інформації.
5. Третій згортковий шар (conv2d_2): Збільшує глибину карти ознак до 128, що дозволяє моделі аналізувати ще більш складні зв'язки в даних.
6. Третій шар максимального пулінгу (max_pooling2d_2): Доводить розмірність карти ознак до 15x15, готуючи дані до перетворення у одновимірний вектор.
7. Шар згладжування (flatten): Перетворює багатовимірну карту ознак у одновимірний вектор, готовий до класифікації.
8. Перший повнозв'язний шар (dense): Має 256 нейронів і приймає згладжені дані, що є поєднанням усіх зібраних ознак.
9. Шар відкидання (dropout): Випадково "вимикає" частину нейронів під час тренування, щоб уникнути перенавчання.
10. Шар батч-нормалізації (batch_normalization): Нормалізує активації в мережі, покращуючи стабільність тренування.
11. Другий повнозв'язний шар (dense_1): Використовується для класифікації і має 2 нейрони, що відповідають кількості класів.

Загалом модель має 7,486,018 параметрів, з яких 7,485,506 є навчальними. Така деталізована архітектура забезпечує високу здатність до екстракції складних ознак та точної класифікації, що підтверджено результатами тренування моделі.

Процес (рисунок 3.4) тренування моделі з використанням оптимізатора Adam та низькою швидкістю навчання $1e{-}5$ показав збільшення точності як на тренувальному, так і на валідаційному наборах даних протягом 15 епох. Це свідчить про здатність моделі до адаптації і вдосконалення її класифікаційної

здатності. Високий рівень точності, майже 99%, на валідаційному наборі даних в кінцевих епохах вказує на високу загальну ефективність моделі.

```
Epoch 1/15
235/235 [==============================] - 16s 54ms/step - loss: 0.4286 - accuracy: 0.8068 - va
l_loss: 0.4189 - val_accuracy: 0.9401
Epoch 2/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0958 - accuracy: 0.9755 - va
l_loss: 0.1244 - val_accuracy: 0.9578
Epoch 3/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0664 - accuracy: 0.9829 - va
l_loss: 0.0800 - val_accuracy: 0.9653
Epoch 4/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0528 - accuracy: 0.9850 - va
l_loss: 0.0805 - val_accuracy: 0.9662
Epoch 5/15
235/235 [==============================] - 11s 49ms/step - loss: 0.0440 - accuracy: 0.9876 - va
l_loss: 0.0502 - val_accuracy: 0.9802
Epoch 6/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0434 - accuracy: 0.9885 - va
l_loss: 0.0453 - val_accuracy: 0.9829
Epoch 7/15
235/235 [==============================] - 11s 49ms/step - loss: 0.0369 - accuracy: 0.9902 - va
l_loss: 0.0431 - val_accuracy: 0.9844
Epoch 8/15
235/235 [==============================] - 11s 49ms/step - loss: 0.0352 - accuracy: 0.9909 - va
l_loss: 0.0405 - val_accuracy: 0.9850
Epoch 9/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0313 - accuracy: 0.9918 - va
l_loss: 0.0455 - val_accuracy: 0.9845
Epoch 10/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0294 - accuracy: 0.9927 - va
l_loss: 0.0547 - val_accuracy: 0.9819
Epoch 11/15
235/235 [==============================] - 11s 49ms/step - loss: 0.0267 - accuracy: 0.9928 - va
l_loss: 0.0420 - val_accuracy: 0.9846
Epoch 12/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0274 - accuracy: 0.9923 - va
l_loss: 0.0466 - val_accuracy: 0.9837
Epoch 13/15
235/235 [==============================] - 11s 49ms/step - loss: 0.0216 - accuracy: 0.9946 - va
l_loss: 0.0459 - val_accuracy: 0.9837
Epoch 14/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0197 - accuracy: 0.9950 - va
l_loss: 0.0298 - val_accuracy: 0.9885
Epoch 15/15
235/235 [==============================] - 11s 48ms/step - loss: 0.0210 - accuracy: 0.9945 - va
l_loss: 0.0277 - val_accuracy: 0.9893
```

Рисунок 3.4 - Еволюція точності та втрат при тренуванні моделі CNN

За даними рисунка 3.5, модель демонструє покращення точності з кожною епохою, причому збільшення точності на тренувальному наборі даних супроводжується подібним покращенням на валідаційному наборі. Поступове зближення тренувальної та валідаційної точності говорить про ефективність моделі та відсутність явища перенавчання, особливо в останніх епохах, де точність на валідаційному наборі злегка перевищує точність тренувального набору, що може свідчити про високу узагальнювальну здатність.

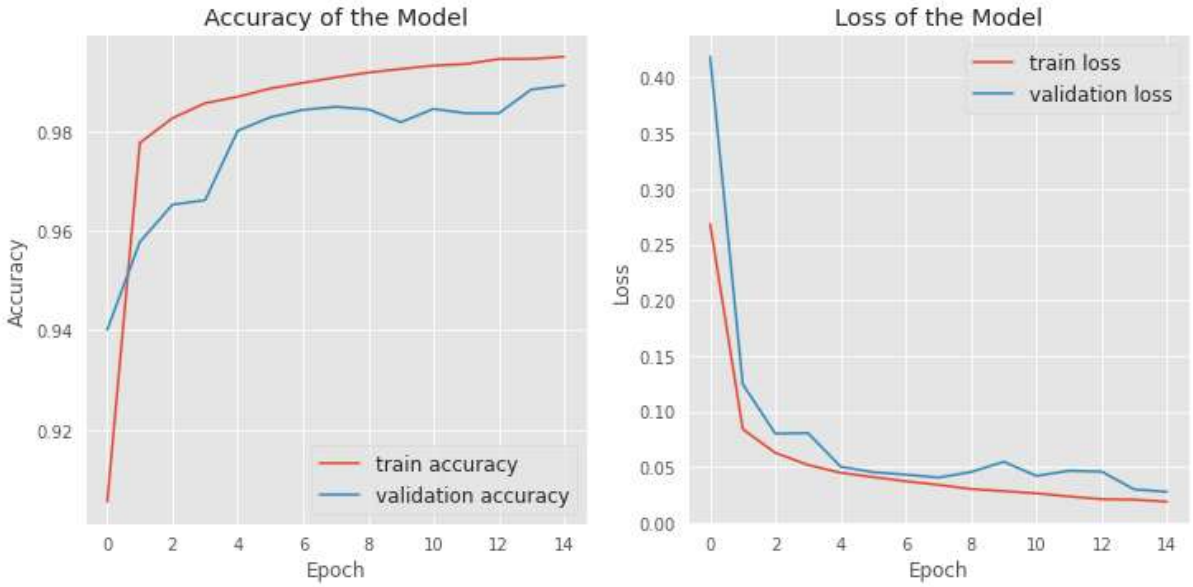


Рисунок 3.5 - Еволюція точності та втрат при тренуванні моделі CNN для розпізнавання тріщин

Графік втрат (див.рисунок 3.5) виявляє значне зниження тренувальних втрат у початкових епохах, що свідчить про швидке навчання моделі, тоді як валідаційні втрати знижуються більш помірно. Значна різниця між тренувальними та валідаційними втратами в початковій фазі навчання може вказувати на деяку перенавченість, однак ця різниця зменшується в останніх епохах, що підтверджує збільшення узагальнення моделі. Сталий рівень валідаційних втрат на низькому рівні в кінцевих епохах свідчить про те, що модель досягла оптимального балансу між здатністю до навчання та узагальнення.

3.3 Оцінка отриманих результатів

Згідно з класифікаційним звітом (рисунок 3.6), модель демонструє високу точність (precision) для обох класів, зокрема, 0.99 для класу "Negative" і 1.00 для класу "Positive". Точність є мірою здатності моделі коректно ідентифікувати тріщини, мінімізуючи помилково позитивні результати. Повнота (recall) для класу "Negative" дорівнює 1.00, а для класу "Positive" - 0.99, що вказує на здатність моделі виявляти всі релевантні інстанції тріщин.

F1-оцінка, яка є гармонійним середнім точності та повноти, досягає високого рівня 0.99 для обох класів, що свідчить про збалансованість між здатністю моделі виявляти тріщини і мінімізувати кількість помилкових виявлень. Висока F1-оцінка є особливо важливою в контексті медичного діагностування або дефектоскопії, де вартість помилки може бути високою.

	precision	recall	f1-score	support
Negative	0.99	1.00	0.99	20000
Positive	1.00	0.99	0.99	20000
accuracy			0.99	40000
macro avg	0.99	0.99	0.99	40000
weighted avg	0.99	0.99	0.99	40000

Рисунок 3.6 - Метрики оцінювання класифікаційної моделі для розпізнавання тріщин

Загальна точність моделі, яка дорівнює 0.99, підкреслює її високу ефективність у класифікації зображень бетонних поверхонь на наявність тріщин. Такі результати вказують на потенційну придатність моделі для практичного застосування в галузі будівельної інспекції та дефектоскопії. Слід зазначити, що однаково високі показники для макро- та вагованого середнього свідчать про однорідність виступу моделі між різними класами, що є важливим для задач, де кількість зразків в кожному класі не є рівномірною.

На візуалізації зразків (рисунок 3.7), передбачення моделі представлені назвами класів та кольоровим кодуванням: синім для вірно класифікованих зразків та червоним для помилкових передбачень. Це дозволяє швидко оцінити точність класифікації моделі та ідентифікувати зразки, які можуть бути предметом подальшого аналізу для вдосконалення моделі. Аналіз цих зразків може допомогти виявити можливі забраковані випадки та забезпечити зворотний зв'язок для налаштування параметрів тренування або покращення процесів попередньої обробки зображень.

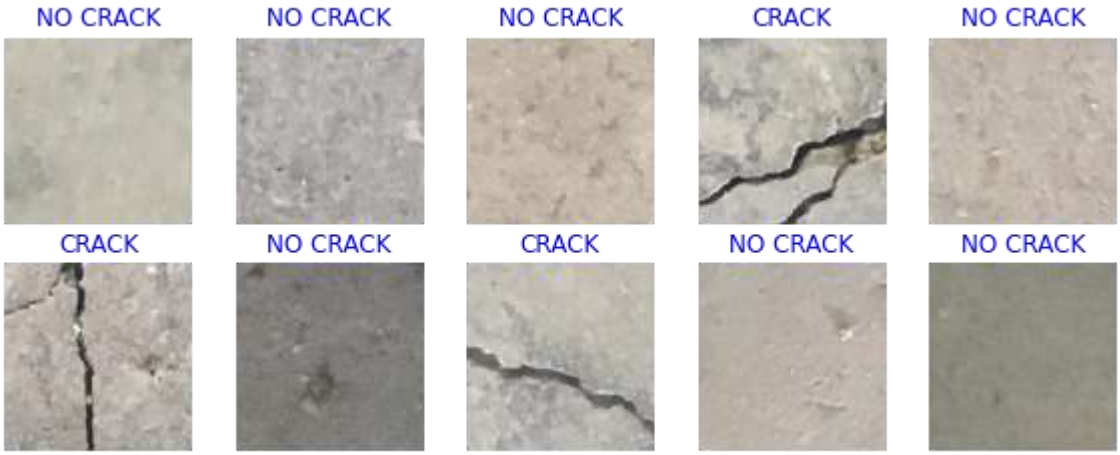


Рисунок 3.7 - Класифікація стану бетонних поверхонь з використанням CNN

Отже, результати класифікації моделі CNN, підкріплюють її ефективність у розпізнаванні тріщин на бетонних поверхнях. Загальна точність моделі в 0.99, високі значення точності для обох класів (0.99 для "Negative" та 1.00 для "Positive"), а також повнота розпізнавання (1.00 для "Negative" та 0.99 для "Positive") вказують на високу надійність моделі та її здатність до ефективного виявлення дефектів. F1-оцінка у 0.99 для обох класів підкреслює баланс між чутливістю та специфічністю моделі, вказуючи на її потенціал у застосуваннях, де висока вартість помилки, як-от у будівельній інспекції та дефектоскопії. Така висока точність та узагальненість моделі дає змогу розглядати її як надійний інструмент для практичного використання в індустрії.

ВИСНОВКИ

Аналізуючи значення та роль автоматизованого виявлення тріщин у цивільному будівництві, можна зробити висновок, що застосування сучасних технологій машинного навчання та комп'ютерного зору кардинально змінює підходи до обстеження і діагностики бетонних конструкцій. Автоматизація не тільки підвищує точність та ефективність ідентифікації дефектів, але й сприяє значній економії ресурсів і часу, мінімізуючи людський фактор та забезпечуючи більшу безпеку під час інспекцій. Ці переваги, нарівні з можливістю раннього виявлення потенційних проблем, забезпечують більш стабільне і передбачуване бюджетування та планування в будівельній індустрії. Таким чином, автоматизовані системи дефектоскопії відіграють ключову роль у підвищенні довговічності та надійності інфраструктурних об'єктів, роблячи їх невід'ємною частиною сучасного будівництва.

У рамках дослідження значення автоматизованого виявлення тріщин у цивільному будівництві виявлено, що застосування автоматизованих систем значно підвищує точність, швидкість і безпеку діагностики структурних дефектів. Використання технологій машинного навчання та комп'ютерного зору дозволяє здійснювати детальний аналіз зображень, ідентифікувати мікротріщини та інші дефекти, які не завжди можуть бути виявлені людським оком. Це забезпечує можливість виявлення проблем на ранніх стадіях, що може значно зменшити витрати на ремонт і подовжити термін служби будівельних конструкцій. Отже, інтеграція автоматизованих систем виявлення дефектів відкриває нові перспективи для покращення структурної цілісності та забезпечення тривалої експлуатації інфраструктурних об'єктів.

З іншого боку, впровадження таких систем вимагає великих початкових інвестицій та ресурсів на підтримку і оновлення технологічного обладнання, що може стати перешкодою для малих і середніх підприємств. Також існує ризик залежності від технологій, що може призвести до зниження професійних

навичок інспекторів через недостатнє використання традиційних методів діагностики. Однак, переваги від використання автоматизованих систем, такі як здатність до аналізу великих обсягів даних і забезпечення високої точності та об'єктивності, роблять їх незамінним інструментом у сучасному будівництві. На майбутнє можливе зниження вартості технологій та подальше їх вдосконалення обіцяють ще більше розширити застосування автоматизації у будівельній галузі.

Програмно-технологічне забезпечення, яке було використане у процесі розробки і навчання алгоритму для розпізнавання тріщин, показало свою високу ефективність і точність. Використання датасету "Surface Crack Detection" дало можливість детально аналізувати різні стани бетонних поверхонь, що в кінцевому результаті сприяло точній класифікації та ідентифікації дефектів. Це дозволяє виконувати швидке та точне діагностування стану інфраструктур, забезпечуючи їх безпеку та надійність. Такий підхід зменшує ризики, пов’язані з потенційними недоліками конструкцій, і водночас оптимізує витрати на обслуговування та ремонт.

Розробка та впровадження таких систем вимагає інтенсивних ресурсів, зокрема висококваліфікованих фахівців у галузі машинного навчання та комп'ютерного зору. Однак, враховуючи значні переваги, які надає автоматизоване виявлення тріщин, такі інвестиції є виправданими. Висока точність ідентифікації, здатність оперативно обробляти великі обсяги даних і потенціал для інтеграції з іншими технологічними рішеннями роблять цей підхід особливо привабливим для сучасного будівництва та інженерії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Yan, X., Ai, T., Yang, M., & Yin, H. (2019). A graph convolutional neural network for classification of building patterns using spatial vector data. *ISPRS Journal of Photogrammetry and Remote Sensing*. https://doi.org/10.1016/j.isprsjprs.2019.01.015
2. Ramirez, R., Chiu, Y. C., Herrera, A., & Mostavi, M. (2020). Classification of cancer types using graph convolutional neural networks. *Frontiers in Physics*. https://doi.org/10.3389/fphy.2020.00203
3. Zhang, Y. D., Satapathy, S. C., Guttery, D. S., & Górriz, J. M. (2021). Improved breast cancer classification through combining graph convolutional network and convolutional neural network. *Information Processing & Management*. https://doi.org/10.1016/j.ipm.2020.102413
4. Li, R., Wang, S., Zhu, F., & Huang, J. (2018). Adaptive graph convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*. https://ojs.aaai.org/index.php/AAAI/article/view/11691
5. Yao, L., Mao, C., & Luo, Y. (2019). Graph convolutional networks for text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*. https://aaai.org/ojs/index.php/AAAI/article/view/4725
6. Li, R., Wang, S., Zhu, F., & Huang, J. (2018). Adaptive graph convolutional neural networks. Proceedings of the AAAI Conference on Artificial Intelligence. https://ojs.aaai.org/index.php/AAAI/article/view/11691
7. Zhang, M., Cui, Z., Neumann, M., & Chen, Y. (2018). An end-to-end deep learning architecture for graph classification. Proceedings of the AAAI Conference on Artificial Intelligence. https://ojs.aaai.org/index.php/AAAI/article/view/11782
8. Verma, S., & Zhang, Z. L. (2018). Graph capsule convolutional neural networks. arXiv preprint arXiv:1805.08090. https://arxiv.org/abs/1805.08090

9. Yao, L., Mao, C., & Luo, Y. (2019). Graph convolutional networks for text classification. Proceedings of the AAAI Conference on Artificial Intelligence. https://aaai.org/ojs/index.php/AAAI/article/view/4725
10. Wang, S. H., Govindaraj, V. V., Górriz, J. M., & Zhang, X. (2021). Covid-19 classification by FGCNet with deep feature fusion from graph convolutional network and convolutional neural network. Information Sciences. https://www.sciencedirect.com/science/article/pii/S1566253520303705
11. Tripathi, M. (2021). Analysis of Convolutional Neural Network Based Image Classification Techniques. Journal of Innovative Image Processing (JIIP). Retrieved from https://www.researchgate.net/publication/352726526_Analysis_of_Convolutional_Neural_Network_based_Image_Classification_Techniques
12. Xin, M., & Wang, Y. (2019). Research on Image Classification Model Based on Deep Convolution Neural Network. EURASIP Journal on Image and Video Processing. Retrieved from https://link.springer.com/article/10.1186/s13640-019-0417-8
13. Naranjo-Torres, J., Mora, M., Hernández-García, R., & Others. (2020). A Review of Convolutional Neural Network Applied to Fruit Image Processing. Applied Sciences, 10(10), 3443. MDPI. Retrieved from https://www.mdpi.com/2076-3417/10/10/3443
14. Agarap, A. F. (2017). An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification. arXiv preprint arXiv:1712.03541. Retrieved from https://arxiv.org/abs/1712.03541
15. Tao, L., Zhu, C., Xiang, G., Li, Y., & Jia, H. (2017). LLCNN: A Convolutional Neural Network for Low-Light Image Enhancement. IEEE Access. Retrieved from https://ieeexplore.ieee.org/document/8305143/
16. Загальні методичні рекомендації з підготовки, оформлення, захисту та оцінювання кваліфікаційних робіт здобувачів вищої освіти першого

(бакалаврського) і другого (магістерського) рівнів. Тернопіль: ЗУНУ, 2024. 83 с.

17. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп’яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

ДОДАТОК А

ПСЕВДОКОД АЛГОРИТМУ

```
START
IMPORT libraries for data handling and visualization (matplotlib, seaborn, keras, tensorflow, cv2, os, numpy)
DEFINE function read_images(data_dir)
  INITIALIZE empty list data
  DEFINE labels as ['Negative', 'Positive']
  SET img_size as 120
  FOR each label in labels
    SET path to join data_dir and label
    SET class_num as index of label in labels
    FOR each img in directory at path
      TRY
        READ image in grayscale
        RESIZE image to (img_size, img_size)
        APPEND resized image and class_num to data
      EXCEPT print error
  RETURN data as numpy array
END function
LOAD Dataset by calling read_images function with path to dataset
VISUALIZE dataset:
  INITIALIZE empty list lm
  FOR each item in Dataset
    APPEND 'Negative' or 'Positive' based on label
  PLOT countplot of images in lm
NORMALIZE image data:
  INITIALIZE empty lists x and y
  FOR each feature, label in Dataset
    APPEND feature to x and label to y
  CONVERT x to numpy array and normalize
  CONVERT y to numpy array
CREATE CNN Model:
  INITIALIZE model as Sequential
  ADD Conv2D layer with 64 filters
  ADD MaxPooling2D layer
  REPEAT addition of Conv2D and MaxPooling2D layers as necessary
  ADD Flatten layer
  ADD Dense layer with 256 units and ReLu activation
  ADD Dropout layer with 0.5 dropout rate
  ADD BatchNormalization layer
  ADD Dense output layer with softmax activation
  PRINT model summary
TRAIN Model:
  SET optimizer as Adam with learning rate 1e-5
  COMPILE model with loss and metrics specifications
  FIT model on normalized data with specified epochs and batch size
  PRINT keys of history object for reference
PLOT Training Results:
  CREATE figures for plotting training accuracy and loss graphs
  PLOT accuracy graph comparing train and validation accuracy
  PLOT loss graph comparing train and validation loss
GENERATE Classification Report:
  PREDICT classes of data using model
  GENERATE and PRINT classification report comparing predictions to true labels
DISPLAY Final Result:
  PRINT success rate and note on potential changes due to learning rate adjustments
END
```

ДОДАТОК Б

КОД ДЛЯ РЕАЛІЗАЦІЇ

```
# %% [markdown]
#
# <font color = 'blue'>
# Content:
#
# 1. [Load Dataset](#1)
# 1. [Visualizing the Dataset](#2)
# 1. [Normalization of image data](#3)
# 1. [Convolutional Neural Network (CNN) Model](#4)
#     * [Model Training](#5)
#     * [Accuracy and Loss Graphs](#6)
#     * [Classification Report](#7)
# 1. [Result](#8)
#

# %% [code]
import matplotlib.pyplot as plt
import seaborn as sns
import keras
from keras.models import Sequential
from keras.layers import Dense, Conv2D , MaxPool2D , Flatten , Dropout
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam, RMSprop, Adagrad
from keras.layers import BatchNormalization
from sklearn.metrics import classification_report,confusion_matrix
import tensorflow as tf
import cv2
import os
import numpy as np
import warnings
warnings.filterwarnings('ignore')

# %% [markdown]
# <a id = "1"></a><br>
# # Load Dataset

# %% [code]
labels = ['Negative', 'Positive']
img_size = 120
def read_images(data_dir):
    data = []
    for label in labels:
        path = os.path.join(data_dir, label)
        class_num = labels.index(label)
        for img in os.listdir(path):
            try:
                img_arr = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                resized_arr = cv2.resize(img_arr, (img_size, img_size))
                data.append([resized_arr, class_num])
            except Exception as e:
                print(e)
    return np.array(data)

Dataset = read_images('../input/surface-crack-detection')
```

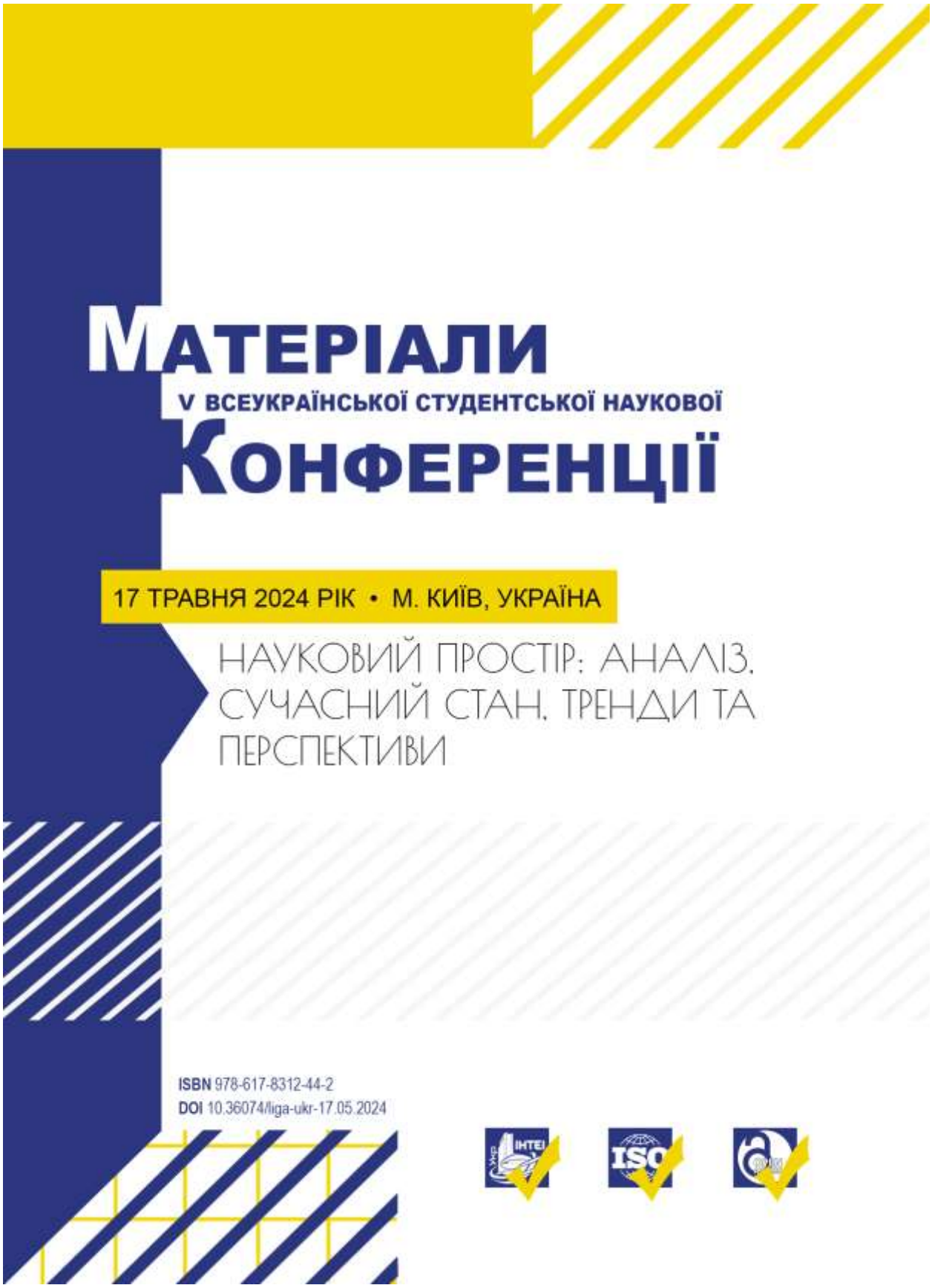
```
# %% [markdown]
# <a id = "2"></a><br>
# # Visualizing the Dataset

# %% [code]
lm = []
for i in Dataset:
   if(i[1] == 0):
      lm.append("Negative")
   elif(i[1] == 1):
      lm.append("Positive")

plt.figure(figsize=(10, 10))
plt.subplot(2, 2, 1)
sns.set_style('darkgrid')
axl = sns.countplot(lm)
axl.set_title("Number of Images")

# %% [markdown]
# <a id = "3"></a><br>
# # Normalization of image data

# %% [code]
x = []
y = []

for feature, label in Dataset:
   x.append(feature)
   y.append(label)

x = np.array(x).reshape(-1, img_size, img_size, 1)
x = x / 255
y = np.array(y)

# %% [code]
plt.subplot(1, 2, 1)
plt.imshow(x[1000].reshape(img_size, img_size), cmap='gray')
plt.axis('off')
plt.subplot(1, 2, 2)
plt.imshow(x[30000].reshape(img_size, img_size), cmap='gray')
plt.axis('off')

# %% [markdown]
# <a id = "4"></a><br>

# %% [code]
model = Sequential()
model.add(Conv2D(64,3,padding="same", activation="relu", input_shape = x.shape[1:]))
model.add(MaxPool2D())

model.add(Conv2D(64, 3, padding="same", activation="relu"))
model.add(MaxPool2D())

model.add(Conv2D(128, 3, padding="same", activation="relu"))
model.add(MaxPool2D())

model.add(Flatten())
```

```
model.add(Dense(256,activation="relu"))
model.add(Dropout(0.5))
model.add(BatchNormalization())
model.add(Dense(2, activation="softmax"))

model.summary()

# %% [markdown]
# <a id = "5"></a><br>

# %% [code]
opt = Adam(lr=1e-5)

model.compile(loss="sparse_categorical_crossentropy", optimizer=opt, metrics=["accuracy"])

history = model.fit(x, y, epochs = 15, batch_size = 128, validation_split = 0.25, verbose=1)

# %% [code]
print(history.history.keys())

# %% [markdown]
# <a id = "6"></a><br>
# # **Graphs**

# %% [code]
plt.figure(figsize=(12, 12))
plt.style.use('ggplot')
plt.subplot(2,2,1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Accuracy of the Model')
plt.ylabel('Accuracy', fontsize=12)
plt.xlabel('Epoch', fontsize=12)
plt.legend(['train accuracy', 'validation accuracy'], loc='lower right', prop={'size': 12})

plt.subplot(2,2,2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Loss of the Model')
plt.ylabel('Loss', fontsize=12)
plt.xlabel('Epoch', fontsize=12)
plt.legend(['train loss', 'validation loss'], loc='best', prop={'size': 12})

# %% [markdown]
# <a id = "7"></a><br>

# %% [code]
from sklearn.metrics import classification_report,confusion_matrix

predictions = model.predict_classes(x)
predictions = predictions.reshape(1,-1)[0]
print(classification_report(y, predictions, target_names = ['Negative','Positive']))

# %% [markdown]
# <a id = "8"></a><br>
#
#
# <div style="color:white;
```

```
#           display:fill;
#           border-radius:5px;
#           background-color:#F12C3B;
#           font-size:200%;
#           font-family:Cambria;
#           letter-spacing:0.5px">
#
# <p style="padding: 20px;
#               color:white;">Thank you
# </p>
# </div>
```

ДОДАТОК В

АПРОБАЦІЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ



УДК 082:001
Н 34

Голова оргкомітету: Коренюк І.О.

Верстка: Зрада С.І.
Дизайн: Бондаренко І.В.

Рекомендовано до видання Вченою Радою Інституту науково-технічної інтеграції та співпраці. Протокол № 36 від 16.05.2024 року.

Конференцію зареєстровано Державною науковою установою «УкрІНТЕІ» в базі даних науково-технічних заходів України та інформаційному бюлетені «План проведення наукових, науково-технічних заходів в Україні» (Посвідчення №29 від 05.01.2024).

Н 34 **Науковий простір: аналіз, сучасний стан, тренди та перспективи**: матеріали V Всеукраїнської студентської наукової конференції, м. Київ, 17 травня, 2024 рік / ГО «Молодіжна наукова ліга». — Вінниця: ТОВ «УКРЛОГОС Груп», 2024. — 586 с.

ISBN 978-617-8312-44-2
DOI 10.62732/liga-ukr-17.05.2024

Викладено матеріали учасників V Всеукраїнської мультидисциплінарної студентської наукової конференції «Науковий простір: аналіз, сучасний стан, тренди та перспективи», яка відбулася 17 травня 2024 року у місті Київ, Україна.

УДК 082:001

ISBN 978-617-8312-44-2

ВИКОРИСТАННЯ ЗАСОБІ МАШИННОГО НАВЧАННЯ ТА КОМП'ЮТЕРНИХ АЛГОРИТМІВ ДЛЯ АНАЛІЗУ ГЕНОМУ ЛЮДИНИ
Данько А.В., ***Науковий керівник: Левус Є.В.*** 344

ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ ВИЗНАЧЕННЯ АРХІТЕКТУРНОГО СТИЛЮ БУДІВЕЛЬ
Сьомко П.Я., ***Науковий керівник: Левус Є.В.*** 346

ДОСЛІДЖЕННЯ МЕТОДІВ РОЗРОБКИ ПРОГРЕСИВНИХ ТА ГІБРИДНИХ ВЕБ-ДОДАТКІВ ПОРІВНЯНО З НАТИВНИМИ ДОДАТКАМИ
Авербах Д.М., ***Науковий керівник: Русакова Н.Є.*** 348

КЛАСИФІКАЦІЯ ПРОГРАМНИХ ДЕТЕКТОРІВ ДЛЯ ВИЯВЛЕННЯ СОЦІАЛЬНИХ БОТІВ
Перегуда Я.І., ***Науковий керівник: Люшенко Л.А.*** 350

МЕТОДИ ПОРІВНЯННЯ ВЕБ-ДИЗАЙНУ: ЗАСОБИ ТА ПІДХОДИ
Суворова А.І., ***Науковий керівник: Татарников А.О.*** 353

ПСИХОЛОГІЧНІ АСПЕКТИ ВИКОРИСТАННЯ СОЦІАЛЬНИХ МЕРЕЖ СТУДЕНТАМИ: ПЕРЕВАГИ ТА НЕБЕЗПЕКИ
Тяжельников О.В., ***Науковий керівник: Татарников А.О.*** 355

РОЗРОБКА ТА РОЗГОРТАННЯ ВЕБ СИСТЕМИ З ВИКОРИСТАННЯМ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ ТА ШИНИ ПОВІДОМЛЕНЬ
Кулешов М.І., ***Науковий керівник: Кирій В.В.*** 357

РОЛЬ АЛГОРИТМІВ САМООРГАНІЗАЦІЇ В ПІДВИЩЕННІ ЕФЕКТИВНОСТІ РОЇВ ДРОНІВ У РЕАЛЬНИХ УМОВАХ
Слюсаренко О.К. 359

СЕКЦІЯ 16.
ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ТА СИСТЕМИ

METHODS OF ASSESSING CYBER SECURITY RISKS OF INFORMATION SYSTEMS OF CRITICAL INFRASTRUCTURE OBJECTS
Рижий Є.В., ***Науковий керівник: Наконечний В.С.*** 361

POSSIBILITIES OF USING COLOR MODELS IN COMPUTER GRAPHICS
Anano Kurtauli, ***Scientific Supervisor: Makasarasvili T.*** 364

PREDICTING ENVIRONMENTAL TRENDS USING DEEP LEARNING: INSIGHTS FROM CESIUM-137 ANALYSIS
Khobor O., ***Scientific Supervisor: Lytvyn V.*** 367

АЛГОРИТМ ВИЯВЛЕННЯ ДРОНІВ З ВИКОРИСТАННЯМ YOLOV8
Капкунов А., ***Науковий керівник: Майків І.М.*** 370

АЛГОРИТМ ВИЯВЛЕННЯ ТРІЩИН НА ПОВЕРХНЯХ НА ОСНОВІ CNN
Сичов Р., ***Науковий керівник: Сапожник Г.В.*** 372

Сичов Руслан, здобувач вищої освіти факультету
комп'ютерних інформаційних технологій
Західноукраїнськийого національний університет, Україна

Науковий керівник: Сапожник Г.В., канд. іст. наук, канд. техн. наук, доцент,
доцент кафедри інформаційно-обчислювальних систем і управління
Західноукраїнськийого національний університет, Україна

АЛГОРИТМ ВИЯВЛЕННЯ ТРІЩИН НА ПОВЕРХНЯХ НА ОСНОВІ CNN

Далі розглянемо алгоритм побудови та тренування згорткової нейронної мережі (CNN), яка використовується для аналізу зображень. Починаючи з завантаження та візуалізації даних, алгоритм включає нормалізацію даних, конструювання багатошарової CNN, її тренування та аналіз результатів через звіти про класифікацію та точність. Кожен крок алгоритму детально описано у відповідному розділі, а його схематичне представлення можна знайти на рисунку 1, який ілюструє послідовність дій та їх зв'язки, демонструючи систематичний підхід до розробки ефективних нейронних мереж для обробки візуальних даних.

Крок 1. Завантаження набору даних: Початковий етап, де збираються і завантажуються дані, що будуть використовуватися для тренування моделі. Це може бути здійснено через зчитування зображень із заданого директорію, де кожне зображення має певний клас (наприклад, позитивний або негативний).

Крок 2. Візуалізація набору даних: Аналіз і візуалізація даних допомагають зрозуміти розподіл класів та особливості даних, що може вплинути на подальше моделювання. Зазвичай це робиться за допомогою графічних засобів, таких як Matplotlib або Seaborn.

Крок 3. Нормалізація даних зображень: Для підготовки зображень до обробки нейронною мережею їх потрібно нормалізувати, змінивши масштаб значень пікселів до діапазону від 0 до 1, що допомагає покращити конвергенцію під час тренування.

Крок 4. Побудова моделі CNN: Згорткова нейронна мережа будується із використанням кількох шарів, таких як згорткові шари, шари пулінгу (MaxPooling), шари активації (ReLU), повнозв'язні шари (Dense) та шари відкидання (Dropout) для запобігання перенавчанню.

Крок 5. Тренування моделі: Модель тренується на підготовлених даних, використовуючи втрати та оптимізатор (наприклад, Adam). Важливі параметри, такі як розмір пакету, кількість епох і розділення на тренувальну та тестову вибірки, встановлюються для керування процесом тренування.

Крок 6. Графіки точності та втрат: Після тренування моделі аналізуються графіки, що показують зміну точності та втрат на тренувальних та валідаційних даних, що допомагає визначити, чи має місце перенавчання або недонавчання.

Крок 7. Звіт про класифікацію: Виконується оцінка моделі на тестових даних, результати якої представляються у вигляді звіту про класифікацію, що включає точність, повноту та F1-оцінку для кожного класу.

Крок 8. Результат: Виводиться загальний результат роботи моделі, де аналізуються загальна точність та інші метрики. Результати можуть

використовуватися для прийняття рішень щодо подальшого вдосконалення моделі або її використання в реальних умовах.

Рис. 1. **Алгоритм виявлення тріщин на поверхнях на основі CNN**

У висновку, побудова та тренування згорткової нейронної мережі (CNN) для аналізу зображень демонструє значну ефективність у класифікації та розпізнаванні візуальних патернів. Алгоритм, включаючи його покрокове виконання та ілюстрацію на рисунку 1, виявився здатним ефективно вирішувати задачі класифікації з високою точністю. Це, в свою чергу, може слугувати основою для подальших досліджень та розробок у галузі комп'ютерного зору та машинного навчання.

Список використаних джерел:

1. Bilokon O. S. Software Architecture of Navigation Systems for Control Modules of Robotics. Èlektronnoe modelirovanie. 2023. Vol. 45, no. 5. P. 103–112. URL: https://doi.org/10.15407/emodel.45.05.103 (date of access: 13.05.2024).