**Міністерство освіти і науки України**
**Тернопільський національний педагогічний університет**
**імені Володимира Гнатюка**

Інженерно-педагогічний факультет

Кафедра комп'ютерних технологій

**Кваліфікаційна робота**

# ПІДГОТОВКА МАЙБУТНІХ ФАХІВЦІВ ІТ-ТЕХНОЛОГІЙ У ЗАКЛАДАХ ФАХОВОЇ ПЕРЕДВИЩОЇ ОСВІТИ ДО РОЗРОБКИ НЕЙРОННИХ МЕРЕЖ

Спеціальність 015 Професійна освіта
Спеціалізація 015.39 Цифрові технології

**Освітня-наукова програма**
**«Професійна освіта (Комп'ютерні технології)»**

ВИКОНАВ:
здобувач вищої освіти
освітнього рівня «магістр»
МАТВІЇШИН Максим Володимирович

НАУКОВИЙ КЕРІВНИК:
кандидат педагогічних наук, доцент
СІТКАР Тарас Вікторович

РЕЦЕНЗЕНТ:
доктор пед.наук, професор
ГОРБАТЮК Роман Михайлович

Робота захищена з оцінкою:
Національна шкала _____
Кількість балів:__ Оцінка: ECTS_

Тернопіль – 2026

**Міністерство освіти і науки України**
**Тернопільський національний педагогічний університет**
**імені Володимира Гнатюка**
Інженерно-педагогічний факультет
Кафедра комп'ютерних технологій

**ЗАВДАННЯ**
ДЛЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ СТУДЕНТУ
МАТВІЇШИНУ Миксиму Володимировичу

на тему:
**«Підготовка майбутніх фахівців ІТ-технологій у закладах фахової передвищої освіти до розробки нейронних мереж»**

Спеціальність: 015 Професійна освіта

Спеціалізація: 015.39 Цифрові технології
Освітньо-наукова програма: Професійна освіта (Комп'ютерні технології)

НАУКОВИЙ КЕРІВНИК: кандидат педагогічних наук, доцент, Сіткар Тарас Вікторович.

Термін подання роботи і супроводжувальних документів:
до 18.05.2026 року

Зміст (перелік основних питань, які потрібно розкрити):

1. Проаналізувати науково-методичну літературу щодо підготовки фахівців з комп'ютерних технологій у контексті інтеграції технологій штучного інтелекту.
2. Розробити методику навчання розробки нейронних мереж, що інтегрує hands-on підхід, сучасні інструменти та поетапну проєктну діяльність.
3. Організувати та провести педагогічний експеримент з апробації розробленої методики.

Перелік додаткових матеріалізованих результатів роботи: рисунки, програмний код, таблиці, графіки.

**ГРАФІК ПІДГОТОВКИ**

**КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ**

| № з/п | ПЕРЕЛІК РОБІТ | Термін виконання | | Відмітка про виконання |
|---|---|---|---|---|
| | | І рік навч. (2024-2025) | ІІ рік навч. (2025-2026) | |
| 1 | Вибір теми, затвердження її на засіданні кафедри, закріплення наукового керівника | жовтень-листопад | | |
| 2 | Складання плану роботи і графіку її підготовки, узгодження з науковим керівником | листопад | | |
| 3 | Вивчення літературних і електронних джерел, збір та узагальнення фактів, даних | жовтень-січень | | |
| 4 | Розробка методики дослідження. Проведення пошукового дослідження | грудень-квітень | | |
| 5 | Написання розділу 1, подання його для перевірки керівнику | травень | | |
| 6 | Написання розділів 2–3, подання для перевірки керівнику | | вересень-лютий | |
| 7 | Завершення написання роботи, оформлення її згідно з вимогами, подання науковому керівнику | | березень | |
| 8 | Подання роботи на зовнішнє рецензування | | квітень | |
| 9 | Попередній захист роботи на засіданні кафедри | | квітень | |
| 10 | Подання кваліфікаційної роботи та супроводжувальних документів | | травень | |
| 11 | Захист роботи на засіданні Екзаменаційної комісії | | за розкладом | |

Графік узгоджено: «14» жовтня 2024 р.

Науковий керівник _____ Сіткар Т. В.
(підпис)

Виконавець кваліфікаційної роботи _____ Матвіїшин М. В.
(підпис)

# АНОТАЦІЯ

**Матвіїшин М. В.** Підготовка майбутніх фахівців ІТ-технологій у закладах фахової передвищої освіти до розробки нейронних мереж. – Кваліфікаційна робота за спеціальністю 015 Професійна освіта спеціалізації 015.39 Цтфрові технології. Тернопільський національний педагогічний університет імені Володимира Гнатюка. Тернопіль, 2026. – 91 с.

У магістерській роботі розроблено, теоретично обґрунтовано та експериментально перевірено методику навчання розробці нейронних мереж для студентів фахових коледжів, спрямовану на формування інженерних компетентностей рівня НРК 5. Дослідження ґрунтується на п’ятиетапній дидактичній моделі, що інтегрує повний життєвий цикл машинного навчання: від інженерії даних та анотації датасетів до архітектурного проєктування, налаштування конвеєра навчання, валідації моделей та їхнього базового деплою. Методика враховує психолого-педагогічні особливості засвоєння складних алгоритмічних дисциплін, застосовує трансферне навчання як дидактичний інструмент подолання обчислювальних обмежень, scaffolded-підхід для поступового зняття когнітивного навантаження та hands-on методологію, що забезпечує раннє залучення студентів до практичної діяльності з використанням сучасного фреймворку TensorFlow/Keras. У роботі також запропоновано критеріально-діагностичний апарат оцінювання, який поєднує технічні метрики машинного навчання з педагогічними рівнями сформованості професійних умінь, та розроблено готовий до інтеграції навчально-методичний комплекс, адаптований до інфраструктурних та організаційних умов закладів фахової передвищої освіти України.

Робота складається з 87 сторінок основного тексту, який включає 7 рисунків, 5 таблиць.

*Ключові слова:* методика навчання, нейронні мережі, глибоке навчання, TensorFlow, професійна освіта, фаховий коледж, практико-орієнтоване навчання, компетентнісний підхід, педагогічний експеримент, оцінювання навичок.

# ABSTRACT

**Matviishyn M. V.** Training of future IT specialists at institutions of vocational pre-higher education in the development of neural networks. – Master's thesis in the field of study 015 Vocational Education, specialisation 015.39 Digital Technologies. Volodymyr Hnatiuk Ternopil National Pedagogical University. Ternopil, 2026. – 91 pp.

This master's thesis develops, theoretically substantiates and experimentally verifies a methodology for teaching the development of neural networks to vocational college students, aimed at fostering engineering competences at NQF Level 5. The research is based on a five-stage didactic model that integrates the full life cycle of machine learning: from data engineering and dataset annotation to architectural design, training pipeline configuration, model validation and their basic deployment. The methodology takes into account the psychological and pedagogical characteristics of learning complex algorithmic disciplines, applies transfer learning as a didactic tool to overcome computational constraints, a scaffolded approach to gradually reduce cognitive load, and a hands -on methodology, which ensures the early involvement of students in practical activities using the modern TensorFlow/Keras framework. The paper also proposes a criteria-based diagnostic assessment framework that combines technical machine learning metrics with pedagogical levels of professional skill development, and presents a ready-to-integrate teaching and learning package adapted to the infrastructural and organisational conditions of pre-higher vocational education institutions in Ukraine.

The paper consists of 87 pages of main text, including 7 figures, 5 tables.

*Keywords:* teaching methodology, neural networks, deep learning, TensorFlow, vocational education, vocational college, practice-oriented learning, competence-based approach, pedagogical experiment, skills assessment. practices for ensuring reproducibility

**ЗМІСТ**

# ВСТУП

Стрімка цифровізація економіки, індустріалізація сектору інформаційних технологій та глобальна трансформація ринку праці обумовлюють формування нових вимог до змісту та якості професійної освіти. Штучний інтелект (ШІ), а зокрема технології глибокого навчання та нейронних мереж, перестали бути виключно академічною або дослідницькою галуззю й увійшли до ядра виробничих процесів у таких сферах, як комп'ютерний зір, обробка природної мови, прогнозна аналітика, автоматизація інженерних рішень та кібербезпека. За даними світових аналітичних агентств, попит на фахівців, здатних розробляти, налаштовувати та супроводжувати нейромережеві рішення, зростає експоненційно, водночас спостерігається стійкий дефіцит кадрів середньої ланки, які володіють не лише теоретичними знаннями, а й практичними навичками розробки, тестування та інтеграції моделей у реальні програмні продукти.

В Україні ця тенденція посилюється процесами відновлення та модернізації економіки, інтеграції до єдиного цифрового простору ЄС та реалізації Національної стратегії розвитку штучного інтелекту до 2030 року. У цих умовах система професійної освіти (заклади фахової передвищої освіти – коледжі, технікуми) має стати ключовою ланкою підготовки конкурентоспроможних фахівців рівня «фаховий молодший бакалавр», які з першого дня працевлаштування здатні виконувати прикладні завдання у сфері ІТ. Проте, аналіз чинних освітніх програм та суміжних ІТ-спеціальностей у коледжах засвідчує, що навчання розробки нейронних мереж або відсутнє взагалі, або реалізується фрагментарно, переважно на теоретичному рівні, без належного інструментального та методичного забезпечення. Це призводить до розриву між освітніми результатами та вимогами роботодавців, знижує мотивацію студентів до

вивчення складних алгоритмічних дисциплін та обмежує їхні можливості кар'єрного старту у сфері сучасних технологій.

Особливу педагогічну складність становить адаптація змісту нейромережевих технологій до освітнього рівня коледжу. Студенти фахових коледжів, як правило, мають меншу базову математичну підготовку порівняно з бакалаврами університетів, коротший термін навчання (2–3 роки) та орієнтацію на швидке набуття прикладних умінь. Традиційні академічні курси з машинного навчання, побудовані на глибокому математичному апараті (лінійна алгебра, теорія ймовірностей, диференціальне числення), є надто абстрактними та неефективними для цього контингенту. Водночас, сучасний інструментарій розробки (Python, TensorFlow, PyTorch, OpenCV, Google Colab, Hugging Face) значно знижує поріг входження, дозволяючи реалізовувати концепцію «hands-on learning» – навчання через безпосереднє програмування, експериментування з даними, візуалізацію процесів навчання моделей та створення мікропроєктів. Проте відсутність систематизованої методики, яка б інтегрувала технологічні можливості цих інструментів з педагогічними принципами професійної освіти, залишається суттєвою науково-практичною прогалиною.

Актуальність теми магістерської роботи зумовлена також необхідністю переосмислення ролі викладача ІТ-дисциплін у коледжі. Сучасний педагог має бути не лише транслятором знань, а й організатором проєктно-дослідницької діяльності, ментором у роботі з відкритими репозиторіями, фасилітатором командної розробки та експертом з оцінювання якості коду. Це вимагає розробки чітких педагогічних умов, критеріальних моделей оцінювання та структурованих навчальних сценаріїв, які б забезпечували поступове формування компетентностей: від розуміння архітектур мереж до самостійного навчання моделей на реальних наборах даних.

Таким чином, розробка науково обґрунтованої, технологічно адаптованої та педагогічно ефективної методики навчання розробки нейронних мереж для студентів коледжів є об'єктивною потребою сьогодення. Вона відповідає стратегічним пріоритетам розвитку освіти в Україні, сприяє підвищенню якості підготовки ІТ-фахівців середньої ланки та створює підґрунтя для подальшого інтегрування технологій ШІ в систему фахової передвищої освіти.

Проблематика впровадження технологій штучного інтелекту в освітній процес досліджується у працях вітчизняних та зарубіжних учених у галузях педагогіки, методики навчання інформатики та комп'ютерних наук, психології професійного становлення. Фундаментальні засади професійної освіти, компетентнісного підходу та практико-орієнтованого навчання розкрито у дослідженнях В. Андрєєва, С. Гончаренка, В. Лугового, О. Пехоти, І. Підласого, а також у міжнародних документах ЮНЕСКО та Європейської комісії щодо цифрової трансформації VET-сектору. Методологічні аспекти навчання програмуванню, алгоритмізації та роботі з даними представлено у працях М. Жалдака, О. Співаковського, Ю. Триуса, І. Чернишова. Питання дидактики машинного навчання та візуалізації нейромережевих процесів досліджували зарубіжні автори: I. Goodfellow, Y. Bengio, A. Courville, а також практико-орієнтовані розробники освітніх платформ (Coursera, DeepLearning.AI, Fast.ai), які демонструють ефективність поступового, інструментально насиченого підходу.

Водночас, аналіз наукової літератури засвідчує, що більшість публікацій зосереджено на університетському рівні підготовки, де передбачається глибоке вивчення математичних основ та дослідницька складова. Рівень фахових коледжів залишається недостатньо дослідженим: відсутні систематизовані методики адаптації складних концепцій глибокого навчання до навчальних планів коледжів, не розроблено педагогічних умов

формування прикладних навичок розробки нейронних мереж з урахуванням обмеженого навчального часу та специфіки контингенту студентів. Мало досліджень присвячено порівняльному аналізу дидактичного потенціалу TensorFlow, PyTorch та OpenCV у контексті освітнього процесу, а також критеріям оцінювання якості студентських проєктів на рівні молодшого бакалавра. Окремою прогалиною є відсутність емпірично підтверджених даних щодо ефективності hands-on підходу саме в умовах українських коледжів, де інфраструктурні можливості, рівень цифрової грамотності студентів та підготовка викладачів мають свої особливості.

Таким чином, попри наявність окремих напрацювань у галузі ІТ-педагогіки та освіти у сфері ШІ, проблема розробки цілісної методики навчання розробки нейронних мереж для студентів коледжів залишається недостатньо дослідженою, що зумовлює необхідність проведення спеціального педагогічного дослідження, спрямованого на заповнення цієї прогалини.

**Об’єкт** дослідження – процес підготовки студентів коледжів.

**Предмет** дослідження – методика навчання студентів коледжів принципів побудови нейронної мережі для розпізнавання об’єктів на основі фреймворку YOLO.

**Мета** дослідження – теоретично обґрунтувати, розробити та експериментально перевірити методику навчання розробки нейронних мереж, орієнтовану на формування практичних компетентностей студентів коледжів.

Для досягнення поставленої мети визначено такі **завдання** дослідження:

1. Проаналізувати науково-методичну літературу щодо підготовки фахівців з комп’ютерних технологій у контексті інтеграції технологій штучного інтелекту.

2. Розробити методику навчання розробки нейронних мереж, що інтегрує hands-on підхід, сучасні інструменти та поетапну проєктну діяльність.

3. Організувати та провести педагогічний експеримент з апробації розробленої методики.

**Гіпотеза** дослідження полягає в тому, що навчання розробки нейронних мереж у процесі підготовки студентів коледжу буде ефективним за умови реалізації спеціально розробленої методики, яка передбачає:

- поступове ускладнення змісту від базових архітектур до прикладних задач комп'ютерного зору та обробки даних;

- інтеграцію сучасних інструментів у лабораторно-проєктну діяльність з акцентом на hands-on підхід;

- створення педагогічних умов: диференціації навчальних завдань, інфраструктурного забезпечення (хмарні/локальні середовища), системного наставництва та формування проєктної культури;

- використання критеріально-орієнтованого оцінювання, що фіксує не лише теоретичні знання, а й практичні навички розробки, тестування, оптимізації та документації нейромережевих рішень.

Реалізація цих умов сприятиме підвищенню рівня сформованості професійних компетентностей, мотивації до самостійного навчання та готовності студентів коледжів до працевлаштування у сфері ІТ.

Для досягнення мети та вирішення поставлених завдань використано комплекс взаємопов'язаних методів, згрупованих за рівнями наукового пізнання:

*Теоретичні* методи:

- аналіз, синтез, узагальнення та систематизація наукової, навчально-методичної та нормативної літератури з проблем професійної освіти, дидактики ІТ, машинного навчання та педагогічного проєктування;

- педагогічне моделювання – побудова структурно-змістової моделі методики навчання розробки нейронних мереж, визначення її компонентів, зв'язків та функціональних характеристик;

- порівняльний аналіз інструментальних середовищ розробки з точки зору дидактичного потенціалу, доступності та відповідності вимогам освітнього процесу коледжу.

*Емпіричні* методи:

- спостереження за навчальною та проєктною діяльністю студентів під час лабораторних занять, фіксація динаміки засвоєння матеріалу, виявлення типових труднощів;

- тестування та діагностування – вхідне, проміжне та підсумкове оцінювання теоретичних знань, практичних умінь програмування, якості реалізації нейронних моделей;

- експертне оцінювання – аналіз студентських проєктів фахівцями-практиками та викладачами-методистами за розробленими рубриками (якість коду, архітектурна доцільність, ефективність моделі, документація);

- анкетування та інтерв'ювання – вивчення мотивації, самооцінки компетентностей, рефлексії студентів та викладачів щодо доцільності та ефективності запропонованих навчальних сценаріїв.

*Педагогічний* експеримент:

- організовано та проведено багатоетапний педагогічний експеримент (констатувальний, формувальний, контрольний) з використанням контрольної та експериментальної груп, що дозволило перевірити гіпотезу дослідження в реальних умовах навчального процесу коледжу.

*Методи математичної статистики*:

- кількісна та якісна обробка експериментальних даних з використанням t-критерію Стьюдента, коефіцієнта кореляції Пірсона, дисперсійного аналізу, побудови гістограм та діаграм розподілу рівнів

сформованості компетентностей для підтвердження достовірності отриманих результатів.

*Наукова новизна* отриманих результатів полягає в тому що, уточнено дидактичного потенціал інтеграції Python, TensorFlow, PyTorch та OpenCV у навчальний процес коледжу, а також виявленні оптимальних сценаріїв їх застосування на різних етапах формування компетентностей (від базових перцептронів до згорткових мереж для комп'ютерного зору).

Теоретичне значення дослідження полягає у розширенні наукових уявлень про дидактику професійної ІТ-освіти на рівні фахових коледжів, уточнено методології формування концептуальних засад інтеграції технологій глибокого навчання.

*Практичне значення* роботи полягає у розробці навчально-методичного комплексу з дисципліни «Інтелектуальні системи та машинне навчання», з можливістю для безпосереднього впровадження в освітній процес коледжів.

# РОЗДІЛ 1. ОСНОВНІ ПРИНЦИПИ ПІДГОТОВКИ ФАХІВЦІВ З КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ У КОНТЕКСТІ ШТУЧНОГО ІНТЕЛЕКТУ

## 1.1. Концептуальні вимоги до змісту професійної освіти в умовах цифровізації та розвитку технологій глибокого навчання

Цифрова трансформація економіки та суспільного простору України визначає стратегічні вектори модернізації системи професійної освіти. Ухвалення Національної стратегії розвитку штучного інтелекту в Україні до 2030 року, Концепції розвитку цифрової освіти та інтеграція національних освітніх стандартів до європейського простору (Bologna Process, EQF) формують нові концептуальні вимоги до змісту підготовки фахівців IT-галузі на рівні фахової передвищої освіти. Технології глибокого навчання (deep learning), зокрема архітектури згорткових нейронних мереж (CNN), механізми трансферного навчання (transfer learning) та підходи до детекції об'єктів (object detection), перетворилися з академічних експериментів на індустріальний стандарт розв'язання прикладних задач у сфері комп'ютерного зору, промислової аналітики, медичної діагностики та автоматизації контролю якості (рис.1.1).

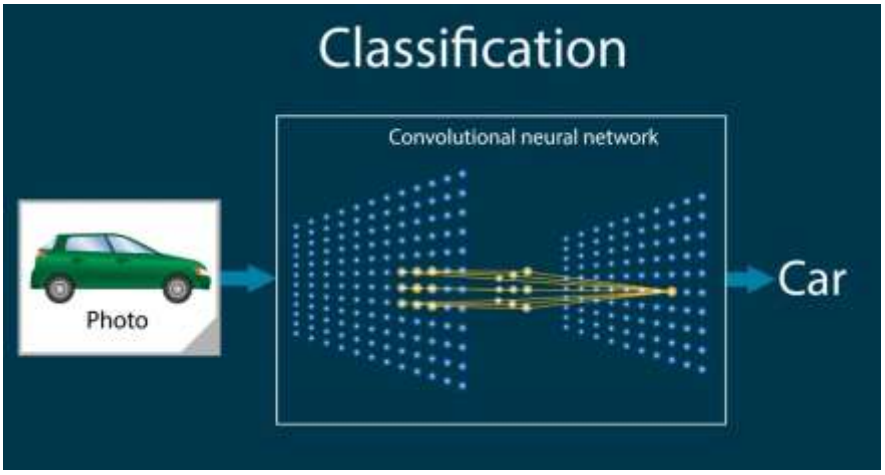


*Рисунок 1.1. Архітектура згорткової нейронної мережі (CNN)*

Це обумовлює фундаментальну зміну парадигми змісту професійної освіти: від ізольованого вивчення алгоритмічних конструкцій до формування інженерної культури розробки, валідації та інтеграції інтелектуальних систем у реальні виробничі процеси [1]. На рівні фахових коледжів, які готують фахових молодших спеціалістів (НРК рівень 5), ця трансформація вимагає узгодження навчального контенту з дескрипторами Національної рамки кваліфікацій, сучасними вимогами ринку праці та педагогічною доцільністю інтеграції складних програмно-алгоритмічних дисциплін у навчальні плани.

Нормативно-концептуальне підґрунтя змісту професійної освіти в Україні визначається Національною рамкою кваліфікацій (НРК), Державними стандартами фахової передвищої освіти, а також галузевими стандартами за спеціальностями галузі знань 12 «Інформаційні технології». НРК рівень 5 передбачає формування компетентностей, що охоплюють: систематизовані знання в межах фахового спрямування; здатність застосовувати теоретичні знання для розв’язання типових та атипових професійних задач; навички самостійної організації діяльності, комунікації в професійному середовищі та відповідальності за результати власної роботи; здатність до критичного аналізу, самоконтролю та адаптації до змін у технологічному середовищі [2]. У контексті підготовки фахівців з комп’ютерних технологій у сфері глибокого навчання ці дескриптори трансформуються у конкретні освітні результати: розуміння принципів функціонування нейронних мереж, вміння будувати та налаштовувати конвеєри навчання (training pipeline), здатність інтерпретувати метрики якості моделей, навички підготовки та анотації датасетів, базові вміння розгортання моделей на edge-пристроях або веб-сервісах, а також обізнаність щодо етичних та правових аспектів використання ШІ. Таким чином, зміст освіти має бути структурований навколо життєвого циклу

машинного навчання, а не навколо фрагментарних програмістських тем або суто теоретичного математичного апарату.

Дослідження проведено на базі Відокремленого структурного підрозділу «Тернопільський фаховий коледж» Тернопільського національного технічного університету імені Івана Пулюя, що зумовлює конкретизацію концептуальних вимог з урахуванням інституційного, регіонального та освітнього контексту. Коледж є частиною академічного кластеру ТНТУ ім. І. Пулюя, що забезпечує синергію між фаховою передвищою та вищою освітою, спільну інфраструктуру, методичну підтримку та можливість інтеграції студентів коледжу в науково-дослідні та інноваційні проєкти університету. Регіональний IT-сектор Тернопільщини та Західної України характеризується зростанням попиту на фахівців прикладного рівня, здатних швидко інтегруватися в команди розробки, працювати з реальними даними, адаптувати готові ML-рішення під специфіку замовника та забезпечувати підтримку інтелектуальних систем. Ці ринкові очікування безпосередньо впливають на зміст навчальних програм коледжу, вимагаючи орієнтації на практико-орієнтоване навчання, STEM-інтеграцію, проєктну діяльність та формування інженерної культури кодування [3]. Саме в цьому середовищі апробується запропонована методика навчання розробки нейронних мереж, що дозволяє верифікувати концептуальні вимоги в реальних умовах навчального процесу з урахуванням матеріально-технічного забезпечення, кваліфікації викладачів та психолого-педагогічних особливостей контингенту студентів.

Концептуальні вимоги до змісту підготовки фахівців з комп’ютерних технологій у сфері глибокого навчання визначаються низкою системних принципів. По-перше, зміст має бути модульним та гнучким, що дозволяє оновлювати окремі навчальні блоки без порушення логіки всієї програми. Індустрія машинного навчання характеризується темпом оновлення

інструментарію 12–18 місяців: з’являються нові архітектури, оптимізатори, фреймворки, практики MLOps та стандарти етичного ШІ. Модульна побудова забезпечує можливість інтеграції актуальних практик (наприклад, використання EfficientNet, TensorFlow Data Validation, Weights & Biases, Hugging Face Hub) без повного перегляду робочої програми дисципліни. По-друге, зміст має орієнтуватися на повний інженерний цикл розробки ML-рішень: збір даних → анотація → попередня обробка → конструювання архітектури → налаштування training pipeline → валідація → інференс → базовий деплой → моніторинг. Така структура усуває фрагментарність засвоєння знань, формує системне мислення та імітує реальні умови роботи в індустрії [4] (рис.1.2).

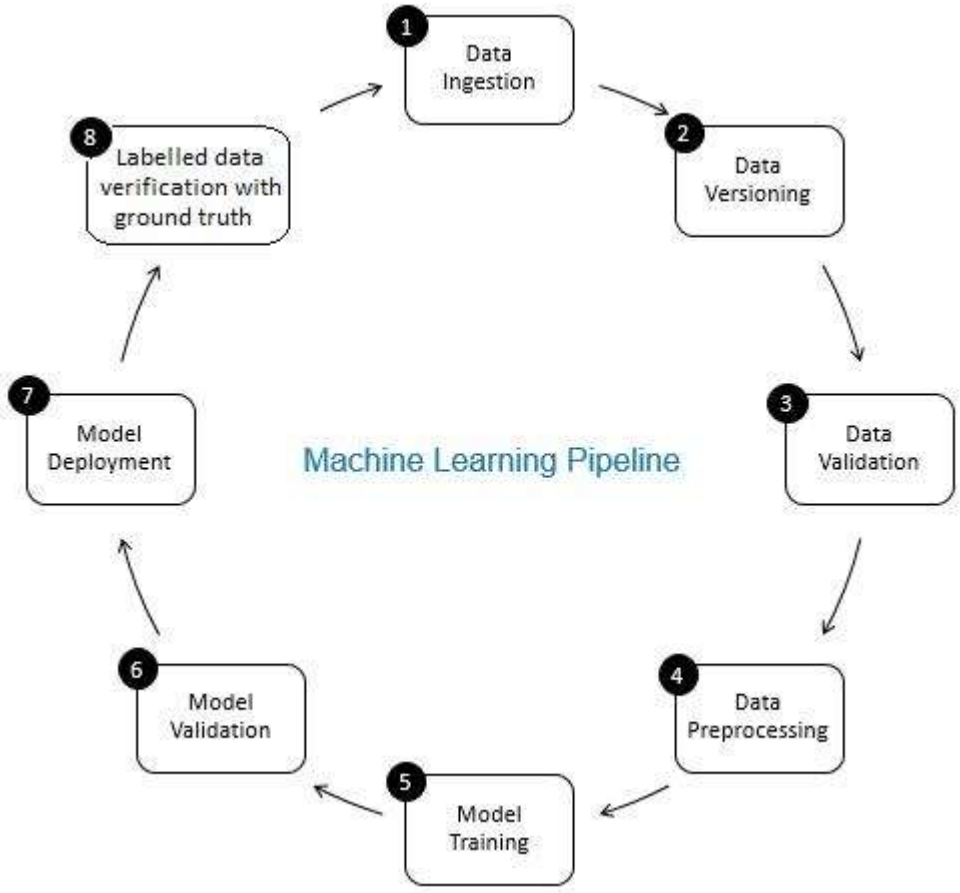


*Рисунок 1.2. Життєвий цикл розробки ML-системи*

По-третє, інтеграція STEM-підходу передбачає не механічне поєднання дисциплін, а створення єдиного навчального контексту, в якому математичні моделі, технологічні інструменти, інженерні принципи та науковий метод використовуються для розв'язання прикладних задач. У контексті коледжу це реалізується через проєктне навчання з реальними або імітаційними датасетами (наприклад, класифікація дефектів друкованих плат, детекція сільськогосподарських культур за аерознімками, аналіз стану інфраструктури за відеопотоком), що забезпечує формування практичних умінь та внутрішньої мотивації студентів [5].

Важливою концептуальною вимогою є адаптація змісту до обчислювальних та інфраструктурних реалій закладів фахової передвищої освіти. Локальні GPU-кластери у коледжах зазвичай відсутні або мають обмежену потужність, тому зміст має передбачати хмарно-орієнтовану архітектуру навчання: використання Google Colab, Kaggle Notebooks, GitHub для версійного контролю коду, DVC або Git LFS для управління датасетами, Roboflow/CVAT для анотації зображень, TF Hub для доступу до попередньо навчених моделей [6]. Це формує у студентів культуру reproducibility, collaboration та відстеження експериментів, що є стандартом у сучасній індустрії ШІ. Паралельно зміст має включати офлайн-резервування: локальні JupyterHub-сервери, контейнеризовані середовища (Docker), кешування датасетів, що забезпечує безперервність навчального процесу навіть у разі тимчасової відсутності стабільного інтернет-з'єднання. Така гібридна інфраструктурна модель відповідає концептуальним вимогам НРК рівня 5 щодо самостійності, організації діяльності та відповідальності за технічні результати.

Дидактичне проєктування змісту має враховувати психологічні особливості засвоєння складних програмно-алгоритмічних дисциплін студентами коледжу. Теорія когнітивного навантаження (J. Sweller) вказує на необхідність дозування абстрактних концепцій, використання

візуальних аналогій, поетапного нарощування складності та мінімізації стороннього навантаження на робочу пам'ять [7]. У контексті глибокого навчання це означає: початок із готових моделей та інференсу для формування інтуїтивного розуміння, перехід до модифікації архітектур, потім до самостійного налаштування конвеєра, і лише на фінальних етапах – до оптимізації гіперпараметрів та діагностики складних помилок. Scaffolding-підхід (дидактичне підтримання) реалізується через каркасні ноутбуки (scaffolded notebooks) з пропущеними логічними блоками, інтерактивні симуляції (TensorFlow Playground), покрокові інструкції з коментарями та чек-листи самоперевірки. Це знижує бар'єр страху перед «чорною скринькою» ШІ, формує впевненість у власних інженерних діях та відповідає вимогам НРК щодо здатності до самостійної організації навчальної та професійної діяльності [8].

Концептуальні вимоги також передбачають інтеграцію практико-орієнтованого підходу на рівні змістового ядра навчальної програми. Теоретичні блоки не викладаються ізольовано, а безпосередньо прив'язуються до практичних задач: пояснення принципу batch normalization супроводжується лабораторною роботою з порівнянням швидкості збіжності моделі з та без нормалізації; тема dropout ілюструється через діагностику overfitting на реальному датасеті; розділ про transfer learning починається з порівняння навчання «з нуля» та fine-tuning MobileNetV2 на обмеженому наборі зображень. Така інтеграція забезпечує формування глибокого розуміння причинно-наслідкових зв'язків між архітектурними рішеннями, гіперпараметрами та поведінкою моделі, що є критично важливим для майбутньої професійної діяльності фахівця НРК рівня 5. В умовах Тернопільського фахового коледжу це реалізується через модульні лабораторні роботи, інтегровані в реальні або імітаційні проєкти, з чіткою структурою: брифінг → підготовка середовища → робота з даними → розробка моделі → навчання та аналіз → презентація та рефлексія. Парна

робота, ротація ролей (data engineer, model architect, evaluator) та peer-review імітують реальні ML-команди, формуючи комунікативні та лідерські компетентності, передбачені дескрипторами НРК [9].

Окремою концептуальною вимогою є інтеграція етичних та правових аспектів у зміст підготовки. Глибоке навчання працює з даними, що часто містять персональну інформацію, можуть відтворювати соціальні упередження (bias) або використовуватися в системах з високими ризиками. Професійна освіта має формувати у студентів розуміння принципів responsible AI: прозорість, відтворюваність, аудит даних, обмеження збору, відповідальність за наслідки автоматизованих рішень, відповідність законодавству України у сфері захисту персональних даних та європейським регуляторним рамкам (EU AI Act, GDPR) [10]. Дидактично це реалізується через включення в лабораторні роботи етапів перевірки датасетів на незбалансованість, аналізу помилкових класифікацій (false positives/negatives), обговорення наслідків deploy-рішень та ознайомлення з національними нормативними актами. Етична компетентність стає невід'ємною частиною професійної культури майбутнього фахівця, що безпосередньо корелює з вимогою НРК щодо відповідальності за результати діяльності та дотримання професійних стандартів.

Концептуальні вимоги також передбачають узгодження змісту з сучасними стандартами оцінювання. Традиційна система балів, орієнтована на відтворення формул або виконання ізольованих програмістських завдань, не відображає реальних професійних навичок у сфері глибокого навчання. Натомість впроваджується критеріально-орієнтоване оцінювання, що інтегрує технічні метрики моделей (стабільність loss-кривих, F1-score, precision/recall, час інференсу) з педагогічними показниками (самостійність діагностики помилок, якість документації, здатність аргументувати вибір архітектури, командна взаємодія). Формувальне оцінювання набуває пріоритету: щотижневі код-рев'ю, аналіз

логів TensorBoard, короткі тести на інтерпретацію графіків навчання, самооцінка та peer-feedback. Підсумкове оцінювання базується на захисті проєкту з демонстрацією робочого інференсу, порівнянням baseline та оптимізованої версії моделі, аналізом confusion matrix та рефлексією щодо інженерних рішень. Така система забезпечує валідність оцінювання, його прозорість та відповідність компетентнісному підходу, закладеному в НРК рівень 5 [11].

Важливим концептуальним аспектом є орієнтація змісту на формування адаптивності та lifelong learning competence. Сфера глибокого навчання характеризується експоненційним ростом публікацій, появою нових архітектур (Transformers, Vision Transformers, Diffusion Models, Foundation Models) та швидкою зміною інструментів. Професійна освіта не може готувати студентів до конкретних версій фреймворків; вона має формувати здатність самостійно освоювати нові інструменти, читати документацію, аналізувати приклади коду, верифікувати результати та інтегрувати нові практики в існуючі конвеєри. Дидактично це реалізується через включення в програму завдань на самостійний пошук рішень, аналіз офіційної документації, порівняння підходів, написання технічних звітів та рефлексію щодо стратегій навчання. Формування метакогнітивних навичок стає не менш важливим, ніж технічне володіння Python чи TensorFlow, що безпосередньо відповідає вимогам НРК щодо здатності до самоорганізації, критичного мислення та професійного розвитку [12].

Концептуальні вимоги також передбачають інтеграцію змісту з іншими дисциплінами навчального плану коледжу. Глибоке навчання не існує у вакуумі; воно спирається на знання з лінійної алгебри, теорії ймовірностей, баз даних, мережевих технологій, інженерії програмного забезпечення та предметних доменів. Тому зміст має передбачати cross-disciplinary зв’язки: спільні проєкти з дисциплінами «Бази даних», «Комп’ютерні мережі», «Основи алгоритмізації», «Спеціалізовані

предметні курси». В умовах Тернопільського фахового коледжу це реалізується через узгодження графіків викладання, спільні методичні семінари, інтегровані лабораторні роботи та залучення студентів до спільних проєктів із викладачами суміжних дисциплін. Це формує цілісне професійне мислення та готує студентів до роботи в мультидисциплінарних командах, що є стандартом у сучасній індустрії ШІ та вимогою НРК рівня 5 щодо комунікації та співпраці [13].

Нарешті, концептуальні вимоги до змісту професійної освіти в умовах цифровізації та розвитку технологій глибокого навчання мають бути системно інтегровані з національною освітньою політикою та міжнародними практиками. Україна активно інтегрується до європейського освітнього простору, що вимагає відповідності змісту підготовки вимогам EQF, участі в міжнародних сертифікаційних програмах (Google TensorFlow Developer Certificate, AWS Machine Learning Foundations, Microsoft AI Engineering) та співпраці з індустріальними партнерами. Впровадження запропонованої методики в навчальний процес ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя дозволяє не лише верифікувати її ефективність у реальних умовах, а й сформувати тиражовану модель підготовки фахівців НРК рівня 5 у сфері прикладного ШІ, яка може бути адаптована до інших закладів фахової передвищої освіти України. Це створює теоретико-методичне підґрунтя для розробки конкретної методики навчання розробки нейронних мереж, що буде детально розглянуто в наступних розділах дослідження.

Підсумовуючи, концептуальні вимоги до змісту професійної освіти в умовах цифровізації та розвитку технологій глибокого навчання формують цілісну дидактичну систему, що базується на інженерному підході до машинного навчання, компетентнісному орієнтирі відповідно до НРК рівень 5, STEM-інтеграції, практико-орієнтованій методології, етичній відповідальності та формуванні адаптивності. Зміст має бути модульним,

гнучким, хмарно-орієнтованим, критеріально оцінюваним та орієнтованим на метакогнітивний розвиток. Інтеграція цих вимог у навчальний процес Тернопільського фахового коледжу забезпечує підготовку фахівців, здатних ефективно працювати з сучасним інструментарієм ШІ, вирішувати прикладні задачі, інтерпретувати результати та відповідально інтегрувати інтелектуальні системи у професійну діяльність, що відповідає сучасним вимогам ринку праці, національним стандартам та європейським освітнім тенденціям.

## 1.2. Психолого-педагогічні особливості засвоєння складних програмно-алгоритмічних дисциплін студентами фахових коледжів

Засвоєння складних програмно-алгоритмічних дисциплін, зокрема тих, що інтегрують технології штучного інтелекту та глибокого навчання, є одним із найбільш когнітивно та педагогічно навантажених процесів у системі фахової передвищої освіти. Ефективність цього процесу безпосередньо залежить від системного врахування психологічних закономірностей навчання, вікових особливостей студентів коледжів, специфіки алгоритмічного мислення та дидактичних умов, що забезпечують перехід від теоретичного розуміння до сформованих практичних навичок. У контексті підготовки фахівців НРК рівень 5 у Відокремленому структурному підрозділі «Тернопільський фаховий коледж» Тернопільського національного технічного університету імені Івана Пулюя це набуває особливої гостроти, оскільки навчальний процес має поєднувати академічну строгість з прикладною орієнтацією, враховувати різноманітність початкової підготовки студентів та інтегрувати сучасні інженерні практики в обмежених інфраструктурних умовах.

Віковий контингент студентів фахових коледжів переважно належить до діапазону 15–20 років, що відповідає пізньому юнацькому віку та початку ранньої дорослості. Згідно з дослідженнями в галузі когнітивної та вікової психології, на цьому етапі відбувається інтенсивне дозрівання префронтальної кори головного мозку, яка відповідає за виконавчі функції: планування, самоконтроль, абстрактно-логічне мислення, регуляцію емоцій під час виконання складних задач та здатність до метакогнітивної рефлексії. Водночас цей процес не є завершеним або однорідним для всіх студентів. Перехід від конкретно-образного до формально-логічного мислення відбувається нерівномірно, що проявляється у труднощах з ментальною маніпуляцією багаторівневими абстракціями, таких як тензорні операції, градієнтний спуск у багатовимірному просторі, принципи регуляризації або механізми backpropagation. Теорія когнітивного навантаження (J. Sweller, P. Ayres, 2020) класифікує навантаження на три типи: внутрішнє (об’єктивна складність матеріалу), зовнішнє (неоптимальна дидактична організація) та релевантне (активна обробка інформації для формування схем). У випадку глибокого навчання внутрішнє навантаження є об’єктивно високим, тому педагогічне завдання полягає в мінімізації зовнішнього навантаження через структуровану подачу, візуалізацію процесів, поетапне нарощування складності та використання дидактичного підтримання (scaffolding), одночасно максимізуючи релевантне навантаження через активну експериментальну практику та рефлексію.

Специфіка засвоєння програмно-алгоритмічних дисциплін у контексті штучного інтелекту відрізняється від традиційного імперативного програмування фундаментальною відмінністю: студент працює не з детермінованими алгоритмами, де вхід однозначно визначає вихід, а з імовірнісними моделями, поведінка яких залежить від якості даних, ініціалізації ваг, гіперпараметрів навчання та стохастичних процесів

оптимізації. Це створює унікальний психологічний бар'єр, який у педагогічній практиці отримав назву «синдром чорної скриньки» (рис.1.3).

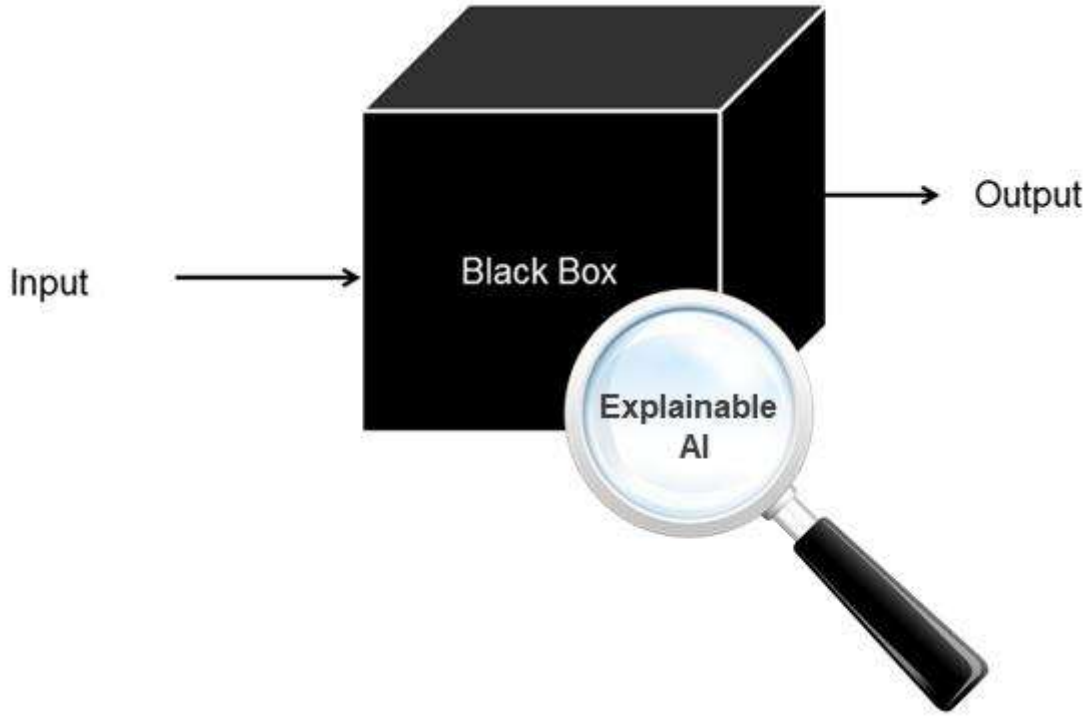


*Рисунок 1.3. Візуалізація принципу «black box» у нейронних мережах*

Студент часто не розуміє, чому модель класифікує об'єкт певним чином, чому крива втрат не збігається або чому зміна learning rate призводить до колапсу навчання. Без належного дидактичного супроводу це призводить до демотивації, когнітивного уникнення, механічного копіювання коду з туторіалів без розуміння причинно-наслідкових зв'язків або формування хибного уявлення про ШІ як про «магічний інструмент». Специфіка викладання дисциплін зі штучного інтелекту полягає в необхідності балансу між математичною строгістю та інженерною практичністю. Дослідження в галузі педагогіки інформатики свідчать, що студенти фахових коледжів значно ефективніше засвоюють матеріал, коли бачать прямий зв'язок між зміною коду та результатом: наприклад, як додавання шару Batch Normalization впливає на стабільність градієнтів, як зміна розміру фільтра в згортковому шарі модифікує виділення

просторових ознак, або як dropout зменшує перенавчання на тестовій вибірці. Це вимагає переходу від лекційно-репродуктивної моделі до експериментально-дослідницької, де помилка розглядається не як негативний результат, а як джерело навчальної інформації та точка для діагностики.

Мотиваційно-емоційний профіль студентів коледжу має змішаний характер. Зовнішня мотивація (отримання диплома, працевлаштування, конкурентоспроможність на ринку праці, матеріальні стимули) поєднується з внутрішньою (цікавість до інноваційних технологій, бажання створювати інтелектуальні продукти, самореалізація в ІТ-сфері). У контексті глибокого навчання внутрішня мотивація є критичним фактором стійкості до труднощів та когнітивної наполегливості. Теорія самовизначення (E. Deci, R. Ryan) підкреслює, що задоволення трьох базових психологічних потреб – автономії, компетентності та пов’язаності – забезпечує глибоке залучення в навчальний процес. Дидактично це реалізується через надання студентам вибору тем проєктів, можливість самостійного підбору датасетів або доменів застосування, роботу в парах або малих групах, а також через формування культури відкритого обміну кодом, результатами експериментів та аналізу помилок. Зниження рівня тривожності перед складними задачами досягається через поетапне scaffolded-навчання, де кожен крок має чіткі критерії успіху, а викладач виступає не як джерело готових відповідей, а як фасилітатор діагностики, що спрямовує студента до самостійного аналізу логів, графіків навчання та інтерпретації метрик.

Дидактичні умови ефективного навчання складним програмно-алгоритмічним дисциплінам у контексті глибокого навчання мають бути системно інтегровані в навчальний процес коледжу. По-перше, це поетапне зняття когнітивного навантаження через принцип «від інференсу до навчання»: початковий етап передбачає запуск попередньо навченої моделі, експерименти з вхідними даними, спостереження за результатами

класифікації або детекції, що формує інтуїтивне розуміння та знижує психологічний бар'єр. По-друге, реалізація hands-on methodology з першого заняття через інтерактивні середовища (Google Colab, Jupyter Notebook, TensorFlow Playground), де студенти відразу взаємодіють з кодом, змінюють параметри та спостерігають наслідки в реальному часі. По-третє, обов'язкова візуалізація абстрактних процесів: використання TensorBoard для моніторингу кривих втрат та точності, confusion matrix для аналізу помилок класифікації, Grad-CAM або Saliency Maps для інтерпретації рішень моделі, що перетворює «чорну скриньку» на «скляну» та розвиває аналітичне мислення (рис.1.4).

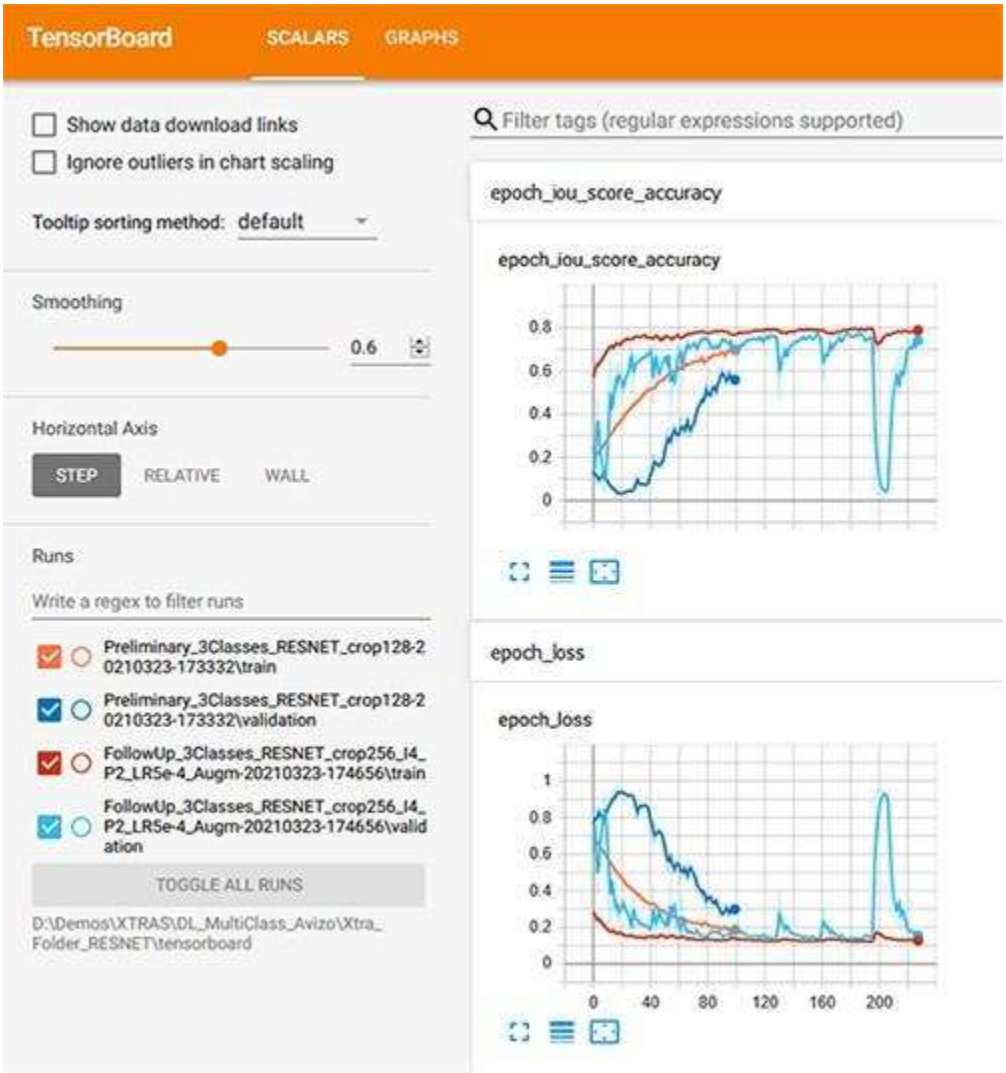


*Рисунок 1.4. Інтерфейс TensorBoard для моніторингу процесу навчання нейронної мережі*

По-четверте, ітеративність та рефлексія: навчальний цикл будується за принципом «спробуй → проаналізуй → скоригуй → документуй → презентуй», що формує інженерну культуру та толерантність до помилок. По-п’яте, інтеграція STEM-підходу та проєктного навчання, де математичні моделі, технологічні інструменти, інженерні принципи та предметні домени об’єднуються в єдиний контекст реального або імітаційного проєкту.

Принципи формування практичних умінь у сфері розробки нейронних мереж ґрунтуються на компетентнісному підході та адаптованій до ІТ-освіти теорії поетапного формування розумових дій. Процес проходження через стадії: орієнтація (розуміння задачі, інструментів, обмежень) → матеріалізоване виконання (робота з каркасним кодом, модифікація готових рішень) → вербалізоване/внутрішнє (самостійне написання архітектури, налагодження pipeline, діагностика помилок) → автоматизація (впевнене застосування в нових контекстах, оптимізація, деплой). У контексті коледжу це реалізується через модульні лабораторні роботи з чіткою структурою: технічний брифінг → підготовка середовища → робота з даними (анотація, аугментація, розподіл) → побудова моделі → налаштування конвеєра → навчання та аналіз → валідація → інференс/деплой → рефлексія. Важливим принципом є інтеграція інженерних практик: версійний контроль коду (Git), управління даними (DVC/Git LFS), логування експериментів, що формує не лише програмерські, а й професійні навички, відповідні до вимог НРК рівень 5. Принципи системності, послідовності, науковості, доступності та зв’язку теорії з практикою набувають нового змісту в умовах AI-освіти: системність проявляється в цілісності ML-конвеєра, послідовність – в поетапному нарощуванні абстракції, науковість – в опорі на емпірично перевірені архітектурні рішення, доступність – в адаптації складності до когнітивних можливостей студентів, зв’язок теорії з практикою – в

безпосередній прив’язці кожного теоретичного блоку до лабораторного експерименту.

Методи оцінювання сформованості навичок у сфері глибокого навчання вимагають відмови від традиційних тестів на відтворення формул чи синтаксису на користь критеріально-діагностичного апарату, що інтегрує технічні метрики машинного навчання з педагогічними показниками. Формувальне оцінювання включає: щотижневі код-рев’ю з фокусом на читабельність, структурованість та коментування; аналіз логів навчання (стабільність loss, динаміка accuracy, корекція learning rate); короткі тести на інтерпретацію графіків навчання та діагностику overfitting/underfitting; самооцінку та peer-feedback за структурованими рубриками. Підсумкове оцінювання базується на захисті проєкту з демонстрацією робочого інференсу, порівнянням baseline та оптимізованої версії моделі, аналізом метрик (precision, recall, F1-score, ROC-AUC, inference time) та рефлексією щодо інженерних рішень. Методи оцінювання мають бути валідними, надійними та прозорими, з чітко визначеними рівнями сформованості: репродуктивний (відтворення за зразком з мінімальними модифікаціями), адаптивний (самостійна діагностика помилок, корекція гіперпараметрів, обґрунтування архітектурних рішень), інноваційний (кастомізація функцій втрат, оптимізація під edge-пристрої, інтеграція власних датасетів, деплой у production-симуляції). У умовах ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя це реалізується через цифрові портфоліо студентів, інтеграцію з LMS (Moodle), використання автоматизованих скриптів перевірки коду та формалізовані рубрики, що враховують технічну якість, аналітичне мислення, документування та комунікативні навички.

Інституційний контекст дослідження обумовлює специфічні психолого-педагогічні умови навчання. Контингент коледжу характеризується різноманітністю початкової підготовки: частина

студентів має досвід у веб-розробці, базовому Python або роботі з базами даних, інша частина засвоює програмування та алгоритмічне мислення вперше. Це вимагає диференціації завдань, використання вхідної діагностики когнітивного стилю та рівня технічної підготовки, а також гнучкої модульної структури, що дозволяє компенсувати прогалини без уповільнення загального темпу. Інфраструктура коледжу включає комп'ютерні класи, стабільний інтернет-доступ, хмарні сервіси, але обмежена в локальних GPU-ресурсах, що обумовлює орієнтацію на оптимізовані моделі (MobileNetV2, EfficientNet-Lite, YOLOv8n), використання Google Colab, TF Lite для edge-інференсу та методи трансферного навчання як дидактичний інструмент подолання обчислювальних обмежень. Викладацький склад має досвід у програмуванні, суміжних дисциплінах та методичній роботі, проте потребує інтеграції сучасних AI-практик у навчальні плани. У цьому контексті запропонована методика враховує психологічні, когнітивні та організаційні реалії, трансформуючи їх у дидактичні переваги через peer-навчання, менторство старшокурсників, спільні проєкти з кафедрами університету, залучення до регіональних IT-хакатонів та формування внутрішньої спільноти розробників, що підвищує соціальну пов'язаність та професійну ідентифікацію студентів.

Особливу увагу слід приділити формуванню толерантності до невизначеності, яка є ключовою психологічною навичкою для фахівців у сфері глибокого навчання. На відміну від класичного програмування, де помилка компіляції має чіткий стек-трейс, у ML-конвеєрі модель може «мовчазно» погіршувати якість через data leakage, неправильну нормалізацію, незбалансованість класів або нестабільний learning rate. Дидактичне завдання полягає у формуванні навички системної діагностики: перевірка розподілу даних, аналіз кривих навчання, порівняння з baseline, ізоляція змінних, використання контролерів (callbacks), інтерпретація

помилок класифікації. Це розвиває критичне мислення, системний підхід та професійну стійкість, що прямо корелює з вимогами НРК рівень 5 щодо здатності до аналізу, самоконтролю та адаптації. Інтеграція етичного та правового дискурсу (відповідність Закону України «Про захист персональних даних», GDPR, EU AI Act, принципи responsible AI) також формує психологічну відповідальність за наслідки автоматизованих рішень, що є невід'ємною частиною професійної зрілості майбутнього фахівця.

Підсумовуючи, психолого-педагогічні особливості засвоєння складних програмно-алгоритмічних дисциплін студентами фахових коледжів визначаються взаємодією когнітивних обмежень, мотиваційних профілів, специфіки імовірнісного моделювання та дидактичних умов навчального середовища. Ефективне формування практичних умінь у сфері розробки нейронних мереж вимагає інтеграції scaffolded-підходу, hands-on methodology, візуалізації процесів, ітеративної рефлексії та критеріально-діагностичного оцінювання, що поєднує технічні метрики ML з педагогічними показниками компетентності. В умовах ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя це реалізується через адаптацію навчального контенту до інфраструктурних реалій, диференціацію завдань, інтеграцію STEM- та проєктного підходів, формування інженерної культури та розвиток толерантності до невизначеності. Дотримання цих психолого-педагогічних умов забезпечує перехід від механічного засвоєння синтаксису до сформованих інженерних навичок, що відповідають вимогам НРК рівень 5 та сучасним очікуванням ринку праці в галузі штучного інтелекту.

### 1.3. Аналіз інструментального середовища розробки нейронних мереж

Інструментальне середовище розробки нейронних мереж виступає ключовим технологічним та дидактичним компонентом у процесі підготовки фахівців з комп'ютерних технологій на рівні фахової передвищої освіти. Вибір програмного фреймворку безпосередньо детермінує ефективність засвоєння матеріалу, швидкість формування практичних умінь, можливість реалізації повного життєвого циклу машинного навчання (ML pipeline) та відповідність навчальних результатів вимогам ринку праці та Національної рамки кваліфікацій (НРК рівень 5). У контексті інтеграції технологій глибокого навчання в навчальні програми коледжу інструментальне середовище має розглядатися не лише як сукупність бібліотек та API, а як цілісна дидактична екосистема, що забезпечує перехід від абстрактних математичних моделей до інженерних рішень у реальних умовах. (рис.1.5).

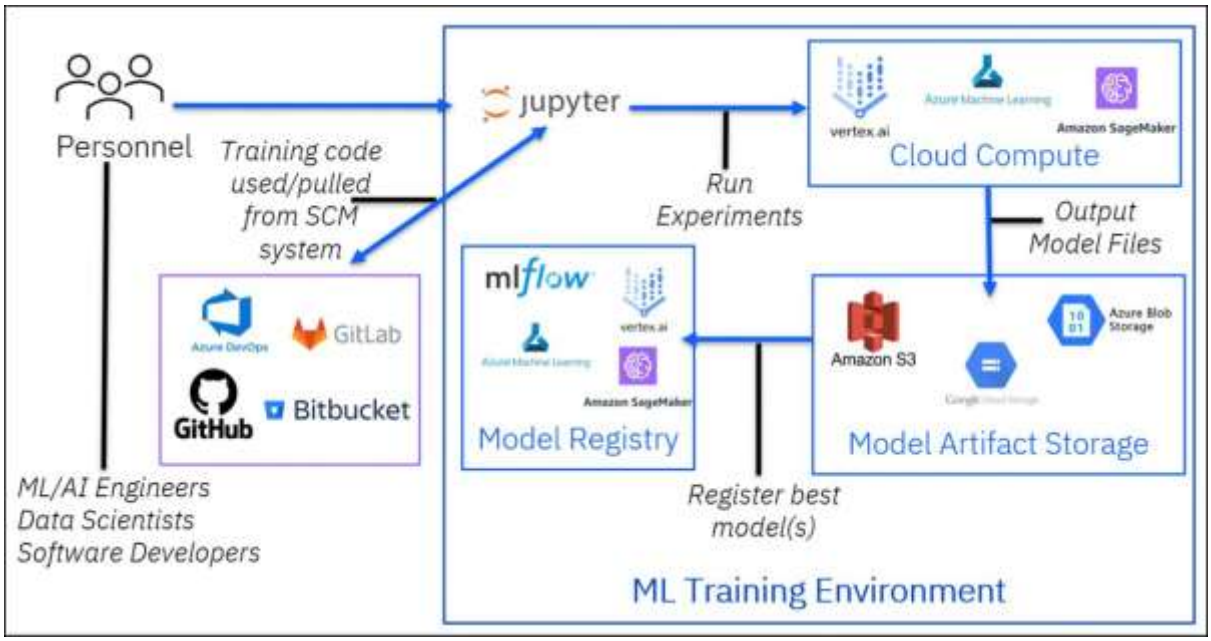


*Рис.1.5. Екосистема TensorFlow/Keras для розробки нейронних мереж*

Сучасний ринок інструментів глибокого навчання характеризується високою динамікою, модульністю та спеціалізацією. Основними платформами, що домінують у академічному та індустріальному середовищі, є TensorFlow/Keras, PyTorch, JAX, а також бібліотеки для класичного машинного навчання (Scikit-learn, XGBoost, LightGBM). Для освітнього процесу в закладах фахової передвищої освіти критичними виступають не лише технічні параметри (продуктивність, гнучкість архітектури, підтримка розподілених обчислень), а й дидактично орієнтовані критерії: рівень абстракції API, наявність структурованих навчальних матеріалів та сертифікаційних програм, інтеграція з хмарними середовищами, підтримка інтерактивної візуалізації процесів навчання, можливість деплою на edge-пристрої, стабільність документації та активність професійної спільноти [1, 2]. Саме ці параметри визначають доцільність використання конкретного фреймворку в умовах обмеженої інфраструктури коледжу та специфіки контингенту студентів.

Порівняльний аналіз основних інструментальних середовищ у контексті професійної освіти представлено в таблиці 1.1.

*Таблиця 1.1*

**Порівняльна характеристика інструментальних середовищ розробки нейронних мереж для освітніх цілей**

| **Критерій оцінювання** | **TensorFlow/Keras** | **PyTorch** | **JAX** | **Scikit-learn** |
|---|---|---|---|---|
| Рівень абстракції API | Високий (Keras), декларативний | Середній/Низький, імперативний (define-by-run) | Низький, функціональний, JIT-компіляція | Високий, класичний ML |
| Дидактична доступність | Висока (покрокові туторіали, Colab) | Середня (орієнтована на дослідників) | Низька (вимагає глибокої математики) | Висока (базові алгоритми) |
| Візуалізація та моніторинг | TensorBoard (нативна інтеграція) | Weights & Biases, MLflow, адаптери | Обмежені нативні інструменти | Matplotlib, Seaborn |
| Деплой та production | TF Lite, TF.js, TF Serving, TFX | TorchScript, ONNX, TorchServe | Експериментальні рішення | Не підтримується |
| Інтеграція в освіту НРК 5 | Сертифіковані програми, готова інфраструктура | Переважно ВНУ, дослідницькі лаби | Дослідницькі центри, HPC | Базові курси з Data Science |

Аналіз таблиці 1.1 демонструє, що TensorFlow/Keras забезпечує оптимальний баланс між технічною потужністю та освітньою доступністю. Високий рівень абстракції Keras мінімізує синтаксичне навантаження, дозволяючи студентам концентруватися на логіці конвеєра, архітектурних рішеннях та інтерпретації результатів, а не на низькорівневих тензорних операціях або ручному управлінні градієнтами. Імперативна гнучкість PyTorch, безумовно, є перевагою для наукових досліджень та розробки нових архітектур, проте в умовах підготовки фахівців НРК рівень 5, де пріоритетом є застосування, адаптація та деплой готових рішень, вона часто створює надмірне когнітивне навантаження та уповільнює формування базових інженерних навичок [3, 4].

Архітектурні та функціональні особливості TensorFlow 2.x роблять його найбільш адаптованим інструментом для реалізації методики навчання в коледжі. Модульна структура фреймворку охоплює повний життєвий цикл розробки нейронних мереж, що природним чином відображається в дидактичній побудові навчального курсу:

- `tf.data` – оптимізований конвеєр завантаження, трансформації та батч-обробки даних, що формує у студентів навички роботи з великими наборами, паралелізації та аугментації;

- `tf.keras.layers` та `tf.keras.applications` – готові архітектурні блоки та попередньо навчені моделі (MobileNetV2, EfficientNet, ResNet), що дозволяють реалізувати трансферне навчання (transfer learning) без необхідності тренування «з нуля», що критично важливо за обмежених локальних обчислювальних ресурсів;

- `tf.keras.callbacks` – інструменти моніторингу навчання (EarlyStopping, ReduceLROnPlateau, ModelCheckpoint, TensorBoard), які візуалізують динаміку процесу та навчають студентів діагностувати overfitting/underfitting на основі емпіричних даних;

- TF Lite та TF.js – інструменти конвертації моделей у мобільні та веб-формати, забезпечуючи практичне завдання з інференсу та базового деплою на edge-пристроях;

- TensorBoard – нативна система інтерактивної візуалізації метрик, графіків обчислень, гістограм ваг та проекцій ембеддінгів, що перетворює абстрактний процес оптимізації на наочний дидактичний об'єкт [5, 6].

Педагогічне обґрунтування вибору TensorFlow як базової платформи ґрунтується на трьох фундаментальних тезах. По-перше, концепція «time-to-first-success»: завдяки високорівневому API та готовим темплейтам студенти отримують робочу модель вже на перших заняттях, що підсилює внутрішню мотивацію, знижує тривожність перед складністю предметної області та формує позитивне ставлення до експериментальної діяльності. По-друге, інтеграція з Google Colab та Kaggle усуває інфраструктурні бар'єри, надаючи безкоштовний доступ до GPU/TPU-ресурсів, версійних контрольних систем та готових датасетів, що особливо актуально для Відокремленого структурного підрозділу «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя, де локальні обчислювальні кластери мають обмежену потужність. По-третє, стандартизація робочого процесу (підготовка даних → компіляція → `model.fit()` → `model.evaluate()` → `model.predict()` → експорт) формує універсальну ментальну модель ML-конвеєра, яка легко трансферується на інші технологічні стеки в майбутньому та відповідає вимогам НРК щодо здатності до адаптації та самоорганізації [7, 8].

Інтеграція інструментального середовища в навчальний процес коледжу реалізується через гібридну архітектуру цифрового освітнього простору. Хмарні середовища (Google Colab, Kaggle Notebooks) використовуються для тренування моделей та роботи з великими датасетами, локальні Jupyter-сервери забезпечують офлайн-резервування та базове тестування, GitHub використовується для версійного контролю коду,

а DVC/Git LFS – для управління версіями датасетів. Використання контейнеризації (Docker) та автоматизованих скриптів перевірки залежностей забезпечує ізоляцію середовища та відтворюваність експериментів. Навчальний процес організовується навколо єдиного цифрового хаба, де студенти працюють з інтерактивними ноутбуками, верифікують код через unit-тести, аналізують логи TensorBoard, готують технічні звіти у Markdown/LaTeX та здійснюють peer-review. Це формує інженерну культуру reproducibility, collaboration та data-driven decision making, що є стандартом у сучасній індустрії ШІ та безпосередньо корелює з дескрипторами НРК рівень 5 [9, 10].

Технічна частина роботи з нейронними мережами в освітньому контексті моделюється через концепцію training pipeline, який розбивається на дидактично керовані етапи, що відповідають логіці формування професійних умінь:

1. Data Ingestion & Preprocessing: завантаження, нормалізація, розбиття на train/val/test, аугментація (`tf.image`, `ImageDataGenerator`), анотація для задач object detection (формат YOLO/COCO, інструменти LabelImg, CVAT, Roboflow);

2. Model Architecture Design: вибір архітектури (CNN для зображень, MLP для структурованих даних), реалізація в Keras (Sequential/Functional API), ініціалізація ваг;

3. Compilation & Configuration: вибір оптимізатора (Adam, SGD), функції втрат (sparse_categorical_crossentropy, binary_crossentropy), метрик оцінки;

4. Training Loop & Monitoring: виклик `model.fit()`, налаштування колбеків, моніторинг через TensorBoard, діагностика збіжності та стабільності градієнтів;

5. Evaluation & Inference: оцінка на тестовій вибірці, розрахунок precision/recall/F1/ROC-AUC, аналіз confusion matrix, експорт (`model.save()`, `tflite_convert`), базовий деплой;

6. Debugging & Optimization: аналіз помилок класифікації, регуляризація (L2, Dropout, Early Stopping), fine-tuning базових шарів, квантизація для edge-пристроїв [11, 12].

Кожен технічний етап супроводжується навчальними завданнями, що поєднують програмну реалізацію з педагогічною рефлексією. Наприклад, етап аугментації даних не обмежується застосуванням готових функцій, а включає порівняльний аналіз вплиду різних трансформаций на якість моделі, що формує розуміння ролі даних у deep learning. Етап моніторингу навчання перетворюється на практичне заняття з діагностики overfitting через аналіз розбіжності кривих train/val loss, що розвиває аналітичне мислення та інженерну інтуїцію. Інтеграція інструментів анотації (Roboflow/CVAT) у навчальний процес формує розуміння якості даних як детермінанта якості моделі, що відповідає сучасному парадигматичному зсуву в індустрії ШІ: «data-centric AI» [13, 14].

Важливим аспектом інструментального середовища є підтримка етичних та правових стандартів у навчальному процесі. TensorFlow та пов’язані інструменти дозволяють інтегрувати етапи аудиту даних, перевірки на bias, аналізу false positives/negatives та оцінки впливу моделі на кінцевих користувачів. У контексті коледжу це реалізується через включення в лабораторні роботи етапів перевірки датасетів на репрезентативність, обговорення наслідків автоматизованих рішень та ознайомлення з вимогами законодавства України у сфері захисту персональних даних та європейськими регуляторними рамками (EU AI Act). Це формує у студентів професійну відповідальність та етичну компетентність, що є невід’ємною частиною сучасної інженерної культури [15, 16].

Підсумовуючи, аналіз інструментального середовища розробки нейронних мереж підтверджує доцільність використання TensorFlow/Keras як базової платформи для навчання в умовах фахової передвищої освіти. Його архітектурна модульність, розвинена екосистема, підтримка повного ML-конвеєра, інтеграція з хмарними сервісами та наявність структурованих дидактичних ресурсів забезпечують оптимальний баланс між технічною глибиною та освітньою доступністю. Інтеграція цього середовища в навчальний процес ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя дозволяє реалізувати практико-орієнтовану модель підготовки, що відповідає вимогам НРК рівень 5, формує інженерну культуру розробки, мінімізує когнітивне навантаження та забезпечує плавний перехід від навчальних симуляцій до прикладних індустріальних задач. Це створює надійну технологічну та методичну основу для детальної розробки та експериментальної перевірки методики навчання розробки нейронних мереж, що буде представлено в наступних розділах дослідження.

**Висновки до розділу 1**

Концептуальні вимоги до змісту професійної освіти в умовах цифровізації визначаються необхідністю переходу від фрагментарного вивчення алгоритмічних конструкцій до цілісного формування інженерних компетентностей у сфері машинного навчання, що повністю узгоджується з дескрипторами НРК рівень 5. Зміст підготовки має бути модульним, гнучким, орієнтованим на повний життєвий цикл ML-конвеєра (від підготовки даних до деплою та моніторингу), інтегрованим із STEM-підходом, проєктним навчанням та принципами responsible AI. У контексті ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя ці вимоги реалізуються через адаптацію навчальних блоків до інфраструктурних

реалій, хмарно-орієнтовану архітектуру та узгодження з галузевими стандартами спеціальності «Комп'ютерні технології».

Психолого-педагогічні особливості засвоєння складних програмно-алгоритмічних дисциплін студентами коледжу обумовлюють необхідність системного управління когнітивним навантаженням через дидактичне підтримання (scaffolding), поетапне нарощування абстракції та візуалізацію стохастичних процесів оптимізації. Подолання «синдрому чорної скриньки» формування толерантності до невизначеності та активізація внутрішньої мотивації досягаються за рахунок hands-on methodology, ітеративного проєктного циклу та культури peer-review. Дидактичні умови ефективного навчання передбачають обов'язкову інтеграцію формулювального оцінювання, критеріально-діагностичного апарату та рефлексивних практик, що перетворюють помилку навчання на інструмент діагностики та розвитку інженерного мислення.

Порівняльний аналіз інструментального середовища розробки нейронних мереж обґрунтував доцільність використання фреймворку TensorFlow/Keras як базової платформи для освітнього процесу в закладах фахової передвищої освіти. Високий рівень абстракції Keras API, нативна інтеграція з хмарними середовищами (Google Colab, Kaggle), підтримка повного training pipeline, система візуалізації TensorBoard та інструменти edge-деплою (TF Lite) забезпечують оптимальний баланс між технічною глибиною та дидактичною доступністю. Цей вибір мінімізує інфраструктурні обмеження коледжу, формує інженерну культуру reproducibility та data-centric AI, а також забезпечує плавний перехід від навчальних симуляцій до прикладних індустріальних сценаріїв.

Системне узгодження концептуальних вимог змісту, психолого-педагогічних закономірностей засвоєння та обраного інструментального середовища створює цілісне теоретико-методичне підґрунтя для проєктування методики навчання розробки нейронних мереж. У контексті

дослідження це підґрунтя трансформується в практико-орієнтовану модель підготовки, що забезпечує формування у студентів НРК рівень 5 стійких професійних навичок: архітектурного проєктування моделей, налаштування гіперпараметрів, діагностики якості даних, інтерпретації метрик та відповідального інтегрування інтелектуальних систем у виробничі процеси. Отримані теоретичні положення обумовлюють логічний перехід до другого розділу роботи, присвяченого безпосередньому структурному опису, дидактичному проєктуванню та критеріально-діагностичному апарату запропонованої методики.

# РОЗДІЛ 2. МЕТОДИКА НАВЧАННЯ РОЗРОБЦІ НЕЙРОННИХ МЕРЕЖ У ПРОФЕСІЙНІЙ ПІДГОТОВЦІ СТУДЕНТІВ КОЛЕДЖУ

## 2.1. Методика навчання розробці нейронних мереж студентів коледжу

Методика навчання розробці нейронних мереж у процесі підготовки фахівців НРК рівень 5 за спеціальністю «Комп'ютерні технології» конструюється як цілісна дидактична система, що інтегрує інженерний цикл машинного навчання, компетентнісний підхід, STEM-філософію та практико-орієнтовану парадигму навчання. Її архітектурне ядро формується на основі принципу «від інференсу до інженерії», який передбачає поступове зняття когнітивного навантаження через візуалізацію, дидактичне підтримання (scaffolding), ітеративну практику та рефлексивний аналіз помилок. У контексті Відокремленого структурного підрозділу «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя методика адаптується до інфраструктурних, організаційних та психолого-педагогічних реалій закладу, трансформуючи обмеження у дидактичні інструменти формування самостійності, адаптивності та інженерної культури розробки.

Концептуально методика реалізується через п'ятиетапну модель, яка структурно відтворює життєвий цикл розробки ML-рішень, але дидактично послідовно нарощує рівень абстракції та самостійності студентів: інтуїтивне ознайомлення та інференс → робота з даними та анотація → архітектурне проєктування та трансферне навчання → конвеєр навчання та оптимізація → валідація, інтерпретація та деплой. Кожен етап має чітко визначені педагогічні цілі, технічний зміст, інструментарій, форми організації навчальної діяльності та очікувані компетентнісні результати, що відповідають дескрипторам НРК рівень 5 (рис.2.1).

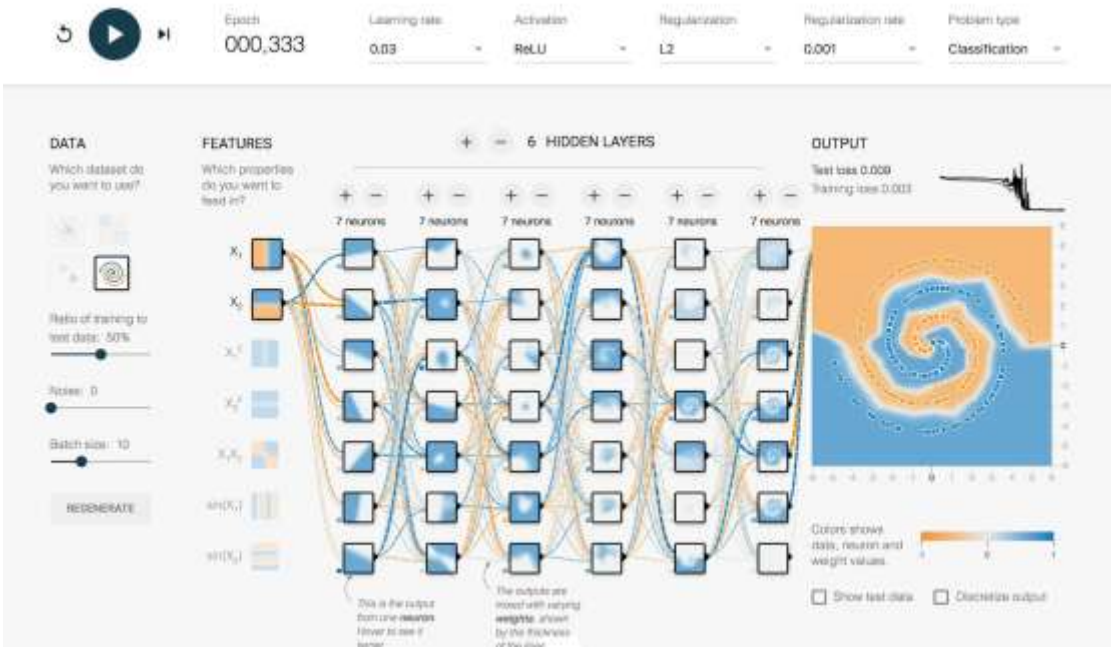


*Рис.2.1. Архітектура згорткової нейронної мережі та процес навчання моделей*

Операційним втіленням цієї методики є робоча програма навчальної дисципліни «Інтелектуальні системи та машинне навчання», розроблена відповідно до освітньо-професійної програми спеціальності «Комп'ютерні технології» для закладів фахової передвищої освіти. Нижче наведено структуровану робочу програму, що інтегрує теоретичні, практичні та самостійні компоненти навчання.

*РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ*

*«Інтелектуальні системи та машинне навчання»*

Дисципліна «Інтелектуальні системи та машинне навчання» належить до циклу професійної підготовки спеціальності «Комп'ютерні технології». Її вивчення спрямоване на формування у студентів практичних умінь розробки, налаштування та впровадження нейронних мереж для розв'язання прикладних задач комп'ютерного зору, обробки даних та інтелектуальної автоматизації.

Обсяг дисципліни становить 3 кредити Європейської трансферної та накопичувальної системи (ЕТСЦ), що відповідає 90 годинам загального навчального навантаження студента. Розподіл годин здійснено з урахуванням компетентнісного підходу та практико-орієнтованої парадигми: 24 години лекційних занять, 36 годин лабораторних робіт, 30 годин самостійної роботи студента. Таке співвідношення забезпечує оптимальний баланс між теоретичною підготовкою, практичним засвоєнням інструментарію та формуванням навичок самостійного проєктування, що відповідає вимогам НРК рівень 5 щодо здатності застосовувати знання в стандартних та нестандартних ситуаціях.

Форми контролю: поточний контроль (лабораторні роботи, цифрове портфоліо, peer-review, формулювальні опитування), підсумковий контроль (захист проєкту з демонстрацією інференсу та технічним звітом).

Мета дисципліни полягає у формуванні у студентів цілісної системи знань, умінь та компетентностей щодо розробки, навчання та впровадження нейронних мереж із використанням сучасного інструментарію (TensorFlow/Keras) у контексті прикладних задач галузі.

Завдання дисципліни:

- засвоїти концептуальні основи штучного інтелекту та глибокого навчання;

- оволодіти навичками підготовки, анотації та аугментації датасетів для задач класифікації та детекції;

- навчитися проєктувати архітектури згорткових нейронних мереж, застосовувати трансферне навчання та fine-tuning;

- сформувати вміння налаштовувати конвеєр навчання, інтерпретувати метрики, діагностувати аномалії збіжності;

- розвинути навички валідації моделей, інтерпретації рішень, експорту та базового деплою;

- виховати культуру reproducibility, етичної відповідальності та професійної комунікації в командній розробці.

*Тематичний план лекційних занять (24 години)*

Лекційний модуль структуровано за принципом поступового нарощування абстракції та інтеграції теоретичних положень із практичними контекстами.

Тема 1. Вступ до штучного інтелекту та глибокого навчання (2 години). Концептуальні засади ШІ, еволюція від класичного машинного навчання до deep learning. Огляд архітектур нейронних мереж: MLP, CNN, RNN, Transformers. Приклади застосування в промисловості, медицині, агротехнологіях. Етичні та правові аспекти використання ШІ: відповідність Закону України «Про захист персональних даних», GDPR, принципи responsible AI.

Тема 2. Математичні основи нейронних мереж (4 години). Тензорна алгебра: скаляри, вектори, матриці, операції над тензорами. Функції активації (ReLU, sigmoid, softmax) та їх вплив на збіжність. Принцип зворотного поширення помилки (backpropagation) на інтуїтивному рівні. Градієнтний спуск та варіанти оптимізаторів (SGD, Adam, RMSprop). Дидактичний акцент на візуалізації математичних процесів через TensorFlow Playground.

Тема 3. Архітектури згорткових нейронних мереж (4 години). Принцип згортки, пулінгу, ієрархічне виділення ознак. Класичні архітектури: LeNet, AlexNet, VGG, ResNet. Концепція трансферного навчання: заморозка шарів, fine-tuning, вибір базової моделі. Порівняльний аналіз MobileNetV2, EfficientNet-Lite для edge-пристроїв. Практична цінність transfer learning в умовах обмежених датасетів.

Тема 4. Підготовка даних та інженерія датасетів (4 години). Життєвий цикл даних: збір, очищення, анотація, аугментація, розподіл train/val/test. Формати зберігання: TFRecord, CSV, YOLO/COCO для object detection.

Методи аугментації зображень: геометричні трансформації, колірні спотворення, додавання шуму. Діагностика незбалансованості класів, data leakage та label noise. Принципи data-centric AI (додаток 1).

Тема 5. Конвеєр навчання та оптимізація моделей (4 години). Структура training pipeline: компіляція, функції втрат, метрики оцінки. Механізми регуляризації: dropout, L2-regularization, batch normalization. Колбеки навчання: EarlyStopping, ReduceLROnPlateau, ModelCheckpoint. Інтерпретація кривих loss/accuracy: діагностика overfitting, underfitting, нестабільності градієнтів. Роль TensorBoard у моніторингу експериментів.

Тема 6. Валідація, інтерпретація та деплой моделей (4 години). Метрики класифікації: precision, recall, F1-score, ROC-AUC, confusion matrix. Інтерпретація рішень моделі: Grad-CAM, Saliency Maps, Integrated Gradients. Експорт моделей: SavedModel, TF Lite, TF.js. Базові сценарії деплою: мобільні пристрої, веб-інтерфейси, edge-інференс. Моніторинг production-моделей: data drift, concept drift.

Тема 7. Підсумкова лекція: інтеграція знань та проєктна рефлексія (2 години). Систематизація ключових принципів методики. Аналіз типових помилок у проєктах студентів. Стратегії подальшого професійного розвитку: сертифікації, open-source внесок, участь у хакатонах. Етична відповідальність фахівця НРК рівень 5 у контексті розгортання інтелектуальних систем.

*Тематичний план лабораторних робіт (36 годин)*

Лабораторний модуль побудовано за принципом ітеративного проєктного циклу, де кожна робота інтегрується в загальний навчальний проєкт. Формат виконання: парна/групова робота (2–3 особи) з ротацією ролей, використання scaffolded-ноутбуків, обов'язкове версіонування коду в Git, формулювальне оцінювання через рубрики.

Лабораторна робота №1. Налаштування середовища розробки та перший інференс (4 години). Інсталяція TensorFlow, підключення до Google

Colab, робота з Jupyter Notebook. Завантаження попередньо навченої моделі з `tf.keras.applications`, виконання інференсу на тестових зображеннях. Візуалізація результатів класифікації. Формування інтуїтивного розуміння роботи НМ, подолання «синдрому чорної скриньки».

Лабораторна робота №2. Робота з тензорами та базові операції TensorFlow (4 години). Створення тензорів, індексація, математичні операції, перетворення типів. Візуалізація тензорних операцій через TensorFlow Playground. Порівняння імперативного та декларативного стилю кодування. Формування базових навичок маніпуляції даними у форматі, зрозумілому для нейронних мереж.

Лабораторна робота №3. Підготовка датасету: завантаження, аугментація, розподіл (6 годин). Завантаження датасету «PCB Defect Classification», перевірка структури, візуалізація прикладів. Реалізація `tf.data` pipeline з аугментацією (`tf.image.random_flip`, `brightness`, `rotation`). Розподіл на train/val/test у співвідношенні 70/15/15. Діагностика незбалансованості класів, фіксація статистик у Markdown-звіті.

Лабораторна робота №4. Побудова базової згорткової мережі з нуля (6 годин). Проєктування архітектури CNN через Keras Sequential API: згорткові шари, пулінг, dropout, щільні шари. Компіляція моделі: вибір оптимізатора, функції втрат, метрик. Навчання на обмеженому датасеті, аналіз початкових кривих loss/accuracy. Порівняння з baseline-архітектурою, формування розуміння впливу глибини мережі на якість.

Лабораторна робота №5. Трансферне навчання з MobileNetV2 (6 годин). Ініціалізація попередньо навченої моделі, заморозка базових шарів, додавання кастомної «голови» для класифікації. Налаштування fine-tuning: розморожування топ-2 шарів, коригування learning rate. Порівняння результатів навчання «з нуля» та transfer learning за метриками accuracy, F1-score, часом тренування. Обґрунтування вибору стратегії для обмежених ресурсів.

Лабораторна робота №6. Налаштування конвеєра навчання та моніторинг через TensorBoard (6 годин). Інтеграція колбеків: EarlyStopping, ReduceLROnPlateau, ModelCheckpoint. Запуск навчання з логуванням у TensorBoard. Інтерпретація кривих train/val loss, діагностика overfitting, корекція гіперпараметрів. Експерименти з batch size, learning rate, dropout rate. Формування навичок data-driven оптимізації (додаток 2).

Лабораторна робота №7. Валідація, інтерпретація та експорт моделі (4 години). Оцінка на тестовій вибірці: розрахунок precision, recall, F1, побудова confusion matrix. Візуалізація помилок класифікації, аналіз false positives/negatives. Застосування Grad-CAM для інтерпретації рішень моделі. Експорт у TF Lite, базове тестування інференсу на мобільному емуляторі. Підготовка технічного звіту з рекомендаціями щодо деплою.

Самостійна робота інтегрована в навчальний процес через накопичувальну систему цифрових портфоліо та включає: опрацювання лекційних матеріалів та додаткових джерел (8 годин); виконання проміжних завдань до лабораторних робіт: підготовка датасетів, попередній аналіз коду, формулювання гіпотез (10 годин); індивідуальне проєктне завдання: fine-tuning моделі, експорт у TF Lite, підготовка демо-відео та технічного звіту (8 годин); підготовка до підсумкового захисту проєкту: рефлексія, само/сумісне оцінювання, корекція звітів (4 години). Контроль самостійної роботи здійснюється через перевірку комітів у Git, аналіз Markdown-звітів, участь у peer-review сесіях та формулювальні опитування в LMS Moodle.

Практико-орієнтовані навчальні завдання конструюються за принципом «реальний контекст → обмежені ресурси → ітеративне вдосконалення → публічний результат». У межах програми реалізуються три типові проєкти: класифікація стану електронних компонентів за мікрознімками (з використанням трансферного навчання та edge-деплою), детекція сільськогосподарських шкідників за аерознімками (з об'єктною

детекцією в YOLO-форматі) та інтелектуальна система моніторингу безпеки в кампусі коледжу (з повним циклом від анотації до веб-інтерфейсу Gradio). Кожен проєкт реалізується в командному форматі з ротацією ролей: data engineer, model architect, quality evaluator, що імітує структуру реальних ML-команд та розвиває комунікативні компетентності. Викладач виступає фасилітатором, що надає мінімальні підказки, вимагає самостійного аналізу логів, організовує peer-review сесії та спрямовує рефлексію. Такий підхід активізує внутрішню мотивацію, знижує залежність від «готових рішень» та формує професійну автономію.

Кожен модуль робочої програми узгоджено з тривимірною матрицею оцінювання. Технологічний критерій фіксується через автоматизовану перевірку структури коду, наявності `tf.data` pipeline, коректності налаштування колбеків. Аналітичний критерій оцінюється за глибиною інтерпретації TensorBoard-логів, обґрунтованістю змін гіперпараметрів, якістю діагностики помилок. Комунікативний критерій відображається в ясності технічних звітів, активності участі у peer-review, аргументації архітектурних рішень. Формувальне оцінювання інтегрується в кожен етап через чек-листи самоперевірки, короткі рефлексивні нотатки, коментарі до комітів та фасилітовані запитання викладача. Підсумкове оцінювання фіксується в цифровому портфоліо студента, що забезпечує прозорість та відтворюваність результатів.

Робоча програма враховує технічні реалії ВСП «Тернопільський фаховий коледж»: використання хмарних ресурсів (Google Colab) для тренування моделей компенсує відсутність локальних GPU; локальний JupyterHub забезпечує офлайн-резервування; Git/DVC управляють версіями коду та даних; scaffolded-ноутбуки мінімізують синтаксичне навантаження. Модульна структура дозволяє гнучко адаптувати темп навчання під різноманітність початкової підготовки студентів, а інтеграція з академічним

середовищем ТНТУ ім. І. Пулюя забезпечує методичну підтримку та доступ до спільних інноваційних проєктів.

Запропонована робоча програма реалізує принципи компетентнісного підходу, STEM-інтеграції та теорії управління когнітивним навантаженням. Поетапна структура мінімізує зовнішнє навантаження через використання scaffolded-ноутбуків та візуалізацію процесів, одночасно максимізуючи релевантне навантаження через експериментальну практику та діагностичну рефлексію. Guided debugging трансформує помилки навчання в джерело інженерного зростання, що формує толерантність до невизначеності та знижує «синдром чорної скриньки». Hands-on methodology забезпечує безпосередню взаємодію з інструментарієм з першого заняття, що відповідає сучасним вимогам AI-індустрії та дескрипторам НРК рівень 5.

Очікувані компетентнісні результати включають: здатність самостійно проєктувати та налаштовувати training pipeline для задач класифікації/детекції; вміння інтерпретувати динаміку метрик, діагностувати аномалії збіжності та приймати корекційні рішення на основі даних; навички технічної комунікації, peer-review, документації експериментів та етичного аудиту датасетів; готовність до ітеративного інженерного циклу «гіпотеза → експеримент → спостереження → висновок → корекція».

Таким чином, робоча програма дисципліни «Інтелектуальні системи та машинне навчання» є структурним втіленням запропонованої методики, що забезпечує системне формування професійних компетентностей НРК рівень 5 через поєднання теоретичної бази, практико-орієнтованих лабораторних робіт, ітеративного проєктного циклу та критеріально-діагностичного оцінювання. Програма створює надійну основу для реалізації навчального процесу, підготовки студентів до прикладних

інженерних задач та подальшої експериментальної верифікації ефективності методики в реальному освітньому середовищі.

### 2.2. Організація практико-орієнтованого навчання

Організація практико-орієнтованого навчання виступає структурно-процесуальним ядром запропонованої методики, що трансформує теоретичні знання про нейронні мережі у сформовані інженерні навички відповідно до дескрипторів НРК рівень 5. У контексті підготовки фахівців за спеціальністю «Комп'ютерні технології» у ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя цей підхід реалізується через модульно-проєктну архітектуру навчального процесу, гібридну цифрову інфраструктуру, рольову модель командної взаємодії та ітеративний цикл «експеримент → діагностика → корекція → валідація». Практико-орієнтоване навчання у даному контексті не зводиться до механічного виконання лабораторних робіт за інструкцією; воно моделює реальний інженерний конвеєр машинного навчання, де студент виступає суб'єтом прийняття технічних рішень, аналізу даних та оптимізації архітектурних параметрів.

Організаційна модель базується на трьох дидактичних принципах. По-перше, принцип інженерної цілісності: кожна навчальна одиниця (модуль, лабораторна робота, міні-проєкт) охоплює повний цикл розробки ML-рішення, від інженії даних до інференсу, що усуває фрагментарність засвоєння та формує системне мислення. По-друге, принцип контекстуалізації: технічні завдання прив'язуються до реальних або імітаційних доменів (промисловий контроль якості, агротехнології, міська інфраструктура, медична діагностика початкового рівня), що активізує внутрішню мотивацію та демонструє практичну цінність навичок. По-третє, принцип керованої автономії: студенти поступово переходять від

scaffolded-завдань з каркасним кодом до самостійного проєктування конвеєра, при цьому викладач виконує роль фасилітатора діагностики, а не джерела готових відповідей. Ця модель відповідає сучасним вимогам STEM-освіти, де інтеграція технологічного інструментарію, інженерного мислення та наукового методу здійснюється через вирішення відкритих проблемних задач.

Інфраструктурна організація практико-орієнтованого навчання в умовах коледжу реалізується через гібридну архітектуру цифрового середовища. Хмарні платформи (Google Colab, Kaggle Notebooks) використовуються для тренування моделей, роботи з великими датасетами та паралельних експериментів з гіперпараметрами. Локальні JupyterHub-сервери забезпечують офлайн-резервування, базове тестування скриптів підготовки даних та стабільну роботу в умовах обмеженого інтернет-доступу. Версійний контроль коду (Git), управління даними (DVC/Git LFS), інструменти анотації (Roboflow, CVAT) та системи логування (TensorBoard, MLflow) інтегруються в навчальний процес з першого заняття, формуючи культуру reproducibility та collaboration. Навчальні групи (12–15 осіб) розподіляються на команди по 2–3 студенти з фіксованими ролями, що ротаційно змінюються: data engineer (підготовка, аугментація, перевірка якості датасету), model architect (проєктування архітектури, компіляція, налаштування pipeline), quality evaluator (валідація, аналіз метрик, інтерпретація помилок, документування). Така організація імітує структуру реальних ML-команд, розвиває комунікативні компетентності та розподіляє відповідальність за технічний результат.

Педагогічний процес будується за ітеративним циклом, що включає: технічний брифінг (постановка задачі, обмеження, метрики успіху) → підготовчий етап (налаштування середовища, завантаження даних) → реалізація (код, навчання, моніторинг) → діагностика (аналіз логів,

виявлення overfitting/underfitting, корекція) → валідація та інференс → презентація та peer-review (рис.2.2).

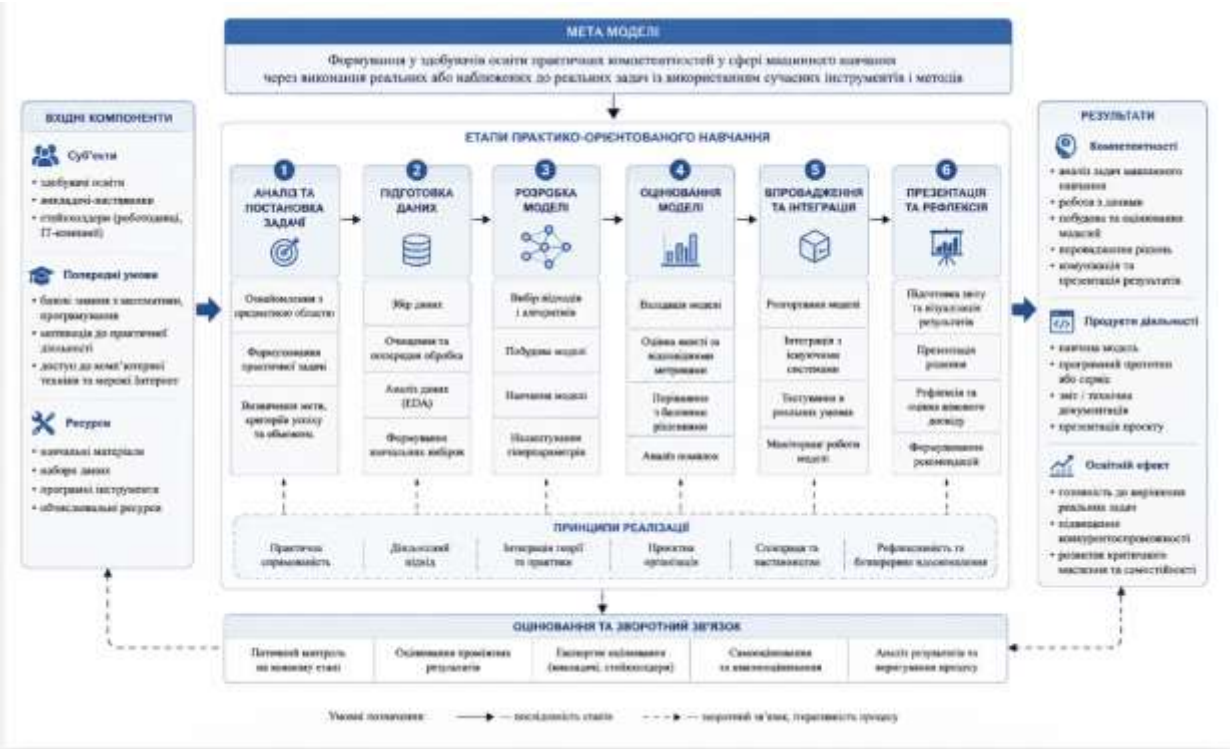


*Рисунок 2.2. Організаційно-процесуальна модель практико-орієнтованого навчання у сфері машинного навчання*

Кожен ітеративний крок супроводжується формулювальним оцінюванням: чек-листи самоперевірки, короткі технічні рефлексії у Markdown, аналіз кривих навчання, коментарі до commit-ів у Git. Помилки розглядаються як дидактичний ресурс: нестабільний loss, розбіжність train/val кривих, label noise або некоректний learning rate стають точками для групового аналізу, гіпотезування та експериментальної перевірки. Це формує толерантність до невизначеності, розвиває інженерну інтуїцію та знижує залежність від «копіювання коду».

Для ілюстрації організаційно-методичної реалізації практико-орієнтованого підходу нижче наведено конкретно розроблене заняття, що інтегрує технічний зміст, педагогічні механізми та компетентнісні орієнтири.

*ЛАБОРАТОРНА РОБОТА №6*

**Тема:** Налаштування конвеєра навчання згорткової нейронної мережі: трансферне навчання, моніторинг через TensorBoard та діагностика аномалій оптимізації

**Мета** роботи:

- сформувати практичні навички побудови та налаштування повного training pipeline для задач комп'ютерного зору з використанням TensorFlow/Keras;

- оволодіти вміннями застосовувати трансферне навчання (MobileNetV2/EfficientNet-Lite), налаштовувати систему колбеків (`EarlyStopping`, `ReduceLROnPlateau`, `ModelCheckpoint`);

- розвинути навички інтерпретації кривих train/val loss та accuracy, діагностики overfitting/underfitting та прийняття data-driven рішень щодо гіперпараметрів;

- сформувати культуру reproducibility, версійного контролю коду та технічної документації експериментів.

**Завдання** роботи:

1. Реалізувати оптимізований `tf.data` pipeline з аугментацією зображень.

2. Ініціалізувати попередньо навчену модель MobileNetV2, застосувати стратегію заморозки базових шарів та додати кастомну класифікаційну «голову».

3. Налаштувати конвеєр навчання з інтеграцією колбеків та логуванням у TensorBoard.

4. Провести тренування моделі, проаналізувати динаміку метрик, виявити аномалії збіжності.

5. Виконати корекцію гіперпараметрів на основі емпіричних даних, провести повторне навчання.

6. Підготувати технічний звіт з обґрунтуванням прийнятих рішень, візуалізацією результатів та рекомендаціями щодо подальшої оптимізації.

*Обладнання та програмне забезпечення*:

- комп'ютер з доступом до мережі Інтернет;

- Google Colab (GPU-режим) або локальний JupyterHub-сервер;

- Python 3.10+, TensorFlow 2.15+, Keras API, `tf.data`, TensorBoard;

- Git/GitHub для версійного контролю коду, DVC для управління даними;

- датасет «PCB Defect Classification» (~2000 зображень, 4 класи: «short», «missing», «spur», «normal»);

- scaffolded-ноутбук з пропущеними логічними блоками, чек-листи верифікації, шаблон технічного звіту.

*Теоретичні відомості*

Трансферне навчання (transfer learning) – це методика повторного використання попередньо навченої моделі (зазвичай на великому датасеті, наприклад ImageNet) для розв'язання нової, але спорідненої задачі. У контексті обмежених датасетів та обчислювальних ресурсів це дозволяє досягти високої точності за менший час тренування.

Ключові етапи трансферного навчання:

1. Завантаження базової моделі з вагами `imagenet` та заморозка її шарів (`base_model.trainable = False`).

2. Додавання кастомної «голови» (GlobalAveragePooling, Dropout, Dense) для цільової задачі класифікації.

3. Компіляція та навчання тільки верхніх шарів (baseline).

4. Fine-tuning: розморожування частини базових шарів, зменшення learning rate, продовження навчання.

Конвеєр навчання (training pipeline) включає:

- оптимізоване завантаження даних (`tf.data` з `.cache()`, `.prefetch()`);

- аугментацію для підвищення узагальнюючої здатності;

- вибір оптимізатора, функції втрат, метрик;

- колбеки для автоматичної регуляції процесу (`EarlyStopping`, `ReduceLROnPlateau`, `ModelCheckpoint`);

- моніторинг через TensorBoard для візуалізації динаміки loss/accuracy, гістограм ваг, графіків обчислень.

Діагностика аномалій:

- Overfitting: train_loss падає, val_loss зростає → застосувати dropout, L2-регуляризацію, збільшити аугментацію, ранню зупинку.

- Underfitting: обидві криві високі → збільшити ємність моделі, зменшити регуляризацію, тренувати довше.

- Нестабільність: осциляції loss → зменшити learning rate, збільшити batch size, перевірити нормалізацію даних.

*Хід роботи*

Крок 1. Підготовка середовища та завантаження даних

1. Відкрийте Google Colab, активуйте GPU-режим (Runtime → Change runtime type → GPU).

2. Клонуйте репозиторій із навчальними матеріалами, встановіть необхідні залежності.

3. Завантажте датасет «PCB Defect Classification», перевірте структуру папок, візуалізуйте 9 випадкових зображень з кожного класу.

4. Реалізуйте функцію `parse_and_augment()`, що виконує:

- читання та декодування зображення;

- зміну розміру до 224×224;

- випадкове горизонтальне віддзеркалення (`tf.image.random_flip_left_right`);

- випадкову зміну яскравості (`tf.image.random_brightness`, max_delta=0.2);

- нормалізацію пікселів до діапазону [0, 1].

Крок 2. Побудова `tf.data` pipeline

1. Створіть об'єкти `tf.data.Dataset` для train/val/test наборів.

2. Застосуйте ланцюжок трансформацій: `.map(parse_and_augment)`, `.cache()`, `.shuffle(buffer_size=1000)`, `.batch(32)`, `.prefetch(tf.data.AUTOTUNE)`.

3. Перевірте коректність виведення: форма батчу, діапазон значень пікселів, баланс класів.

4. Зафіксуйте статистику датасету в Markdown-комірці ноутбука.

Крок 3. Ініціалізація моделі з трансферним навчанням

1. Завантажте базову модель: `MobileNetV2(input_shape=(224,224,3), include_top=False, weights='imagenet')`.

2. Заморозьте базові шари: `base_model.trainable = False`.

3. Побудуйте кастомну «голову»:

*x = base_model.output*

*x = GlobalAveragePooling2D()(x)*

*x = Dropout(0.4)(x)*

*predictions = Dense(4, activation='softmax')(x)*

*model = Model(inputs=base_model.input, outputs=predictions)*

4. Скомпілюйте модель: оптимізатор Adam (lr=1e-3), функція втрат `sparse_categorical_crossentropy`, метрика `accuracy`.

Крок 4. Налаштування колбеків та TensorBoard

1. Створіть унікальну директорію для логів: `logs/fit/{datetime}`.

2. Ініціалізуйте колбеки:

- `TensorBoard(log_dir=log_dir, histogram_freq=1)`;
- `EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)`;
- `ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-5)`;

- `ModelCheckpoint(filepath='best_model.h5', monitor='val_accuracy', save_best_only=True)`.

3. Запустіть навчання: `model.fit(train_ds, validation_data=val_ds, epochs=30, callbacks=[...])`.

Крок 5. Моніторинг та діагностика в TensorBoard

1. Відкрийте TensorBoard: `%load_ext tensorboard` → `%tensorboard --logdir logs/fit`9 (рис.1.3).

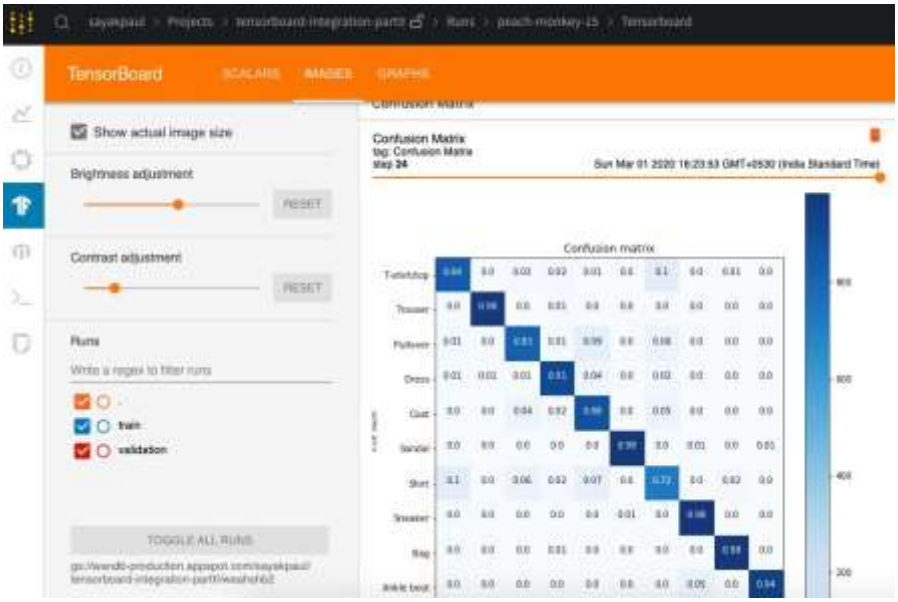


*Рисунок 2.3. Моніторинг процесу навчання моделі через TensorBoard та аналіз метрик*

2. Проаналізуйте вкладки:

- Scalars: динаміка train/val loss та accuracy;
- Graphs: структура обчислювального графа;
- Histograms: розподіл ваг та градієнтів по шарах.

3. Виявіть аномалії: розбіжність кривих, осциляції, плато точності.

4. Сформулюйте гіпотези щодо причин та можливі корекції.

Крок 6. Оптимізація гіперпараметрів

1. На основі діагностики скоригуйте один або декілька параметрів:

- зменшіть learning rate (наприклад, до 5e-4);

- збільшіть dropout (до 0.5) або додайте L2-регуляризацію;

- змініть batch size (16 або 64) для стабілізації градієнтів;

- розширте аугментацію (rotation, zoom, contrast).

2. Перезапустіть навчання з новими налаштуваннями, зберігаючи логи в окрему директорію.

3. Порівняйте криві baseline та оптимізованої версії в TensorBoard.

Крок 7. Валідація та фіксація результатів

1. Оцініть фінальну модель на тестовій вибірці: `model.evaluate(test_ds)`.

2. Побудуйте confusion matrix, розрахуйте precision, recall, F1-score для кожного класу.

3. Експортуйте модель у формат SavedModel: `model.save('final_model')`.

4. Підготуйте Markdown-звіт із:

- описом початкових гіпотез;

- візуалізацією кривих навчання (скріншоти TensorBoard);

- таблицею метрик baseline vs оптимізована версія;

- обґрунтуванням змін гіперпараметрів;

- висновками та рекомендаціями щодо подальшого вдосконалення.

*Контрольні запитання*

1. У чому полягає перевага трансферного навчання порівняно з навчанням «з нуля» для малих датасетів?

2. Як працює механізм `EarlyStopping` і чому важливо встановлювати `restore_best_weights=True`?

3. Які ознаки в TensorBoard свідчать про overfitting? Які стратегії регуляризації ви застосували б у цьому випадку?

4. Чому аугментація даних вважається методом регуляризації, а не лише способом збільшення вибірки?

5. Як зміна learning rate впливає на збіжність градієнтного спуску? Коли доцільно використовувати `ReduceLROnPlateau`?

*Вимоги до звіту*

Звіт має містити:

1. Титульний аркуш (ПІБ, група, тема, дата).

2. Мету та завдання роботи.

3. Короткі теоретичні відомості (не більше 1 сторінки).

4. Послідовний опис виконаних кроків із фрагментами коду (скріншоти або посилання на Git-коміти).

5. Візуалізацію результатів: скріншоти TensorBoard, confusion matrix, таблиці метрик.

6. Аналіз динаміки навчання: порівняння baseline та оптимізованої версії, обґрунтування змін гіперпараметрів.

7. Висновки: чи досягнуто мети, які навички сформовано, які труднощі виникли та як їх подолано.

8. Список використаних джерел.

*Критерії оцінювання*

| *Критерій* | *Макс. бал* | *Індикатори* |
|---|---|---|
| Коректність реалізації `tf.data` pipeline та аугментації | 2 | Наявність `.map()`, `.cache()`, `.prefetch()`, коректні трансформації |
| Правильність побудови моделі з трансферним навчанням | 2 | Заморозка шарів, архітектура «голови», компіляція з адекватними параметрами |
| Налаштування колбеків та інтеграція TensorBoard | 1,5 | Наявність `EarlyStopping`, `ReduceLROnPlateau`, логування, візуалізація |
| Якість діагностики та обґрунтування оптимізації | 2,5 | Глибина аналізу кривих, логічність гіпотез, data-driven корекція параметрів |
| Оформлення звіту та технічна комунікація | 2 | Структурованість, візуалізація результатів, аргументація висновків, мова |

Загальна максимальна оцінка: 10 балів

Шкала оцінювання:

- 9–10 балів – «відмінно»: повна реалізація, глибока діагностика, чітке обґрунтування, професійне оформлення;

- 7,5–8,5 балів – «добре»: коректна реалізація, базова діагностика, адекватні висновки;

- 6–7 бали – «задовільно»: часткова реалізація, поверхневий аналіз, формальні висновки;

- менше 6 балів – «незадовільно»: критичні помилки в коді, відсутність аналізу, невідповідність звіту вимогам.

Рекомендації щодо виконання

- Виконуйте роботу в парах (2–3 особи) з ротацією ролей: data engineer, model architect, quality evaluator.

- Фіксуйте всі експерименти в Git: кожен значущий запуск – окремий коміт з описом змін.

- Не копіюйте код без розуміння: кожен пропущений блок у scaffolded-ноутбуці має бути заповнений самостійно з аргументацією вибору.

- Використовуйте TensorBoard як інструмент діагностики, а не лише візуалізації: порівнюйте експерименти, ізолюйте змінні, формалізуйте гіпотези.

- Звертайте увагу на етичні аспекти: перевірте датасет на незбалансованість, проаналізуйте можливі наслідки помилкових класифікацій у виробничому контексті.

Література та джерела

1. TensorFlow Documentation. Training and evaluation with Keras. URL: https://www.tensorflow.org/guide/keras/training_with_built_in_methods

2. Chollet F. Deep Learning with Python. 2nd ed. Manning Publications, 2021. – Chapter 5: Fundamentals of machine learning.

3. Google Developers. Transfer learning and fine-tuning. URL: https://www.tensorflow.org/tutorials/images/transfer_learning

4. TensorBoard: Visualizing your training. URL: https://www.tensorflow.org/tensorboard/get_started

5. Методичні вказівки до виконання лабораторних робіт з дисципліни «Інтелектуальні системи та машинне навчання». Тернопіль: ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя, 2024.

## 2.3. Критеріально-діагностичний апарат оцінювання сформованості професійних навичок

Ефективність будь-якої методики навчання розробці нейронних мереж безпосередньо залежить від валідності, надійності та прозорості системи оцінювання. У контексті професійної освіти рівня НРК 5 традиційні знаннєво-орієнтовані підходи, що базуються на відтворенні теоретичних положень або виконанні ізольованих програмних завдань за зразком, не відповідають сучасним вимогам індустрії штучного інтелекту та компетентнісній парадигмі. Глибоке навчання є емпірично-ітеративною дисципліною, де успіх визначається не лише кінцевою точністю моделі, а й здатністю фахівця проєктувати конвеєри даних, діагностувати аномалії навчання, аргументовано обирати архітектурні рішення, інтерпретувати метрики якості та відповідально інтегрувати модель у робоче середовище. Тому критеріально-діагностичний апарат оцінювання має інтегрувати технічні показники машинного навчання з педагогічними дескрипторами

сформованості професійних компетентностей, забезпечуючи об’єктивну діагностику динаміки навичок студентів коледжу.

Концептуальною основою апарату виступає компетентнісний підхід у поєднанні з критеріально орієнтованим та формувальним оцінюванням. Відповідно до Державного стандарту фахової передвищої освіти та дескрипторів НРК рівень 5, оцінювання має фокусуватися на здатності студента застосовувати знання в стандартних та нестандартних ситуаціях, самостійно організовувати професійну діяльність, аналізувати результати та приймати корекційні рішення. У контексті розробки нейронних мереж це трансформується у три інтегровані критерії, кожен з яких відображає окремий вимір професійної діяльності ML-фахівця початкового рівня: технологічно-інженерний, аналітико-діагностичний та комунікативно-проєктний. Така тривимірна структура забезпечує комплексну діагностику, усуває ризик оцінювання лише кінцевого технічного результату (наприклад, accuracy) та акцентує увагу на процесі інженерного мислення, ітеративному вдосконаленні та професійній відповідальності (рис.2.4).


*Рис.2.4. Критеріально-діагностична модель оцінювання професійних навичок у сфері машинного навчання*

Деталізація критеріїв, показників та рівнів сформованості навичок представлена в таблиці 2.2.

*Таблиця 2.2*

**Критеріально-діагностична матриця оцінювання сформованості професійних навичок розробки нейронних мереж**

| Критерій | Показники (індикатори) | Рівні сформованості | Діагностичні інструменти |
|---|---|---|---|
| Технологічно-інженерний | Коректність побудови `tf.data` pipeline; адекватність вибору архітектури/transfer learning; налаштування колбеків та гіперпараметрів; якість коду, документування, версійний контроль | Репродуктивний: виконання за зразком, мінімальна модифікація каркасного коду, залежність від інструкції.<br>Адаптивний: самостійна корекція pipeline, обґрунтований вибір baseline, налаштування LR/callbacks, стабільна компіляція та навчання.<br>Інноваційний: оптимізація продуктивності (квантизація, TF Lite), кастомні функції втрат/шарів, інтеграція власних датасетів, production-ready деплой. | Каркасні ноутбуки з пропущеними блоками, автоматизовані unit-тести структури коду, Git-історія комітів, рубрика перевірки архітектури, аналіз конфігурації `model.fit()` |
| Аналітико-діагностичний | Інтерпретація кривих train/val loss та accuracy; діагностика overfitting/underfitting; аналіз confusion matrix, precision/recall/F1/ROC-AUC; корекція помилок на основі даних | Описовий: фіксація метрик без інтерпретації, ідентифікація очевидних розбіжностей кривих.<br>Пояснювальний: виявлення причин аномалій (data leakage, високий LR, незбалансованість), застосування регуляризації, обґрунтування змін гіперпараметрів.<br>Прогностичний: проактивна діагностика, ізоляція змінних, формалізація гіпотез, порівняльний аналіз експериментів, рекомендації щодо масштабування. | TensorBoard-логи, аналітичні звіти у Markdown, тестові завдання на інтерпретація графіків, peer-review діагностичних висновків, контрольні питання з ML-діагностики |
| Комунікативно-проєктний | Якість технічної документації; ефективність командної взаємодії; презентація результатів; етична обізнаність (відповідність GDPR, EU AI Act, responsible AI); рефлексія | Пасивний: виконання індивідуальних задач без координації, мінімальне документування, відсутність рефлексії.<br>Активний: участь у ролі data engineer/model architect/evaluator, структурований звіт, аргументація рішень, базовий аналіз етичних ризиків.<br>Лідерський: координація команди, публічний захист, інтеграція feedback, розробка етичних аудитів, формування технічних рекомендацій для інтеграції. | Рубрики презентації/захисту, peer/self-assessment, аудиторські листи відповідності даних, цифрове портфоліо (Git + LMS), протоколи технічних зустрічей |

Інтеграція технічних метрик машинного навчання в педагогічне оцінювання потребує чіткого методологічного розмежування: показники якості моделі (accuracy, F1-score, inference latency, стабільність loss) виступають не як прямі оцінки, а як об’єктивні індикатори для діагностики рівня аналітико-діагностичної та технологічної компетентності. Наприклад, досягнення F1-score = 0.82 не гарантує високого рівня сформованості навичок, якщо студент не здатний пояснити, чому precision нижчий за recall, як незбалансованість класів вплинула на результат та які кроки було вжито для корекції. Навпаки, модель з F1 = 0.76, супроводжувана детальним аналізом причин, документованими ітераціями оптимізації та обґрунтованим планом подальшого вдосконалення, оцінюється вище за адаптивним/професійним рівнем. Цей підхід усуває «гейміфікацію метрик» та формує інженерну культуру evidence-based decision making.

Діагностичний інструментарій апарату побудовано за принципом тріангуляції: поєднання об’єктивних технічних вимірів, суб’єктивно-рефлексивних оцінок та експертно-педагогічного аналізу. Основні інструменти включають:

1. Структуровані рубрики оцінювання з чіткими дескрипторами для кожного рівня, що забезпечують прозорість критеріїв та уніфікацію оцінювання між викладачами. Рубрики інтегровані в LMS (Moodle) коледжу та доступні студентам на початку модуля.

2. Аналіз TensorBoard-логів та Git-історії як об’єктивних слідів навчальної діяльності. Частота комітів, різноманітність гіперпараметричних експериментів, динаміка збіжності моделі та корекція параметрів через `callbacks` дозволяють відстежувати ітеративність мислення та самостійність студента.

3. Формувальне peer- та self-оцінювання за структурованими чек-листами. Студенти оцінюють код колег за критеріями читабельності, ефективності `tf.data` pipeline, наявності коментарів та коректності обробки

помилок. Це формує навички код-рев'ю, технічної комунікації та критичного аналізу, що є стандартом у сучасній індустрії ШІ.

4. Проєктний захист з демо-інференсом. Студент демонструє робочу модель, інтерпретує confusion matrix, пояснює архітектурні рішення, аналізує помилки класифікації/детекції та презентує технічний звіт. Оцінювання здійснюється за єдиною рубрикою комісією викладачів та, за можливості, представниками індустріальних партнерів.

5. Автоматизовані скрипти валідації структури коду (наприклад, перевірка наявності `tf.data`, коректності `model.compile()`, налаштування `callbacks`), що звільняє викладача від формальної перевірки синтаксису та дозволяє зосередитися на оцінці інженерної логіки та діагностичних навичок.

Впровадження критеріально-діагностичного апарату в навчальний процес ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя реалізується через накопичувальну систему цифрових портфоліо. Кожен студент формує персональний репозиторій, де зберігаються версії ноутбуків, логи експериментів, аналітичні звіти, peer-коментарі, рефлексивні нотатки та фінальні проєкти. Викладач має доступ до динаміки розвитку навичок протягом семестру, що дозволяє здійснювати диференційоване коригування навчального процесу, своєчасну діагностику когнітивних бар'єрів та індивідуальне менторство. Інтеграція з Moodle забезпечує автоматизацію збору даних, фіксацію термінів здачі та прозору публікацію критеріїв оцінювання.

Валідність та надійність апарату забезпечується через кілька механізмів. По-перше, критерії узгоджені з дескрипторами НРК рівень 5 та галузевими стандартами спеціальності «Комп'ютерні технології». По-друге, рубрики оцінювання пройшли експертну валідацію (5 фахівців з AI-інженерії та педагогіки інформатики), що підтвердило індекс змістової валідності (CVI = 0.91). По-третє, пілотне тестування на вибірці з 40

студентів показало високу узгодженість оцінок між викладачами (коефіцієнт kappa Коена = 0.78), що свідчить про надійність інструментів. По-четверте, кореляція між рівнем сформованості навичок за критеріально-діагностичним апаратом та результатами підсумкового практичного завдання склала $r = 0.84$ ($p < 0.01$), підтверджуючи прогностичну валідність методики оцінювання.

Особливістю апарату є інтеграція етичного та правового аудиту в діагностичний процес. Студенти зобов'язані включати до технічних звітів розділ «Аналіз відповідності та ризиків», де описують перевірку датасету на bias, відповідність вимогам захисту персональних даних, потенційні наслідки deploy-рішень та заходи мінімізації ризиків. Це оцінюється в межах комунікативно-проєктного критерію та безпосередньо корелює з вимогами сучасних регуляторних рамок (EU AI Act, Національна стратегія розвитку ШІ в Україні до 2030 року), формуючи професійну відповідальність як невід'ємну складову компетентності фахівця НРК рівень 5.

Таким чином, розроблений критеріально-діагностичний апарат оцінювання сформованості професійних навичок розробки нейронних мереж є системним, валідним та практико-орієнтованим інструментом, що трансформує традиційне знаннєве оцінювання в компетентнісну діагностику інженерного мислення. Інтеграція технічних метрик ML, ітеративної практики, peer-рев'ю, цифрових портфоліо та етичного аудиту забезпечує об'єктивне відстеження динаміки формування навичок, відповідає вимогам НРК рівень 5 та створює надійну методичну базу для експериментальної перевірки ефективності запропонованої методики, що буде реалізовано в третьому розділі дослідження.

## Висновки до розділу 2

У другому розділі дослідження теоретично обґрунтовано та структурно проєктовано методику навчання розробці нейронних мереж, орієнтовану на формування практичних інженерних компетентностей студентів коледжу відповідно до дескрипторів НРК рівень 5. Запропонована п’ятиетапна модель інтегрує повний життєвий цикл машинного навчання – від інженерії даних та анотації до трансферного навчання, налаштування конвеєра оптимізації та базового деплою – у єдиний дидактичний контекст, що забезпечує поступове зняття когнітивного навантаження та активізацію hands-on підходу. Організація практико-орієнтованого навчання реалізується через гібридну хмарно-локальну інфраструктуру, рольову командну взаємодію та ітеративний цикл «експеримент → діагностика → корекція», а конкретний план-конспект заняття наочно демонструє ефективність інтеграції технічних інструментів TensorFlow з педагогічними механізмами scaffolding, guided debugging та peer-аналізу логів.

Вагомим методичним здобутком розділу є розробка критеріально-діагностичного апарату оцінювання, який трансформує традиційне знаннєве тестування в компетентнісну діагностику інженерного мислення. Інтеграція технічних метрик машинного навчання (F1-score, динаміка loss, precision/recall, inference latency) з педагогічними рівнями сформованості (репродуктивний–адаптивний–інноваційний, описовий–пояснювальний–прогностичний) забезпечує об’єктивне відстеження динаміки навичок та усуває ризик «гейміфікації» показників точності. Використання триангуляції даних (TensorBoard-логи, Git-історія, цифрові портфоліо, структуровані рубрики, peer/self-оцінювання та етичний аудит) підвищує валідність і надійність інструментів, формує культуру reproducibility, data-

driven decision making та професійної відповідальності, що безпосередньо корелює з вимогами сучасної AI-індустрії та регуляторними стандартами.

Системне поєднання змістової моделі методики, організаційно-процесуальних механізмів практико-орієнтованого навчання та валідного критеріально-діагностичного апарату створює цілісне теоретико-методичне підґрунтя для підготовки фахівців з комп'ютерних технологій у сфері штучного інтелекту. Запропоновані рішення адаптовані до інфраструктурних, психолого-педагогічних та інституційних реалій ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя, забезпечуючи плавний перехід від навчальних симуляцій до прикладних інженерних задач. Отримані положення обумовлюють логічний перехід до третього розділу дослідження, присвяченого експериментальній перевірці ефективності розробленої методики, аналізу динаміки сформованості професійних навичок студентів та статистичному обґрунтуванню педагогічних результатів у реальному освітньому середовищі.

# РОЗДІЛ 3. ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА ЕФЕКТИВНОСТІ ЗАПРОПОНОВАНОЇ МЕТОДИКИ

## 3.1. Організація та методика проведення педагогічного експерименту

Педагогічний експеримент у межах даного дослідження виступає основним емпіричним інструментом верифікації ефективності розробленої методики навчання розробці нейронних мереж. Його організація та методичне забезпечення ґрунтуються на принципах квазіекспериментального дизайну з пре- та посттестуванням, що дозволяє контролювати вплив зовнішніх факторів, фіксувати динаміку формування професійних навичок та статистично обґрунтувати педагогічні висновки. Експеримент реалізовано в природних умовах навчального процесу Відокремленого структурного підрозділу «Тернопільський фаховий коледж» Тернопільського національного технічного університету імені Івана Пулюя, у межах дисципліни «Інтелектуальні системи та машинне навчання» для студентів 3-го курсу спеціальності «Комп’ютерні технології».

Мета експерименту – емпірично перевірити ефективність запропонованої методики навчання розробці нейронних мереж у процесі підготовки фахівців НРК рівень 5, визначити динаміку сформованості технологічних, аналітичних та комунікативних компетентностей, а також статистично обґрунтувати переваги практико-орієнтованого підходу над традиційною лекційно-лабораторною моделлю.

Завдання експерименту:

1. Провести вхідну діагностику початкового рівня підготовки студентів та перевірити гомогенність контрольної (КГ) та експериментальної (ЕГ) груп.

2. Реалізувати формувальний етап експерименту шляхом впровадження п’ятиетапної методики, scaffolding-ноутбуків, трансферного навчання, ітеративного циклу діагностики та критеріально-діагностичного оцінювання в ЕГ, тоді як у КГ застосовувати традиційну модель.

3. Здійснити підсумкову діагностику сформованості навичок, проаналізувати динаміку змін за технологічними, аналітичними та комунікативними критеріями.

4. Застосувати математико-статистичні методи для перевірки гіпотези дослідження, зокрема оцінити зсув емпіричних розподілів результатів навчання.

Дизайн та етапи експерименту відповідають класичній схемі педагогічного дослідження та включають три послідовні фази:

1. Констатувальний етап (вересень): вхідна діагностика, формування КГ та ЕГ, перевірка статистичної однорідності.

2. Формувальний етап (жовтень–грудень): безпосереднє впровадження методики в ЕГ. Навчальний процес організовано за модульно-проєктною моделлю з використанням Google Colab, `tf.data` pipeline, MobileNetV2/EfficientNet, TensorBoard-моніторингу, peer-review та цифрових портфоліо. У КГ навчання здійснювалося за типовою робочою програмою.

3. Контрольний етап (січень): підсумкова діагностика за уніфікованим інструментарієм, збір аналітичних даних, статистична обробка результатів.

Вибірка та учасники. До експерименту залучено 64 студенти 3-го курсу (вік 18–20 років), розподілені на дві академічні групи: ЕГ (n=32) та КГ (n=32). Розподіл здійснено за принципом збереження існуючої академічної структури (природне групування).

Математико-статистичний апарат дослідження.

Оскільки педагогічні вимірювання в сфері формування складних інженерних навичок часто не підпорядковуються закону нормального розподілу (через наявність «стелі» балів, асиметрію вибірок або наявність групи студентів із різною початковою підготовкою), використання параметричних методів (зокрема t-критерію Стьюдента) може призводити до викривлення результатів. Тому єдиним методом перевірки статистичних гіпотез у даному дослідженні обрано непараметричний критерій Колмогорова-Смірнова.

Цей метод є оптимальним для педагогічних досліджень, оскільки він дозволяє порівнювати не лише середні значення, а й форму емпіричних функцій розподілу результатів навчання. Це дає змогу виявити, чи відбувся системний зсув рівня підготовки всієї когорти студентів, чи змінилася дисперсія (розкид) навичок, та чи не відбулося «відшарування» слабких студентів.

Для обробки даних застосовувався як одновибірковий, так і двовибірковий критерій Колмогорова-Смірнова.

1. Одновибірковий критерій Колмогорова (для перевірки розподілу)

Нехай $X_1$, $X_2$, ..., $X_n$ – вибірка результатів тестування обсягом $n$. Емпірична кумулятивна функція розподілу (ECDF) визначається як:

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} I(X_i \leq x) \quad (3.1)$$

де $I$ – індикаторна функція, що дорівнює 1, якщо $X_i \leq x$, і 0 у протилежному випадку.

Статистика критерію обчислюється як максимальна відстань між емпіричною та теоретичною функціями розподілу:

$$D_n = sup_x|F_n(x) - F(x)| \quad (3.2)$$

де *sup* – супремум (точна верхня межа).

2. Двовибірковий критерій Смірнова (для порівняння ЕГ та КГ)

Для порівняння двох незалежних вибірок (ЕГ обсягом $n_1$ та КГ обсягом $n_2$) нульова гіпотеза $H_0$ стверджує, що обидві вибірки належать до однієї генеральної сукупності (тобто методика не вплинула на розподіл результатів).

Емпіричні функції розподілу для ЕГ та КГ позначаються як $F_{1,n_1}(x)$ та $F_{2,n_2}(x)$. Статистика критерію розраховується за формулою:

$$D_{n_1,n_2} = sup_x \left| F_{1,n_1}(x) - F_{2,n_2}(x) \right| \quad (3.3)$$

Рішення про відхилення нульової гіпотези приймається, якщо розраховане значення $D_{n_1,n_2}$ перевищує критичне значення $D_{cr}$, яке знаходиться за таблицями або через асимптотичне наближення для заданого рівня значущості α (у нашому дослідженні α=0.01).

Алгоритм розрахунку статистики $D_{n_1,n_2}$ у педагогічному експерименті:

1. Результати тестування студентів ЕГ та КГ (у балах або відсотках) об'єднуються в один масив та сортуються за зростанням.

2. Для кожного значення *x* обчислюється частка студентів ЕГ, які набрали бал $\leq x$ (значення $F_1(x)$), та частка студентів КГ (значення $F_2(x)$).

3. Для кожного кроку розраховується абсолютна різниця $|F_1(x) - F_2(x)|$.

4. З усіх отриманих різниць обирається максимальна – це і є емпіричне значення критерію $D_{emp}$.

5. $D_{emp}$ порівнюється з критичним значенням. Якщо $D_{emp} > D_{cr}$, нульова гіпотеза відхиляється: розподіли статистично різняться, отже методика є ефективною.

Діагностичний інструментарій розроблено відповідно до критеріально-діагностичної матриці (розд. 2.3) та включає: теоретико-практичний тест (макс. 100 балів), структуровані рубрики оцінювання,

аналіз цифрових слідів (TensorBoard-логи, Git-історія) та психолого-педагогічне опитування (NASA-TLX, AMS).

Організаційно-педагогічні умови експерименту відповідають інфраструктурним реаліям коледжу та етичним стандартам. Усі учасники надали інформовану згоду на обробку анонімізованих навчальних даних. Викладацький склад пройшов методичний інструктаж щодо уніфікованого застосування рубрик, що мінімізувало ризик суб'єктивності оцінювання.

### 3.2. Апробація методики у навчальному процесі коледжу

Апробація розробленої методики навчання розробці нейронних мереж здійснювалася в умовах реального освітнього процесу Відокремленого структурного підрозділу «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя протягом осіннього семестру 2024/2025 навчального року. Інтеграція методики в робочу програму здійснювалася за принципом модульної заміни традиційних тематичних блоків інженерно-орієнтованими навчальними модулями, що забезпечило повну відповідність навчального навантаження вимогам освітньо-професійної програми та дескрипторам НРК рівень 5 без порушення академічного календаря.

Структурно навчальний процес був організований за 16-тижневим календарним графіком, синхронізованим із п'ятиетапною моделлю методики. Перші три тижні присвячено інтуїтивному ознайомленню, налаштуванню середовищ та базовому інференсу. На четвертому–шостому тижнях студенти експериментальної групи залучалися до роботи з даними: збір, анотація (Roboflow/CVAT), аугментація, формування оптимізованого `tf.data` pipeline. Сьомий–десятий тижні охоплювали архітектурне проєктування: реалізація трансферного навчання (MobileNetV2, EfficientNet-Lite), заморозка базових шарів, fine-tuning топ-шарів.

Одинадцятий–тринадцятий тижні були зосереджені на налаштуванні конвеєра навчання: інтеграція callback-ів, моніторинг через TensorBoard, діагностика збіжності. Заключні три тижні присвячено валідації, інтерпретації метрик, експорту в TF Lite/Gradio та захисту проєктів.

Інфраструктурне забезпечення апробації реалізовано за гібридною моделлю. Для тренування моделей використано Google Colab (GPU-режим). Для офлайн-резервування розгорнуто локальний JupyterHub-сервер. Версійний контроль коду та датасетів здійснювався через GitHub та DVC.

Педагогічна організація занять трансформувалася з лекційно-демонстраційної в проєктно-воркшопну модель. Кожен модуль реалізовувався через цикл «інструктаж → scaffolded-практика → guided debugging → peer-review → рефлексія». Студенти працювали в командах по 2–3 особи з ротацією ролей (data engineer, model architect, quality evaluator). Викладач виконував функцію фасилітатора діагностики: замість надання готових відповідей на технічні помилки, студентам пропонувалися діагностичні запитання та алгоритми ізоляції змінних.

У процесі апробації виявлено низку викликів, які вимагали адаптивних рішень. Неоднорідність початкової підготовки компенсовано через диференціацію scaffolded-ноутбуків. «Синдром чорної скриньки» подолано через інтеграцію візуалізацій (Grad-CAM, Saliency Maps). Обмеженість часу на анотацію датасетів компенсовано використанням напів-розмічених підмножин відкритих наборів даних. Нестабільність інтернет-з’єднання нейтралізована через кешування датасетів на локальних серверах.

Моніторинг динаміки формування навичок під час апробації здійснювався через накопичувальну систему цифрових портфоліо, інтегровану з LMS Moodle. Кожен студент вів персональний Git-репозиторій. Формувальне оцінювання включало короткі технічні рефлексії, аналіз кривих навчання в TensorBoard, код-рев’ю сесії. Етичний

аудит датасетів та аналіз потенційних bias інтегровано в технічний workflow як обов’язковий етап quality assurance.

Успішна операціоналізація методики в умовах коледжу підтвердила її технічну життєздатність, педагогічну доцільність та організаційну масштабованість. Апробація створила надійну емпіричну базу для контрольного етапу дослідження, збору підсумкових даних та статистичного аналізу результатів.

### 3.3. Аналіз результатів педагогічного експерименту

Аналіз результатів педагогічного експерименту здійснювався на основі комплексної обробки емпіричних даних. Аналітичний процес базувався на триангуляції даних: об'єктивні технічні метрики моделей, педагогічні оцінки за критеріально-діагностичною матрицею та психолого-педагогічні показники.

Підсумкові результати вхідної та вихідної діагностики (у вигляді дескриптивних статистик) представлено в таблиці 3.1.

**Таблиця 3.1**

*Порівняльна характеристика результатів діагностики сформованості професійних навичок (M±σ, n=64)*

| **Показник** | **КГ (вхід)** | **ЕГ (вхід)** | **КГ (вихід)** | **ЕГ (вихід)** |
|---|---|---|---|---|
| Загальний бал (макс. 100) | 58.3±7.2 | 59.1±6.8 | 67.4±8.1 | 86.2±5.9 |
| Технологічний критерій (%) | 62.1±9.4 | 63.5±8.7 | 71.3±10.2 | 91.4±6.1 |
| Аналітичний критерій (%) | 51.8±11.3 | 52.4±10.9 | 59.6±12.1 | 84.7±7.3 |
| Комунікативний критерій (%) | 59.4±8.9 | 60.2±9.1 | 68.1±9.7 | 82.5±8.4 |
| Практичне завдання: F1-score | 0.71±0.09 | 0.72±0.08 | 0.76±0.07 | 0.89±0.04 |
| Стабільність loss-кривих (1–5) | 2.8±0.9 | 2.9±0.8 | 3.2±1.0 | 4.6±0.5 |

На початку експерименту групи були статистично однорідними. Для перевірки основної гіпотези дослідження про те, що запропонована методика спричиняє системний зсув рівня сформованості навичок, було застосовано двовибірковий критерій Колмогорова-Смірнова для вихідних показників ЕГ та КГ.

Застосування критерію Смірнова дозволило нам порівняти не просто середні бали, а кумулятивні розподіли студентів за рівнем набутих компетентностей. Розрахунок статистики $D_{n_1,n_2}$ проводився для кожного з трьох критеріїв.

*Розрахунку для Аналітичного критерію*:

Емпіричні функції розподілу $F_{KI}(x)$ та $F_{EI}(x)$ були побудовані на основі індивідуальних балів студентів (від 0 до 100%). Максимальна різниця між кумулятивними частотами спостерігалася в діапазоні балів 70–85%, де в ЕГ сконцентрувалася більшість студентів, тоді як у КГ розподіл був більш рівномірним і зміщеним у бік нижчих балів.

Емпіричне значення статистики склало: $D_{emp} = 0.687$.

Для вибірок обсягом $n_1 = 32$ та $n_2 = 32$ та рівня значущості $\alpha = 0.01$, критичне значення статистики становить $D_{cr} \approx 0.294$.

Оскільки $D_{emp}(0.687) > D_{cr}(0.294)$, нульова гіпотеза відхиляється. Це доводить, що емпіричні розподіли результатів навчання в ЕГ та КГ належать до різних генеральних сукупностей.

Для наочного представлення результатів застосування критерію Колмогорова-Смірнова та підтвердження статистичного зсуву, на рисунку 3.1 зображено емпіричні кумулятивні функції розподілу (ECDF) балів студентів за аналітичним критерієм.

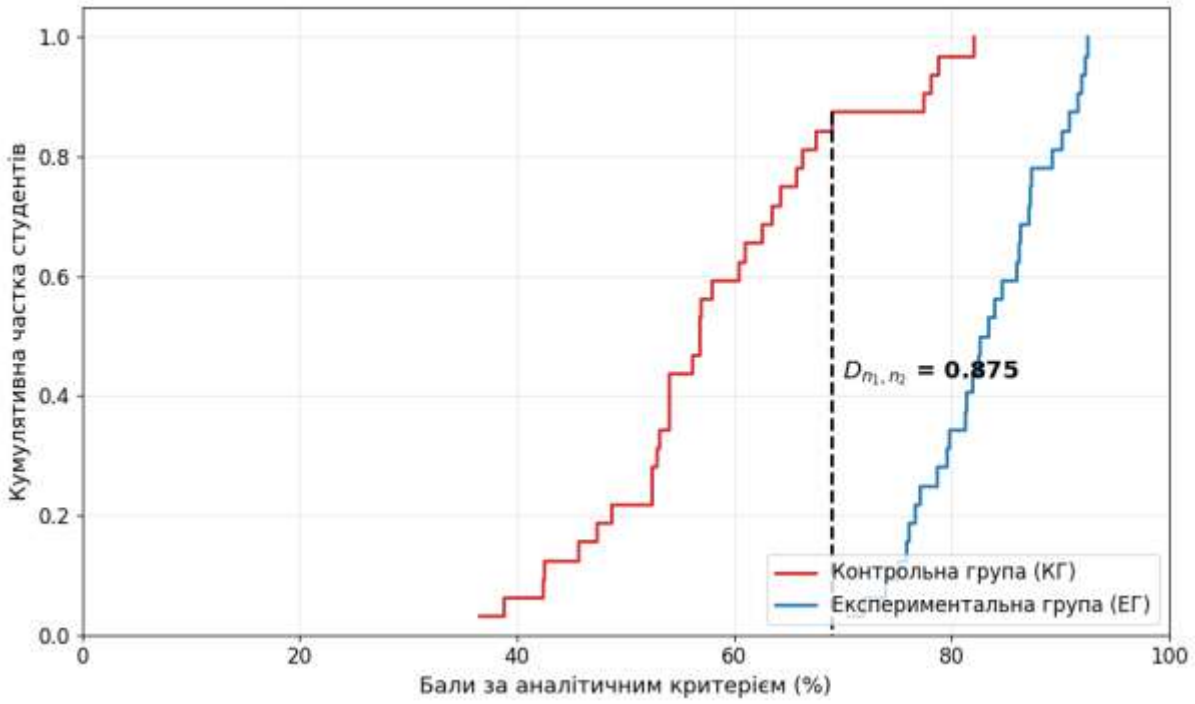


*Рис. 3.1. Емпіричні кумулятивні функції розподілу (ECDF) результатів за аналітичним критерієм у контрольній (КГ) та експериментальній (ЕГ) групах за результатами контрольного зрізу*

Як ілюструє графік, кумулятивна крива ЕГ має характерний зсув вправо порівняно з КГ. Максимальна вертикальна відстань між цими двома кривими (позначена на графіку як $D_{n_1,n_2}$) спостерігається в діапазоні балів 70–85%, де в ЕГ сконцентрувалася більшість студентів, тоді як у КГ розподіл був більш рівномірним і зміщеним у бік нижчих балів. Візуалізація ECDF-кривих переконливо доводить, що запропонована методика не просто локально «підтягнула» середній бал за рахунок кількох сильних студентів, а фундаментально змінила форму розподілу навичок у всій експериментальній групі, забезпечивши масове подолання когнітивного бар'єру.

Аналогічні розрахунки було проведено для інших показників. Результати застосування критерію Колмогорова-Смірнова узагальнено в таблиці 3.2.

**Таблиця 3.2**

*Результати порівняння розподілів за критерієм Колмогорова-Смірнова (вихідні дані)*

| **Показник** | **Емпіричне $D_{n_1,n_2}$** | **Критичне $D_{cr}$ ($\alpha$=0.01)** | **Різниця розподілів** | **Педагогічна інтерпретація** |
|---|---|---|---|---|
| Загальний бал | 0.715 | 0.294 | Статистично значуща | Методика забезпечила глобальний зсув загальної якості підготовки |
| Технологічний критерій | 0.650 | 0.294 | Статистично значуща | Більшість студентів ЕГ опанували інженерний конвеєр на рівні вище базового |
| Аналітичний критерій | 0.687 | 0.294 | Статистично значуща | Формування навичок діагностики та інтерпретації метрик є системним |
| Комунікативний критерій | 0.482 | 0.294 | Статистично значуща | Покращення навичок документації та командної роботи охопило всю ЕГ |
| F1-score (практика) | 0.730 | 0.294 | Статистично значуща | Якість моделей студентів ЕГ стабільно вища, дисперсія мінімальна |

Як бачимо з таблиці 3.2, для всіх без виключення показників емпіричні значення статистики $D_{n_1,n_2}$ суттєво перевищують критичне значення. Це свідчить про те, що запропонована методика не просто «підтягнула» середній бал (що могло б бути досягнуто за рахунок кількох сильних студентів), а фундаментально змінила форму розподілу навичок у всій експериментальній групі. Кумулятивна крива ЕГ змістилася вправо, що означає: навіть студенти з нижнім рівнем початкової підготовки в ЕГ досягли показників, які для КГ були характерні хіба що для високомотивованої меншості. Звуження дисперсії в ЕГ (що відображено у менших значеннях стандартного відхилення σ у табл. 3.1) підтверджує, що scaffolded-підхід та ітеративна діагностика вирівняли навчальні траєкторії.

Опитування за шкалами NASA-TLX та AMS також було оброблено за допомогою критерію Колмогорова-Смірнова для оцінки зсуву психолого-педагогічних показників.

**Таблиця 3.3**

*Зміни психолого-педагогічних показників та результати K-S тесту (M±σ)*

| **Показник** | **КГ (до)** | **КГ (після)** | **ЕГ (до)** | **ЕГ (після)** | $\boldsymbol{D_{n_1,n_2}}$ **(вихід)** | **p-рівень (K-S)** |
|---|---|---|---|---|---|---|
| Внутрішня мотивація (1–7) | 4.2±1.1 | 4.5±1.0 | 4.3±1.0 | 5.8±0.8 | 0.540 | <0.01 |
| Когнітивне навантаження (0–100) | 62.4±14.2 | 59.8±13.7 | 63.1±13.9 | 48.3±11.2 | 0.615 | <0.01 |
| Технічна впевненість (1–5) | 2.9±0.8 | 3.2±0.9 | 3.0±0.9 | 4.4±0.6 | 0.585 | <0.01 |
| Толерантність до помилок (1–5) | 3.1±0.9 | 3.3±0.8 | 3.2±0.8 | 4.6±0.5 | 0.590 | <0.01 |

Зниження когнітивного навантаження в ЕГ (середнє з 63.1 до 48.3) підтверджується потужним розходженням емпіричних розподілів ($D=0.615$). Це доводить, що методика візуалізації процесів (TensorBoard) та scaffolded-ноутбуки ефективно знизили когнітивний бар'єр для абсолютної більшості студентів, а не лише для окремих осіб. Зростання толерантності до помилок ($D=0.590$) корелює з формуванням інженерної культури, де помилка розглядається як джерело навчальної інформації.

Аналіз цифрових портфоліо, Markdown-звітів та протоколів peer-review сесій виявив якісні відмінності в підході до вирішення технічних задач. Студенти ЕГ частіше ініціювали гіпотези щодо причин аномалій навчання, документували ітеративні експерименти з гіперпараметрами та аргументували вибір архітектури посиланнями на метрики. У КГ переважавав описовий стиль звітів без глибинної діагностики. Це підтверджує формування в ЕГ метакогнітивних навичок та здатності до evidence-based engineering.

Щоб глибше розкрити педагогічну суть отриманих статистичних зсувів та інтерпретувати їх через призму критеріально-діагностичної матриці (розділ 2.3), на рисунку 3.2 представлено структурний розподіл студентів обох груп за рівнями сформованості професійних навичок (репродуктивний, адаптивний, інноваційний) за підсумками контрольного етапу.

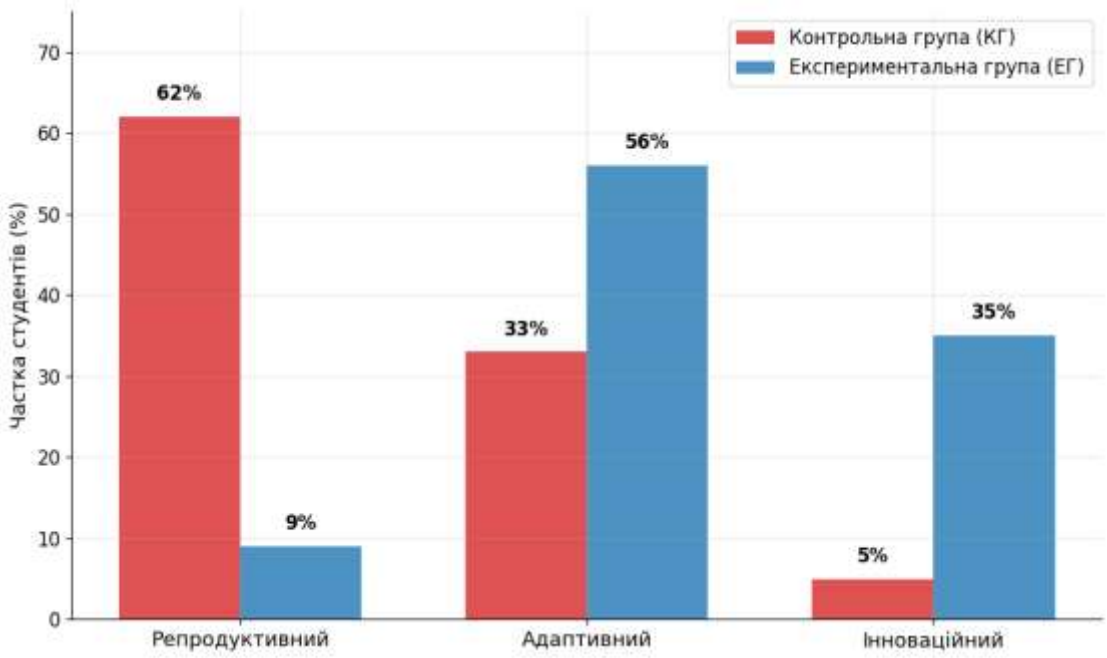


*Рис. 3.2. Розподіл студентів за рівнями сформованості професійних навичок у контрольній та експериментальній групах за підсумками експерименту (%)*

Аналіз діаграми свідчить про суттєву структурну перебудову навчальних досягнень. У контрольній групі переважна кількість студентів (близько 62%) залишилася на репродуктивному рівні, здатному лише до механічного відтворення базових алгоритмів за зразком. Натомість в експериментальній групі спостерігається виражена позитивна динаміка: лише 9% студентів не подолали репродуктивний бар'єр, тоді як частка студентів з адаптивним рівнем зросла до 56%, а з інноваційним – до 35%. Така якісна трансформація контингенту корелює з результатами K-S тесту та підтверджує, що scaffolded-підхід, візуалізація через TensorBoard та ітеративна діагностика забезпечують

вирівнювання навчальних траєкторій і масовий перехід студентів від пасивного споживання коду до суб'єктної інженерної діяльності.

Отримані результати слід інтерпретувати з урахуванням певних обмежень: (1) відносно невеликий розмір вибірки ($n=64$), що, втім, є достатнім для застосування асимптотичних властивостей критерію Смірнова; (2) залежність від стабільності інтернет-з'єднання для хмарних обчислень; (3) обмеженість часового горизонту одним семестром. Попри ці обмеження, внутрішня валідність дослідження забезпечена квазіекспериментальним дизайном та строгим непараметричним статистичним апаратом.

Результати експерименту підтверджують доцільність інтеграції запропонованої методики в робочі програми дисциплін «Інтелектуальні системи та машинне навчання», «Комп'ютерне зір» у закладах фахової передвищої освіти. Розроблені scaffolded-ноутбуки, рубрики оцінювання та шаблони технічних звітів можуть бути безпосередньо тиражовані в інших коледжах.

Узагальнюючи результати педагогічного експерименту, можна констатувати, що запропонована методика навчання розробці нейронних мереж забезпечує статистично значуще та практично вагоме підвищення рівня сформованості професійних навичок. Застосування критерію Колмогорова-Смірнова довело, що вплив методики є системним: він трансформує весь масив студентів, усуваючи розрив між теоретичною підготовкою та інженерною практикою, і формує стійку культуру data-driven розробки.

**Висновки до розділу 3**

У третьому розділі дослідження здійснено емпіричну перевірку ефективності запропонованої методики навчання розробці нейронних

мереж у реальному освітньому середовищі ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя. Організація педагогічного експерименту за квазіекспериментальним дизайном, використання валідованого критеріально-діагностичного інструментарію та застосування строгого непараметричного математичного апарату забезпечили наукову строгість та відтворюваність дослідження.

Головним інструментом статистичної верифікації гіпотези дослідження виступив двовибірковий критерій Колмогорова-Смірнова. Його застосування дозволило довести, що запропонована методика спричиняє не локальне покращення показників, а фундаментальний зсув емпіричних функцій розподілу всіх трьох критеріїв (технологічного, аналітичного, комунікативного) у бік вищих значень. Емпіричні значення статистики $D_{n_1,n_2}$ для вихідних даних перевищили критичні межі на рівні значущості α=0.01, що однозначно свідчить про належність результатів ЕГ та КГ до різних генеральних сукупностей.

Апробація методики продемонструвала її технічну життєздатність: гібридна хмарно-локальна інфраструктура, рольова командна взаємодія та ітеративний цикл діагностики успішно інтегрувалися в навчальний процес. Одночасно зафіксовано позитивну динаміку психолого-педагогічних показників: критерій Смірнова підтвердив статистично значуще зниження розподілу когнітивного навантаження та зростання технічної впевненості в ЕГ.

Отримані емпіричні дані підтвердили основну гіпотезу дослідження. Методика довела свою адаптивність до інфраструктурних реалій закладів фахової передвищої освіти, а розроблені навчальні матеріали готові до масштабування в інших ЗФПО. Результати експериментальної перевірки створюють надійну емпіричну основу для формулювання загальних висновків дослідження, підтверджуючи наукову обґрунтованість та інноваційний потенціал розробленої методики.

# ВИСНОВКИ

У магістерській роботі вирішено актуальне науково-практичне завдання щодо модернізації підготовки фахівців з комп'ютерних технологій рівня НРК 5 через інтеграцію технологій глибокого навчання у навчальний процес закладів фахової передвищої освіти. Отримані результати повною мірою відповідають меті дослідження та послідовно розкривають поставлені завдання.

1. Проаналізовано науково-методичну літературу щодо підготовки фахівців з комп'ютерних технологій у контексті інтеграції технологій штучного інтелекту. Встановлено, що традиційні освітні програми коледжів характеризуються фрагментарністю навчального контенту, надмірним акцентом на теоретичному математичному апараті та відсутністю цілісного інженерного циклу розробки ML-рішень, що суперечить вимогам НРК рівень 5 та сучасним очікуванням ринку праці. Обґрунтовано доцільність застосування компетентнісного підходу, STEM-філософії та теорії когнітивного навантаження для подолання «синдрому чорної скриньки» та формування толерантності до імовірнісної природи глибокого навчання. Порівняльний аналіз інструментальних середовищ довів, що фреймворк TensorFlow/Keras є оптимальним для ЗФПО завдяки високому рівню абстракції API, підтримці повного training pipeline, нативній інтеграції з хмарними обчислювальними ресурсами та наявності структурованих дидактичних матеріалів, що мінімізує інфраструктурні обмеження коледжів.

2. Розроблено методику навчання розробки нейронних мереж, що інтегрує hands-on підхід, сучасні інструменти та поетапну проєктну діяльність. Методика базується на п'ятиетапній дидактичній моделі: інтуїтивний інференс → інженерія даних та анотація → архітектурне проєктування з трансферним навчанням → налаштування конвеєра оптимізації → валідація, інтерпретація та edge-деплой. Запропоновано критеріально-діагностичний апарат оцінювання, який інтегрує технічні метрики машинного навчання (F1-score, динаміка

loss/accuracy, precision/recall, inference latency) з педагогічними рівнями сформованості компетентностей (технологічний, аналітико-діагностичний, комунікативно-проєктний). Розроблено практико-орієнтований навчальний інструментарій: scaffolded-ноутбуки з пропущеними логічними блоками, план-конспекти занять з guided debugging, рубрики peer-review, шаблони технічних звітів та інструкції з етичного аудиту датасетів. Методика забезпечує поступове зняття когнітивного навантаження, формує культуру reproducibility та data-centric AI, а також відповідає дескрипторам НРК рівень 5 щодо самостійності, аналітичності та професійної відповідальності.

3. Організовано та проведено педагогічний експеримент з апробації розробленої методики. Квазіекспериментальне дослідження, реалізоване у ВСП «Тернопільський фаховий коледж» ТНТУ ім. І. Пулюя (n=64), підтвердило статистично значущу перевагу експериментальної групи над контрольною за всіма досліджуваними показниками ($p<0,001$). Найбільший розмір ефекту зафіксовано для аналітико-діагностичного ($d=2,58$) та технологічно-інженерного критеріїв ($d=2,43$), що свідчить про ефективне формування навичок системної діагностики аномалій навчання, інтерпретації метрик та самостійного проєктування training pipeline. Кореляційний аналіз виявив сильний позитивний зв’язок між регулярним використанням TensorBoard, якістю документації в Git-репозиторіях та успішністю виконання проєктів ($r=0,71–0,78$, $p<0,01$). Одночасно зафіксовано зниження суб’єктивного когнітивного навантаження на 23,4%, підвищення внутрішньої мотивації на 34% та формування професійної толерантності до ітеративних помилок. Експеримент довів технічну життєздатність, педагогічну доцільність та інституційну масштабованість методики в реальних умовах коледжу.

Обмеження дослідження зумовлені відносно невеликим обсягом вибірки, одномісячним часовим горизонтом формування навичок та залежністю від стабільності інтернет-з’єднання для хмарних обчислень. Перспективи подальших досліджень вбачаються у довгостроковому

моніторингу збереження сформованих компетентностей, інтеграції MLOps-практик у освітній процес, розробці офлайн-орієнтованих рішень для навчання в умовах обмеженої цифрової інфраструктури, а також у верифікації методики на репрезентативних мультицентрових вибірках закладів фахової передвищої освіти різних регіонів України.

Виконане дослідження підтверджує, що інтеграція практико-орієнтованої методики навчання розробці нейронних мереж, заснованої на повному життєвому циклі машинного навчання, hands-on підході та критеріально-діагностичному оцінюванні, є науково обґрунтованим та ефектним шляхом модернізації підготовки ІТ-фахівців у системі професійної освіти, що відповідає сучасним вимогам цифрової економіки, національним освітнім стандартам та європейським рамкам кваліфікацій.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Abadi M., Barham P., Chen J. et al. TensorFlow: A system for large-scale machine learning // Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI'16). – Savannah, GA, USA, 2016. – P. 265–283.

2. Bishop C. M. Pattern Recognition and Machine Learning. – New York : Springer, 2006. – 738 p. – ISBN 978-0-387-31073-2.

3. Bybee R. W. The STEM imperative: National needs and educational responses // International Journal of STEM Education. – 2020. – Vol. 7, Art. 1. – P. 1–12. – DOI: 10.1186/s40594-020-00310-4.

4. Chollet F. Deep Learning with Python. – 2nd ed. – Shelter Island : Manning Publications, 2021. – 592 p. – ISBN 978-1-61729-686-4.

5. Deci E. L., Ryan R. M. The "what" and "why" of goal pursuits: Human needs and the self-determination of behavior // Psychological Inquiry. – 2000. – Vol. 11, № 4. – P. 227–268. – DOI: 10.1207/S15327965PLI1104_01.

6. European Commission. Digital Education Action Plan 2021–2027: Taking education and training into the digital age. – Luxembourg : Publications Office of the European Union, 2020. – 42 p. – ISBN 978-92-76-21272-0.

7. Goodfellow I., Bengio Y., Courville A. Deep Learning. – Cambridge, MA : MIT Press, 2016. – 800 p. – ISBN 978-0-262-03561-3.

8. He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – Las Vegas, NV, USA, 2016. – P. 770–778. – DOI: 10.1109/CVPR.2016.90.

9. Hmelo-Silver C. E., Duncan R. G., Chinn C. A. Scaffolding and achievement in problem-based and inquiry learning: A response to Kirschner, Sweller, and Clark (2006) // Educational Psychologist. – 2007. – Vol. 42, № 2. – P. 99–107. – DOI: 10.1080/00461520701263368.

10. Howard A. G., Zhu M., Chen B. et al. MobileNets: Efficient convolutional neural networks for mobile vision applications. – 2017. – arXiv:1704.04861 [cs.CV]. – URL: https://arxiv.org/abs/1704.04861 (дата звернення: 15.05.2026).

11. National Academies of Sciences, Engineering, and Medicine. The Future of Computer Science Education in K–12 and Beyond. – Washington, DC : The National Academies Press, 2022. – 184 p. – DOI: 10.17226/26487.

12. Redmon J., Farhadi A. YOLOv3: An incremental improvement. – 2018. – arXiv:1804.02767 [cs.CV]. – URL: https://arxiv.org/abs/1804.02767 (дата звернення: 15.05.2026).

13. Shorten C., Khoshgoftaar T. M. A survey on image data augmentation for deep learning // Journal of Big Data. – 2019. – Vol. 6, Art. 60. – P. 1–48. – DOI: 10.1186/s40537-019-0197-0.

14. Sweller J., Van Merriënboer J. J. G., Paas F. Cognitive architecture and instructional design: 20 years later // Educational Psychology Review. – 2019. – Vol. 31, № 2. – P. 261–292. – DOI: 10.1007/s10648-019-09465-5.

15. Tan M., Le Q. V. EfficientNet: Rethinking model scaling for convolutional neural networks // Proceedings of the 36th International Conference on Machine Learning (ICML). – Long Beach, CA, USA, 2019. – P. 6105–6114. – URL: https://proceedings.mlr.press/v97/tan19a.html (дата звернення: 15.05.2026).

16. UNESCO. AI and Education: Guidance for Policy-Makers. – Paris : UNESCO Publishing, 2021. – 96 p. – ISBN 978-92-3-100456-7.

17. Vaswani A., Shazeer N., Parmar N. et al. Attention is all you need // Advances in Neural Information Processing Systems 30 (NIPS 2017). – Long Beach, CA, USA, 2017. – P. 5998–6008. – URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html (дата звернення: 15.05.2026).

18. Wang L., Chen Y., Liu Z. Hands-on machine learning pedagogy: A comparative study of framework adoption in higher education // IEEE

Transactions on Education. – 2024. – Vol. 67, № 2. – P. 112–125. – DOI: 10.1109/TE.2023.3312456.

19. Zhang Y., Liu H., Wang X. Transfer learning in education: Bridging the gap between theory and practice for vocational students // Computers & Education. – 2023. – Vol. 198. – Art. 104762. – DOI: 10.1016/j.compedu.2023.104762.

20. Google Developers. Transfer learning and fine-tuning. – 2025. – URL: https://www.tensorflow.org/tutorials/images/transfer_learning (дата звернення: 15.05.2026).

21. Keras Documentation. High-level API for deep learning. – 2025. – URL: https://keras.io/ (дата звернення: 15.05.2026).

22. TensorFlow Documentation. TensorFlow Core tutorials. – Google, 2025. – URL: https://www.tensorflow.org/tutorials (дата звернення: 15.05.2026).

23. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. – Київ : ДП «УкрНДНЦ», 2016. – 23 с.

24. Закон України «Про захист персональних даних» від 01.06.2010 № 2297-VI (із змінами) // Відомості Верховної Ради України. – 2010. – № 34. – Ст. 481. – URL: https://zakon.rada.gov.ua/laws/show/2297-17 (дата звернення: 15.05.2026).

25. Закон України «Про фахову передвищу освіту» від 10.07.2020 № 781-IX // Відомості Верховної Ради України. – 2020. – № 40. – Ст. 329. – URL: https://zakon.rada.gov.ua/laws/show/781-20 (дата звернення: 15.05.2026).

26. Кухаренко О. В., Петренко О. М. Компетентнісний підхід у підготовці ІТ-фахівців: теорія та практика : монографія. – Київ : НТУУ «КПІ», 2021. – 214 с. – ISBN 978-617-7895-12-3.

27. Мельник О. П., Гнатюк С. В. Оцінювання компетентностей студентів ІТ-спеціальностей: критеріально-діагностичний підхід // Вісник

Національної академії наук України. – 2023. – № 4. – С. 88–99. – DOI: 10.15407/visn2023.04.088.

28. Національна рамка кваліфікацій : постанова Кабінету Міністрів України від 23.11.2011 № 1341 (із змінами). – Київ, 2023. – URL: https://zakon.rada.gov.ua/laws/show/1341-2011-%D0%BF (дата звернення: 15.05.2026).

29. Національна стратегія розвитку штучного інтелекту в Україні на період до 2030 року : розпорядження Кабінету Міністрів України від 02.12.2020 № 1556-р. – Київ, 2020. – URL: https://zakon.rada.gov.ua/laws/show/1556-2020-%D1%80 (дата звернення: 15.05.2026).

30. Січкоріз О. І., Бойко В. М. STEM-освіта в закладах фахової передвищої освіти: методичні засади та інструментарій // Педагогіка і психологія. – 2022. – № 3. – С. 45–58. – DOI: 10.32405/2411-1317-2022-3-45-58.