Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

УДК 05.13.06
DOI: https://doi.org/10.30837/2522-9818.2025.2.129

І. Чухран, С. Удовенко, В. Шергін, Л. Чала

МЕТОД ДОПОВНЕННЯ 3D-МОДЕЛЕЙ ТОЧКОВИХ ХМАР З ВИКОРИСТАННЯМ ГРАФОВИХ НЕЙРОННИХ МЕРЕЖ

Побудова моделей подання тривимірних об’єктів у вигляді просторової сукупності слабкоструктурованих точок (3D-моделей точкових хмар) набуває поширення в різних сферах, зокрема в завданнях створення віртуальної реальності, маршрутизації автономних роботів і 3D-реконструкції. Необхідність формувати та обробляти дані 3D-моделей хмар точок є суттєвою для застосунків, що потребують ретельного аналізу особливостей навколишнього середовища, пов’язаних з подоланням перешкод, визначенням маршрутів транспортних засобів та моделюванням реальних сцен. Утім, з різних причин точкові хмари часто містять пропущені ділянки, що створює значні проблеми для подальшого оброблення інформації. Неповні дані точкових хмар можуть мати серйозні наслідки, наприклад, у системах автономної навігації, де помилки призводять до зіткнень або інших небезпечних ситуацій. Розв’язання цієї проблеми є ключовим для надійного оброблення 3D-даних. **Метою статті** є розроблення та дослідження методу автоматичного доповнення й реконструкції точкових хмар з використанням графових нейронних мереж. Основними **завданнями дослідження** є аналіз наявних підходів до побудови та відновлення тривимірних графових моделей, розроблення та програмна реалізація методу автоматичного доповнення точкових хмар з використанням графових нейронних мереж, а також моделювання запропонованого методу для завдань доповнення та 3D-реконструкції моделей точкових хмар. У роботі розглянуто загальну модель доповнення точкових хмар із застосуванням графових нейронних мереж, що дають змогу кодувати неповні 3D-моделі хмар точок із використанням графів та прогнозуванням положення відсутніх фрагментів моделей. Запропоноване рішення для доповнення неповних 3D-хмар точок має наукову новизну та поєднує потужність графових нейронних мереж (GNN) з архітектурою мережі *Point Completion Network* (PCN). Окреслений підхід сприяє якісному відновленню неповних 3D-даних у практичному застосуванні. Значущість роботи підтверджують результати моделювання запропонованого методу для класичних датасетів та їх порівняння з деякими наявними підходами до розв’язання досліджуваної проблеми. Перспективним напрямом продовження досліджень з окресленої теми є випробування різних архітектур нейронних графових мереж, налаштування гіперпараметрів, застосування альтернативних функцій втрат і більш потужних обчислювальних ресурсів для навчання побудованих нейромережних моделей.

Ключові слова: точкова хмара; глибоке навчання; графова нейронна мережа; автоматичне доповнення точкових хмар; кодування графів; 3D-реконструкція графових моделей.

Вступ

Застосування подання тривимірних об’єктів або середовищ у формі неструктурованих наборів точок у просторі, відомих як 3D-хмари точок, набуває дедалі більшого поширення в таких галузях, як автономна навігація, робототехніка, технології віртуальної та доповненої реальності, а також тривимірна реконструкція.

Відповідно до усталених наукових визначень 3D-хмара точок є сукупністю даних, що містить точки, координати яких задані в тривимірній системі та які відтворюють зовнішню поверхню об’єктів або просторових середовищ. Такі дані зазвичай отримують за допомогою сучасних методів лазерного сканування [1] або способом фотограмметричного аналізу, коли вимірювання відстаней трансформуються в множину точок, що характеризують поверхню досліджуваних об’єктів.

Зберігання даних 3D-хмар точок можливе в різних сучасних форматах, які мають унікальні властивості, адаптовані до особливих потреб оброблення та прикладного використання. Серед найбільш поширених форматів можна виокремити:

− формат PLY (*Polygon File Format*), що дає змогу зберігати тривимірні скановані дані;

− формат OBJ (*Object File Format*), що використовується, зокрема, для зберігання 3D-хмар точок та містить інформацію про топологію сіток та вершин;

− формати PTS або PTX, що застосовуються в геодезичних програмних застосунках для зберігання та оброблення відсканованих тривимірних наборів даних;

− формат LAS (*LiDAR Aerial Surveying Format*), що впроваджується для зберігання та оброблення великих обсягів геопросторових даних;

− формат XYZ, що зберігає 3D-координати хмар точок разом з додатковою інформацією про їх колір.

© І. Чухран, С. Удовенко, В. Шергін, Л. Чала, 2025

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

Здатність до точного захоплення та оброблення даних тривимірних (3D) хмар точок є критично важливою для застосунків, що потребують глибокого розуміння просторового середовища, зокрема для завдань уникнення перешкод, планування траєкторій та моделювання сценічного простору. Проте внаслідок таких факторів, як оклюзії, обмеження сенсорних систем або недоліки процедур збору даних, 3D-хмари точок часто визначаються наявністю пропущених ділянок або неповним відтворенням об'єктів чи сцен. Ця неповнота суттєво перешкоджає подальшим етапам оброблення, які залежать від точної та вичерпної тривимірної інформації, що може спричинити помилки або зниження ефективності в критично важливих системах.

Неповнота даних у хмарах точок має особливо серйозні наслідки для застосунків, де безпека є пріоритетом, наприклад, у системах автономної навігації безпілотних транспортних засобів або роботизованих операціях у динамічних середовищах. Відсутність чи неточність інформації про перешкоди, особливості рельєфу або інші елементи оточення може спричинити зіткнення чи навігаційні збої. Крім того, неповні хмари точок ускладнюють адекватне сприйняття та моделювання сцен, що обмежує можливості таких технологій, як віртуальна та доповнена реальність і тривимірна реконструкція. Отже, розроблення методів доповнення хмар точок є ключовим для ефективного оброблення 3D-даних і максимального розкриття потенціалу відповідних застосунків.

Традиційні методи, зокрема геометричні підходи чи техніки реконструкції поверхонь, часто виявляються недостатньо ефективними в разі наявності складних структур об'єктів моделювання або значних (у відсотках) відсутніх ділянок 3D-моделей, а також за умов неоднорідної щільності хмар точок і присутності шумів. Останні досягнення у сфері глибокого навчання, зокрема в геометричному глибокому навчанні, відкривають нові перспективи для розв'язання окресленої проблеми. Графові нейронні мережі (GNN), що належать до моделей глибокого навчання, адаптованих до роботи з нерегулярними й неевклідовими структурами даних, продемонстрували перспективні результати в завданнях, пов'язаних із класифікацією, кластеризацією, сегментацією та генерацією даних 3D-хмар точок.

У цьому дослідженні запропоновано інноваційний підхід до доповнення хмар точок, який ґрунтується на використанні графових нейронних мереж. Цей метод передбачає необхідність кодування вхідної хмари точок із застосуванням графової моделі, точки якої є вузлами, а ребра відтворюють геометричні зв'язки між точками. Таке графове подання забезпечує ефективне моделювання нерегулярної структури даних хмари точок і є основою для побудови моделі кодера, що має навчитися виділяти ключові ознаки та формувати подання на основі структури графа.

Упровадження графових моделей дає змогу певною мірою вдосконалити технології оброблення хмари точок із застосуванням геометричного глибокого навчання. Автоматичне доповнення відсутніх фрагментів хмар сприяє підвищенню безпеки функціювання критично важливих систем, що мають використовувати достовірну 3D-інформацію щодо особливостей навколишнього середовища.

Мета статті – розроблення та дослідження методу автоматичного доповнення та реконструкції точкових хмар з використанням графових нейронних мереж.

Завдання дослідження:

– аналіз наявних підходів до автоматичного доповнення хмар точок;

– порівняння метрик, що можуть бути застосовані для побудови графічних моделей хмар точок;

– розроблення методу автоматичного доповнення точкових хмар з використанням графових нейронних мереж;

– моделювання запропонованого методу для завдань доповнення та 3D-реконструкції моделей точкових хмар.

Аналіз наявних підходів до автоматичного доповнення хмар точок

Побудова та доповнення тривимірних моделей базується на використанні традиційних методів і методів глибокого навчання. До традиційних насамперед належать геометричні методи та методи вирівнювання.

Традиційними методами доповнення хмар точок є методи на основі геометрії хмар і методи на основі вирівнювання. Методи, що ґрунтуються на геометричному підході [2], використовують геометричні властивості, такі як симетрія форми та її дисперсія, отримані з наявної бази даних форм, для реконструкції повноцінної моделі на основі

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32) ISSN 2524-2296 (online)

часткових вхідних даних. Деякі із зазначених методів [3–5] локально заповнюють пропущені ділянки внаслідок інтерполяції гладких поверхонь на основі сусідніх структур. Проте такі підходи виявляються недостатньо ефективними в разі, коли вхідні моделі мають значні пропущені ділянки, оскільки їм бракує семантичної інформації, наприклад, про структуру чи топологію. Водночас, зважаючи на симетричність багатьох об'єктів, окремі методи [6, 7] здатні виявляти симетрію у вхідних тривимірних моделях і заповнювати відсутні частини, спираючись на їх симетричну геометрію.

Методи, основані на вирівнюванні [8, 9], використовують великомасштабні бази даних хмар точок для пошуку відповідних фрагментів, які згодом застосовуються для заповнення пропущених ділянок. Ключовим етапом у цих підходах є формування бази даних, яка охоплює тривимірні форми однієї категорії. Деякі методи [10] обирають окремі частини моделей як базові елементи, тоді як інші [11, 12] використовують деформацію тривимірних фігур для створення таких баз. Зазначений підхід дає змогу успішно доповнювати 3D-моделі зі схожою структурою, однак його ефективність знижується, коли вхідні моделі виходять за межі заздалегідь визначених категорій.

Отже, традиційні методи значною мірою покладаються на геометричні властивості, сформовані вручну, для виведення структури відсутніх ділянок на основі часткової форми. Їх результативність суттєво обмежена в разі, коли тривимірні моделі не відповідають попередньо встановленим припущенням чи знанням, що ускладнює досягнення задовільних результатів.

Методи доповнення тривимірних моделей з використанням глибокого контрольованого або неконтрольованого навчання передбачають необхідність побудови спеціалізованих нейромережних моделей.

Нейромережний підхід до вирівнювання хмар точок, оснований на навчанні нейронних моделей, запропонований у роботі [13], де використано рекурентні нейронні мережі для кодування двовимірних хмар точок із подальшим декодуванням у контексті розв'язання геометричних задач, зокрема двовимірної задачі комівояжера. Запропонований у цій статті метод також розв'язує проблему генерації хмар точок змінного розміру з використанням механізму уваги, однак він не адаптований до тривимірного (3D) простору.

У дослідженні [14] запропоновано підхід, який базується на поданні 3D-сітки об'єкта у вигляді набору фрагментів атласу частин об'єкта. Первинні форми фрагментів кодуються за допомогою багатошарового перцептрона в латентний простір, після чого точки декодуються в латентне подання щодо закодованої форми-зразка, що дає змогу генерувати 3D-сітки замість хмар точок.

У праці [15] запропоновано доповнювати хмари точок на основі груп точок, а не окремих одиниць, що дає змогу кожну модель подавати у вигляді тривимірної воксельної сітки. У статті [16] розглянуто використання воксельних мереж для доповнення хмар точок способом генерації країв. На початковому етапі хмари точок інтегруються в регулярні воксельні сітки, а потім за допомогою ребер уявної форми створюються цілісні об'єкти.

У дослідженнях [17, 18] проаналізовано підхід, що передбачає застосування одновимірних згорток до точок для розв'язання задач класифікації та сегментації на наборах хмар точок із використанням простого та ієрархічного методів. У згаданих студіях спочатку формуються локальні та глобальні ознаки, які разом із обробленими даними передаються до кодувальної мережі. На подальших етапах ці дані трансформуються в хмари точок із грубим і тонким виходом, де грубий вихід зберігає лише ключові точки, що сприяють формуванню більш точного результату на наступних етапах генерації.

Аналогічний підхід застосовано в роботі [19], де використано кодувальник PCN для вибірки різних поверхонь. У цьому разі попередня інформація збирається подібно до глобальних ознак з набору поверхонь. Мережа декодувальника навчається відбирати потрібні точки на основі аналізу узагальненого вектора кодованих ознак неповного вихідного об'єкта. Для подолання обмежень автокодера, таких як ігнорування структури, злиття та уточнення, у дослідженні [20] запропоновано метод вибірки за критерієм мінімальної щільності. Цей підхід визначає відновлювану точку з найменшою щільністю, що підвищує ефективність оброблення.

У праці [21] для заповнення хмар точок на основі реальних даних застосовано змагальне навчання. У роботі [22] запропоновано використання генеративної змагальної мережі (GAN), у якій генератор, що є адаптаційною мережею, трансформує

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

вхідну інформацію в латентне подання так, щоб дискримінатор не розрізняв результати перетворення від результатів, отриманих з навчальних наборів даних. Генератор, власне, виконує основну функцію відтворення необроблених часткових наборів точок у завершені та цілісні набори, що реалізується внаслідок упорядкованої роботи у двох окремих латентних просторах, сформованих на основі відсканованих і синтезованих даних об'єктів.

Альтернативний підхід, подібний до GAN, розглянуто в роботі [23], де генерація загальної форми хмари точок здійснюється з неповного набору точок із неминучими змінами наявних точок, а також з огляду на шум і геометричні втрати. У цьому методі мережа ієрархічно оцінює відсутні частини хмари точок, застосовуючи великомасштабну генеративну мережу, що базується на характерних точках.

Окремі дослідження спрямовані на вдосконалення процесу доповнення хмар точок. Так, у роботі [24] запропоновано нейронні мережі з архітектурою кодувальник-декодувальник для безпосереднього оцінювання всієї хмари точок на основі неповних даних. Цей підхід реалізовано через наскрізну нейронну архітектуру, що зосереджується на прогнозуванні відсутньої геометрії та інтеграції відомої вхідної інформації з передбаченою хмарою точок. Метод містить дві нейронні мережі: перша відповідає за прогнозування відсутньої частини, витягуючи інформацію з неповних даних, а друга забезпечує об'єднання та уточнення розподілу точок у результуючій хмарі.

У статті [25] запропоновано нейромережу автокодера, що аналізує латентні подання та формує кілька зразків доповнених 3D-хмар точок. У цьому разі інформація, що обробляється, розподіляється на два окремі потоки, а заповнення прогалин реалізується через парадигму гіпермереж. Гіпермережі, як зазначено в роботі [26], є нейронними мережами, що генерують ваги для цільової мережі, оптимізуючи її функціональність.

Інші підходи пропонують ітеративну генерацію відсутніх частин хмари точок способом поступового додавання нових елементів до наявних. Наприклад, у праці [27] завершення хмар точок у 3D-просторі здійснюється за принципом генерування дочірніх точок унаслідок послідовного розщеплення батьківських точок із використанням оператора *Snowflake Point Deconvolution* (SPD), запропонованого авторами.

У низці досліджень описані рішення, які, на нашу думку, є найбільш перспективними з погляду уваги до геометричних особливостей із застосуванням графових нейронних мереж. Зокрема в роботі [28] подано синтез мережі ECG (*Edge-aware Completion Graph*) для завершення хмар точок, що інтегрує ребра з графічною згорткою. Цей підхід сприяє генерації деталізованої тривимірної (3D) хмари точок із різномасштабними крайовими властивостями. Щоб зважати на локальні геометричні деталі в разі розширеної вибірки, автори запропонували модуль *Edge-aware Feature Expansion* (EFE), який забезпечує плавне розширення та вдосконалення вибірки точкових об'єктів з огляду на їх локальні краї.

Відповідно до мети цього дослідження, далі буде розглянуто завдання розроблення методу доповнення 3D-хмар точок із використанням графових нейронних мереж.

Метрики, що можуть бути застосовані для побудови графічних моделей хмар точок

У межах аналізу можливих метрик для оцінювання відстані між наборами даних тривимірних точок розглядається проблема оцінювання якості реконструкції 3D-моделей, що передбачає порівняння двох поверхонь. Такими можуть бути, наприклад, поверхня, побудована на основі хмари 3D-точок, та еталонна поверхня. Відомі метрики в цьому разі базуються на оцінці похибок відстані між поверхнями. Їх сутність полягає в обчисленні відстані між всіма точками однієї поверхні до найближчих точок іншої поверхні з подальшим узагальненням результатів у вигляді статистичних показників. Найпоширенішими метриками цього типу є середньоквадратична помилка (RMSE) та середня абсолютна помилка (MAE). Для зазначених метрик точність реконструкції є часткою правильно реконструйованих точок у загальному наборі реконструйованих даних, тобто тих точок, відстань від яких до найближчої точки еталонної поверхні не перевищує заданого порогу.

Окремі метрики продемонстрували свою ефективність в оцінюванні результатів сегментації, передбаченої моделями глибокого навчання. Їх основна ідея полягає в одночасному вимірюванні точності класифікації та коректності локалізації. До таких метрик належать точність (*precision*),

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

відгук (*recall*), індекс перетину над об'єднанням (IoU, *Jaccard Index*) та показник F1. Однак застосування зазначених метрик може потребувати вокселізації вибірки даних, що дещо знижує довіру до результатів оцінювання. Далі розглядаються найбільш поширені метрики цього типу.

Метрика просторової відстані Хаусдорфа (HD) інколи використовується для оцінювання результатів 3D-моделювання. Вона дає змогу оцінювати відстань між точками P_A та P_B графової моделі з використанням функції найбільш близьких сусідів $NN(x, P)$:

$$HD(P_A, P_B) = \frac{1}{2}\max_{x\in P_A}\left|x - NN(x, P_B)\right| + \frac{1}{2}\max_{x\in P_B}\left|x - NN(x, P_A)\right|, \quad (1)$$

де $NN(x, P) = \arg\min_{x'\in P} \| x - x' \|$.

Якщо розглядати розподіли ймовірностей P_A та P_B для наборів хмар точок, то відстань між цими хмарами можна визначати як дивергенцію Кульбака – Лейблера (KL):

$$D_{KL}(P_A \| P_B) = \sum_{x\in\chi} P_A(x)\log\left(\frac{P_A(x)}{P_B(x)}\right), \quad (2)$$

де χ – простір вибірки для точок хмар, що порівнюються.

Показник KL дає змогу, зокрема, оцінити різницю між синтезованою та реальною вибірками в процесі 3D-реконструкції. Альтернативним показником, що для визначення різниці між 3D-об'єктами також використовує розподіли ймовірностей, є відстань *Earth-Mover's Distance* (EMD):

$$EMD(P_A, P_B) = \min_{\pi\in\Pi(P_A, P_B)}\left[\sum_{i=1}^{n}\sum_{j=1}^{m}\pi_{ij}\left|p_{Ai} - p_{Bj}\right|\right]. \quad (3)$$

Іншим підходом, що бере до уваги геометрію, є обчислення відстані фаски (*Chamfer distance*), яка визначається як середня відстань між парами найближчих сусідів між двома наборами хмари точок P_A та P_B відповідно:

$$CD(P_A, P_B) = \frac{1}{2n}\sum_{i=1}^{n}\left|x_i - NN(x_i, P_B)\right| + \frac{1}{2m}\sum_{j=1}^{m}\left|x_j - NN(x_j, P_A)\right|. \quad (4)$$

За припущенням, що хмари точок є вирівняними по осях, а навколо них існує канонічна сітка вокселів, доцільно використовувати метрику розбіжності Дженсена – Шеннона (*Jensen-Shannon Divergence* – *JSD*) між граничними розподілами, визначеними в евклідовому 3D-просторі. Ця метрика дає змогу оцінити, наскільки хмари точок A мають тенденцію збігатися в середньому з хмарами точок B. Для цього підраховуємо кількість точок, що лежать у межах кожного вокселя для всіх хмар точок A та, відповідно, для B, й оцінюємо розбіжність *JSD* між отриманими емпіричними розподілами (P_A, P_B).

$$JSD(P_A \| P_B) = \frac{1}{2}D(P_A \| M) + \frac{1}{2}D(P_B \| M), \quad (5)$$

де $M = \frac{1}{2}(P_A + P_B)$ та $D(\cdot\|\cdot)$ – KL-дивергенція між двома розподілами [29].

Для кожної хмари точок в A спочатку знаходимо її найближчого сусіда в B. Покриття вимірюється як частка хмар точок у B, які були зіставлені з хмарами точок в A. Близькість може бути обчислена за допомогою *CD*- або *EMD*- відстані між точками з використанням метрики *COV-CD* або *COV-EMD*:

$$COV(P_A, P_B) = \frac{\left|\arg\min_{Y\in P_B} D(X, Y) \middle| X \in P_A\right|}{|P_B|}, \quad (6)$$

де D може бути *CD* або *EMD*.

Високий показник покриття свідчить про те, що більша частина B подана в межах A. Покриття не вказує на те, наскільки добре хмари точок містяться в множині A; приклади, що збігаються, не обов'язково мають бути близькими. Для оцінювання точності A щодо B кожна хмара точок B зіставляється з хмарою точок A з мінімальною відстанню (MMD) і визначається середнє значення відстаней у зіставленні. Тут може бути використана будь-яка відстань між наборами точок, що дає метрики *MMD-CD* і *MMD-EMD*:

$$MMD(P_A, P_B) = \frac{1}{|P_B|}\sum_{Y\in P_B}\min_{X\in P_A} D(X, Y), \quad (7)$$

де D відповідає значенням *CD* або *EMD*.

Показник *MMD* безпосередньо залежить від відстаней між порівнюваними наборами точок.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

Загалом метрики *MMD*, *COV-CD* та *COV-EMD* добре корелюють між собою. Метрика *JSD* відрізняється від них тим, що оцінює подібність між A і B лише за допомогою граничної статистики й вимагає попередньо вирівняних даних, але водночас є більш зручною для обчислень. Проте ця метрика є ефективною, коли необхідно здійснювати послідовність порівнянь між 3D-моделями хмар точок.

Метод автоматичного доповнення точкових хмар з використанням графових нейронних мереж

Розглянемо особливості структури графових нейронних мереж (GNN) та їх функціональних компонентів. Графові мережі є методами глибокого навчання, які працюють на основі використання графів. Ці мережі дають змогу аналізувати та обробляти неевклідові дані, такі як графи, що містять вузли та ребра [30]. Спочатку такі мережі були вперше використані на спрямованих ациклічних графах. Пізніше графові моделі були застосовані до рекурентних нейронних мереж та нейронних мереж прямого поширення, що вимагають ітераційного визначення переходу станів на графах.

Останні дослідження у сфері глибоких нейронних мереж, зокрема згорткових нейронних мереж (CNN) [31], привели до розширення можливостей GNN. Мережі CNN оперують з різномасштабними просторовими ознаками та компонують їх для побудови якісних зображень [32]. Для CNN властиве локальне з'єднання використання декількох шарів. У цих мережах обробляються евклідові дані для опису зображень і текстів, що дає змогу застосовувати графове подання. Однак узагальнення CNN на графові моделі не завжди сприяє визначенню локалізованих фільтрів згортки та операторів об'єднання, що перешкоджає поширенню моделей CNN з евклідової ділянки до неевклідової (рис. 1).

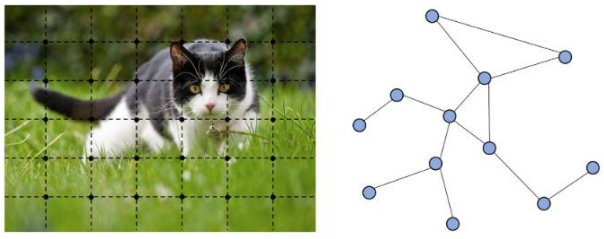


Рис. 1. Приклад зображення в евклідовому просторі (ліворуч) та його графічного спрощеного подання в неевклідовому просторі (праворуч)

Застосування геометричного глибокого навчання для побудови графічних моделей в неевклідовому просторі є новим напрямом досліджень у сфері машинного навчання. Ефективним є навчання графів [33], де здійснюється подання вершин, ребер або підграфів графів у вигляді векторів невеликої розмірності. Але традиційні підходи до машинного навчання графових моделей здебільшого використовують функції, які розробляються вручну, що зумовлює їх негнучкість і високу обчислювальну вартість. У роботах [34, 35] розглянуто метод автоматичного вбудовування графів на основі згенерованих випадкових блукань. Однак у цьому методі параметри не розподіляються між вузлами кодувальної частини моделі, що спричиняє лінійне зростання обчислюваних витрат зі зростанням кількості вузлів. Крім того, використання прямого вбудовування обмежує його здатність до узагальнення (це означає, що він не може працювати з новими типами графів).

Розглянемо деякі термінологічні особливості застосування графових моделей.

Графи цих моделей можна розподілити на структуровані та неструктуровані. Структуровані графи мають явно визначену будову, наприклад, молекули, фізичні системи та графи знань. Неструктуровані графи, навпаки, мають неявну структуру, яка має бути побудована на основі конкретного завдання, як-от графи слів або графи-сценарії для зображень. Також графи можуть бути орієнтованими (із заданими напрямками ребер між вершинами) та неорієнтованими. Графи, що можуть змінюватися з часом, називаються динамічними, а ті, що не залежать від часу, вважаються статичними.

Можна виокремити три основні типи завдань, пов'язаних з аналізом графів:

– завдання на рівні вузлів (зокрема класифікація, кластеризація та регресія вузлів);

– завдання на рівні ребер (а саме класифікація типів ребер та прогнозування зв'язків між вершинами графа);

– завдання на рівні графів (зокрема класифікація, регресія та порівняння графів).

В алгоритмах навчання мереж на основі графів можливе використання трьох різних режимів навчання:

– контрольоване, що надає позначену для навчання інформацію;

– напівконтрольоване навчання, що передбачає наявність незначної кількості позначених вузлів

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

і великої кількості непозначених вузлів для навчання (в більшості задач класифікації вершин і ребер упроваджується цей тип навчання);

– неконтрольоване навчання, що пропонує лише немарковані дані для налаштування моделі (цей тип начання використовується, зазвичай, для кластеризації вузлів).

Для побудови графових нейронних мереж застосовують такі основні обчислювальні модулі:

– модуль поширення (для поширення характеристичної та топологічної інформації між вузлами мережі); у цих модулях оператор згортки застосовується для агрегування інформації від сусідніх вузлів;

– модуль вибірки (для проведення поширення на графах великої розмірності); цей модуль зазвичай поєднується з модулем поширення;

– модуль об'єднання або субдискретизації (використовується для отримання інформації з вузлів у разі необхідності подання високорівневих підграфів або графів).

Загальну структуру конвеєра GNN зображено на рис. 2.

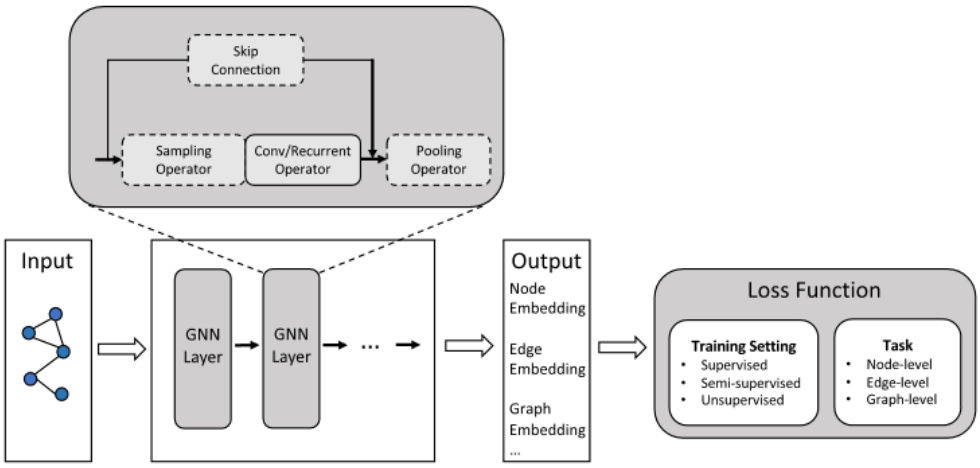


Рис. 2. Загальна структура конвеєра GNN

Розглянемо більш детально функціональні особливості окремих компонентів GNN.

Модуль поширення в моделях GNN реалізує операції згортки, основна ідея яких полягає в узагальненні згорток з різних ділянок графа. У модулі поширення використовуються як спектральні, так і просторові підходи. Спектральні базуються на обробленні графових сигналів з визначенням операторів згортки в спектральній ділянці. У цьому разі графічний сигнал x спочатку подається в спектральній ділянці за допомогою графового перетворення Фур'є, а потім здійснюється операція згортки.

Просторові підходи ґрунтуються на реалізації згорток на графі для просторово близьких сусідів. Вони основані на теоремі згортки та спрямовані на використання локалізованих операторів, що можуть бути розпаралелені та масштабовані. Відповідні перетворення формально визначаються як

$$\mathcal{F}(x) = U^T x, \quad \mathcal{F}^{-1}(x) = Ux, \tag{8}$$

де U – матриця власних векторів нормалізованого лапласіана графа

$$L = I_N - D^{-1/2} A D^{-1/2}, \tag{9}$$

де D – матриця степенів; A – матриця суміжності графа.

Лапласіан нормалізованого графа є дійсною симетричною додатною напівскінченністю, тому його можна розкласти на множники у вигляді

$$L = U \Lambda U^T, \tag{10}$$

де Λ – діагональна матриця власних значень.

З огляду на теореми згортки операція згортки здійснюється таким чином [36]:

$$g * x = \mathcal{F}^{-1}\left(\mathcal{F}(g) \odot \mathcal{F}(x)\right) = U\left(U^T g \odot U^T x\right). \tag{11}$$

Формулу (11) можна спростити, якщо замість $U^T g$ використовувати навчальну матрицю g_w:

$$g_w * x = U g_w U^T x. \tag{12}$$

У цьому разі можуть бути застосовані різні типи навчальних матриць. Зокрема в спектральній мережі

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

може бути використана діагональна навчальна матриця-фільтр:

$$g_w = \mathrm{diag}(w);\ \ w \in \mathbb{R}^N. \qquad (13)$$

Однак фільтр (13) не є просторово локалізованим, що ускладнює його практичну реалізацію. У роботі [37] g_w апроксимується з використанням усіченого розкладання за допомогою поліномів Чебишева $T_k(x)$ до K^{th} порядку:

$$g_w * x \approx \sum_{k=0}^{K} w_k T_k(\bar{L}), \qquad (14)$$

де $\bar{L} = \frac{2}{\lambda_{max}} L - I_N$; λ_{max} – максимальне власне значення матриці L; w – вектор коефіцієнтів Чебишева.

У рівнянні (14) поліноми Чебишева визначаються в такий спосіб:

$$T_k(x) = 2xT_{k-1}(x), \qquad (15)$$

де $T_0(x) = 1$ та $T_1(x) = x$.

У праці [38] показано, що для $\lambda_{\max} \approx 2$ і для двох вільних параметрів w_0 та w_1 можна подати рівняння (13) в такому вигляді:

$$g_w * x \approx w_0 x + w_1 (L - I_N) x = w_0 x - w_1 D^{-1/2} A D^{-1/2} x, \qquad (16)$$

або (за умови $w = w_0 = -w_1$)

$$g_w * x \approx w\left(I_N - D^{-1/2} A D^{-1/2}\right) x. \qquad (17)$$

Практичне застосування модуля поширення в моделях GNN пов'язане з використанням інтеграції механізму уваги (GAT), що дає змогу обчислювати приховані стани кожного вузла на основі стратегії самоуваги. До того ж механізм GAT використовує багатоголову увагу для покращення процесу навчання мережі, формуючи K незалежних матриць уваги для обчислення прихованих станів, які потім об'єднуються. Іноді доцільно регулювати кількість шарів k графової нейронної мережі для покращення результатів. Збільшення кількості шарів дає змогу кожному вузлу отримувати більше інформації від сусідніх вузлів. Однак експериментально встановлено, що більш глибокі моделі не завжди покращують роботу мережі, а занадто глибокі моделі можуть навіть працювати гірше. Це пов'язано з тим, що занадто велика кількість шарів призводить до поширення зашумленої інформації від сусідів, унаслідок чого виникає ефект надмірного згладжування, тобто вузли починають мати схожі подання після агрегування.

Ефективним підходом для розв'язання окресленої проблеми є використання модулів вибірки на великих графах. Існують такі основні різновиди модулів вибірки графів: вибірка вершин (вузлів), вибірка шарів та вибірка підграфів. Одним зі способів зменшення кількості сусідніх вершин є вибірка скороченої множини вузлів з околів кожної вершини (операція *GraphSAGE*) [39]. У цьому разі обирається фіксована обмежена кількість сусідів (до 40 для однієї вершини).

Модуль пошарової вибірки (операція *FastGCN*) використовує обмежений набір вузлів для агрегації в кожному шарі з контрольованим коефіцієнтом розширення. У цьому разі важливі вузли із значною ймовірністю потрапляють до вибірки.

Модуль вибірки підграфів (операція *ClusterGCN*) способом кластеризації графів здійснює вибірку декількох підграфів графової моделі та пошук околів у межах обраних підграфів.

У мережах GNN для формування узагальнених характеристик використовуються модулі прямого об'єднання (субдискретизації) та модулі ієрархічного об'єднання. Модулі прямого об'єднання безпосередньо застосовують графову модель, упроваджуючи різні стратегії вибору вузлів. Модулі ієрархічного об'єднання реалізують багаторівневу стратегію для GNN, що дає змогу захоплювати локальну й глобальну інформацію з вхідного графа. Ця стратегія передбачає рекурсивне застосування операції об'єднання для поступового формування сукупності підграфів з ієрархічною структурою. На кожному рівні ієрархії із вхідного графа вилучаються вкладені вершини. Далі вилучені вершини об'єднуються та передаються на наступний структурний рівень. Ця операція триває до досягнення заданої кількості рівнів з остаточним поданням вузлів мережі.

У статті запропоновано метод доповнення хмари точок, що дає змогу відновлювати відсутні фрагменти 3D-зображень. Метод використовує графові нейронні мережі глибокого навчання, які обробляють просторові дані вхідних графів з частково викривленою структурою.

Запропонований метод передбачає кодування хмари точок або її частини у форматі графа, кожна точка якого є вузлом, а ребра визначають локальні геометричні зв'язки між вузлами. Таке графове подання є вхідною інформацією для графової моделі кодера доповненої мережі, наприклад, графової

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

згорткової мережі (GCN) або *GraphSAGE*, яка навчається витягувати важливі ознаки та подання з графа хмари точок.

Цей підхід орієнтований на розв'язання таких завдань формування та доповнення хмар точок:

– оброблення інформації, що не має регулярної сіткової структури (на відміну від традиційних згорткових нейронних мереж графові нейронні мережі ефективно справляються з нерегулярними даними, зокрема і з викривленими хмарами точок);

– фіксація геометричних зв'язків між вершинами графової моделі (графове подання хмари точок дає змогу виявляти як локальні геометричні зв'язки через ребра, так і глобальну структурну інформацію через загальне графове подання моделі, що сприяє підвищенню якості відновлення відсутніх фрагментів);

– масштабування графових моделей (мережі GNN здатні обробляти розріджені графові подання структури моделі, що дає змогу ефективно працювати з великомасштабними хмарами точок різної щільності);

– забезпечення надійності та узагальнення (мережі GNN використовують закодоване графічне подання структурної інформації, закодованої в графі, що забезпечує можливість моделі навчатися на надійних зразках та узагальнювати її за наявності неповних точкових даних).

Ефективність запропонованого методу оцінюватиметься за якістю доповнених хмар точок, що вимірюється такими показниками: відстань між точками, повнота та візуальна перевірка. Розглянемо загальний опис цього методу. У попередніх дослідженнях з реконструкції неповних хмар точок графові нейронні мережі застосовувалися здебільшого для оброблення закодованих особливостей моделі.

У цій роботі запропоновано безпосереднє використання графової структури точкових хмар. Синтез графової моделі здійснюється за допомогою алгоритму найближчих сусідів KNN (*K-nearest neighbors*). Зазначимо, що інші методи, зокрема побудова графа за радіусами точок, є менш стабільними та ефективними через ризик захоплення надмірної кількості зв'язків, якщо не передбачено додаткової адаптації радіуса. Натомість KNN завжди повертає чітко визначену кількість сусідів для кожної точки (рис. 3).

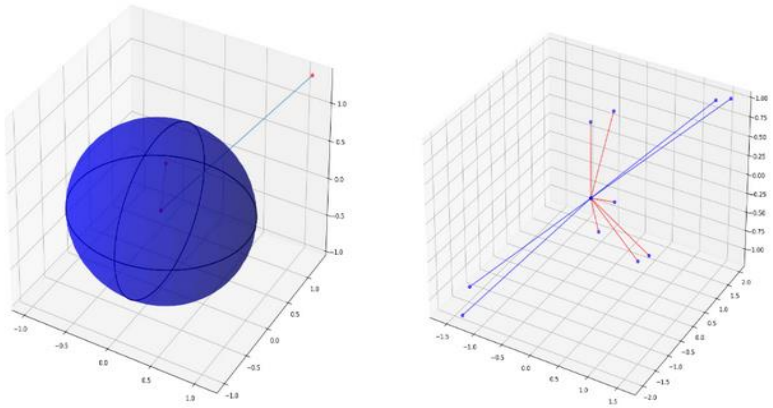


Рис. 3. Побудова графа за радіусами точок (праворуч) та відповідно до алгоритму KNN (ліворуч)

Модель кодера мережі GNN здатна в процесі навчання уточнювати структуру й локальні зв'язки в навчальних даних хмари точок. Навчена модель дає змогу використовувати фрагменти хмари точок як вхідну інформацію та генерувати відсутні точки (доповнювати модель) на основі раніше отриманих закономірностей.

Архітектура мережі, що реалізує запропонований метод, базується на мережі *Point Completion Network* (PCN), запропонованій у роботі [17], де розглянуто завдання реконструкції точкових хмар (рис. 4).

Мережа PCN містить три основні частини: кодувальну (кодувальник) *Encoder*, що здійснює глобальне подання вхідної неповної точкової хмари; мережу загального відтворення *Mapping Network*, що генерує грубу хмару точок за результатами глобального подання; декодувальну частину (декодувальник) *Decoder*, що уточнює та підвищує роздільну здатність грубої хмари точок для отримання доповненої графової моделі.

В архітектурі запропонованої графової мережі збережено загальну структуру PCN

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

з адаптацією її для оброблення графовими даними за допомогою кодувальника *Graph Encoder* (замість стандартного кодувальника) (рис. 5).

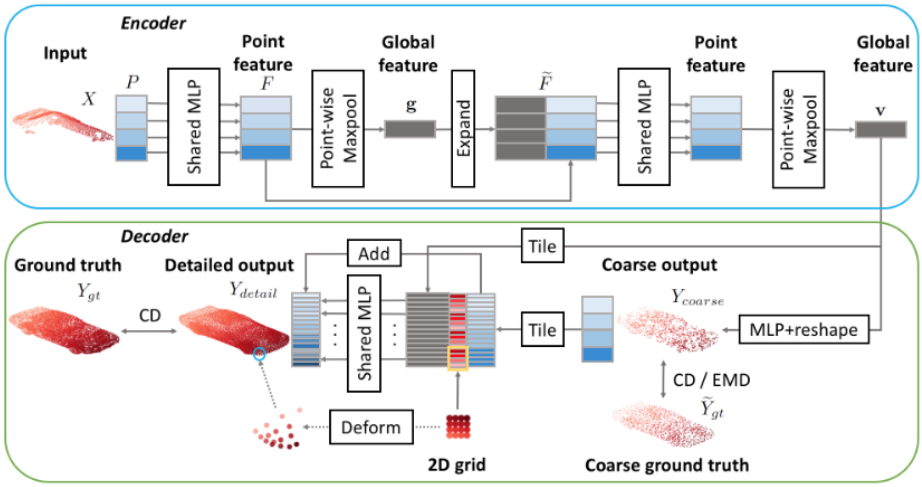


Рис. 4. Архітектура мережі доповнення точок PCN

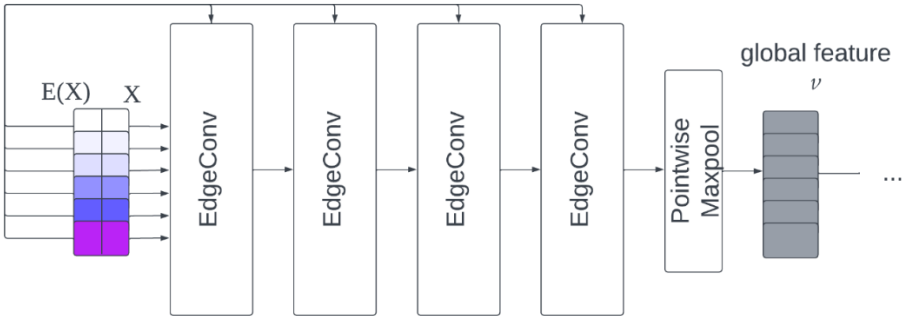


Рис. 5. Структура кодувальної частини запропонованої мережі

Основна розбіжність між кодувальником *Graph Encoder* і стандартною кодувальною частиною в мережі PCN полягає в тому, що:

– крім звичайного набору точок, кодувальник *Graph Encoder* приймає також значення індексів у побудованому графі цих точок;

– замість одновимірних операцій згортки в кодувальнику *Graph Encoder* виконуються операції графової згортки *EdgeConv*, запропоновані в роботі [40].

На відміну від операцій згортки в базових конфігураціях GCN в операціях графової згортки *EdgeConv* до зваженої агрегації даних щодо вершин графа додаються значення ребер, які з'єднують ці вершини:

$$x_i' = \sum_{j \in NN_i} h_\Theta \left(x_i \,||\, x_i - x_j \right), \qquad (18)$$

де NN_i – найближчі сусіди вершини i; h_Θ – нелінійна функція з параметрами Θ.

На рис. 6 наведено приклад візуальної інтерпретації операції згортки *EdgeConv*.

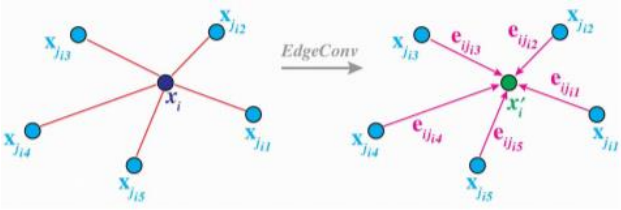


Рис. 6. Візуальна інтерпретація операції згортки *EdgeConv*

Розроблена нейромережа містить кілька основних компонентів: кодувальник *Graph Encoder*, мережа загального відтворення *Mapping Network* і декодувальник *Decoder*.

Кодувальник *Graph Encoder* є ключовим елементом запропонованої модифікації нейронної мережі, відповідальним за формування глобальної репрезентації вхідного графа. Цей компонент містить шари згортки *EdgeConv*, що застосовуються

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32) ISSN 2524-2296 (online)

до вхідних даних для поступового розширення рецептивного поля та отримання інформації із сусідніх вершин. Під час такого розширення ознаки вершин та індекси графових ребер передаються на кожен шар для здійснення операції згортки та оновлення ознак вершин. Оновлені ознаки вершин використовуються для глобальної максимальної субдискретизації, що формує глобальне подання обробленого графа, зважаючи на дані з окремих його частин.

Глобальне подання графа отримують за допомогою *Graph Encoder*, що бере до уваги як локальні ознаки сусідів кожної вершини, так і загальну структуру графа. Аналогічне подання використовують інші компоненти мережі, зокрема *Mapping Network* і *Decoder*, для генерування точкової хмари з огляду на доповнення.

Застосування кодувальника *Graph Encoder* із шарами згортки *EdgeConvLayer* забезпечує ефективне оброблення неструктурованих даних, поданих у вигляді графа, та дає змогу отримувати найбільш інформативні ознаки для реалізації завдань графової реконструкції зображень.

Mapping Network є компонентом нейронної мережі для доповнення неповних 3D точкових хмар, що відповідає за перетворення глобального подання графа, отриманого з *Graph Encoder*, у грубу точкову хмару (*coarse point cloud*). Груба точкова хмара є проміжним поданням, що містить меншу кількість точок порівняно з остаточною доповненою точковою хмарою, але зберігає загальну структуру та форму об’єкта.

Послідовність повнозв’язних шарів *Mapping Network* виконує лінійні перетворення із застосуванням стандартних функцій активації (зазвичай *ReLU*). Груба точкова хмара, сформована *Mapping Network*, містить меншу кількість точок, якщо порівнювати з остаточною доповненою хмарою, що дає змогу зменшити обчислювальні витрати та вимоги до пам’яті для подальшого оброблення. Втім, глобальна репрезентація графа, здійснена за допомогою кодувальника *Graph Encoder*, дає змогу грубій точковій хмарі зберігати основні структурні властивості об’єкта.

Після формування грубої точкової хмари вона передається до фінального компонента *Decoder*, що збільшує кількість точок для остаточного формування нейронною мережею уточненої хмари з високою деталізацією. Згорткові шари цього декодера виконують операції згортки вздовж ознак для кожної точки окремо (не беручи уваги інформацію від сусідніх точок). Після кожного шару згортки в декодері здійснюються операції пакетної нормалізації та активації, а також оновлюються ознаки точок із збереженням інформації з попередніх шарів.

Процес генерації уточненої точкової хмари в модулі *Decoder* ґрунтується на методі *folding*, запропонованому в дослідженні [41]. Основна ідея цього методу полягає у створенні регулярної сітки опорних точок (*folding seed*) з подальшою її деформацією за допомогою згорткових шарів для отримання остаточної точкової хмари. Такий підхід забезпечує генерацію точок із високою роздільною здатністю, зберігаючи водночас топологічні властивості об’єкта. Модуль *Decoder* відіграє центральну роль у відновленні деталізованої та повної точкової хмари на основі її початкової грубої версії. Використання згорткових шарів у поєднанні з методом *folding* дає змогу ефективно уточнювати та розширювати набір точок, зберігаючи в цьому разі геометричні особливості зображення.

Для навчання та оцінювання продуктивності нейронної мережі *GraphPointCompletionNetwork* доцільно використовувати набір даних *ShapeNet* [42]. *ShapeNet* є великомасштабним репозиторієм тривимірних моделей об’єктів, що охоплює різні категорії, зокрема літаки й автомобілі. Моделі цього набору подані сукупностями дискретних точок у тривимірному просторі, які описують геометричну структуру зображень.

Для моделювання запропонованого методу набір даних *ShapeNet* було спеціально підготовлено та поділено на два складники:

– неповні версії точкових хмар з набору *ShapeNet*, що були вхідною інформацією для навчання нейронної мережі;

– точки, що були штучно вилучені з оригінальних точкових хмар з метою створення неповних версій. Ці дані застосовуються для оцінювання якості подальшої реконструкції точкових хмар.

Для формування неповних точкових хмар з набору *ShapeNet* був розроблений спеціалізований скрипт, що реалізує такий алгоритм:

– завантаження оригінальних точкових хмар із файлів у форматі PLY;

– вибірка фіксованої кількості точок із кожної хмари для уніфікації розміру вибірки;

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

– формування неповних точкових хмар із застосуванням поділу вихідної хмари на дві частини за допомогою роздільної площини: точки з одного боку площини класифікуються як наявні, тоді як точки з другого боку – як відсутні;

– збереження отриманих неповних версій точкових хмар і вилучених точок у форматі PLY.

Для розділення точкової хмари використовується випадково згенерована гіперплощина в тривимірному просторі, що визначається нормальним вектором та зміщенням. Це дає змогу чітко визначити, по який бік площини розташована кожна точка. Крім того, було реалізовано функцію, що виконує розподіл отриманих неповних точкових хмар на тренувальну та валідаційну вибірки. Розподіл здійснюється випадковим чином відповідно до заданих пропорцій, а результати зберігаються у вигляді списків шляхів до відповідних файлів, що забезпечує можливість відтворення експерименту. Для підвищення варіативності навчальної вибірки було прийнято рішення розширити обсяг згенерованої інформації. Зокрема для кожного об’єкта створюється чотири різні гіперплощини, що дає змогу отримати різні варіанти його розділення. Такий підхід сприяє збільшенню різноманітності навчальних даних і покращенню узагальнювальної здатності моделі.

На початковому етапі моделювання запропонованого методу досліджувалися лише набори даних, що належать до класу літаків. Це допомогло ретельно опрацювати всі необхідні екземпляри та сформувати набір даних, що містить об’єкти єдиної категорії. Отже, підготовлений набір *ShapeNet* разом зі спеціалізованим скриптом для генерації неповних версій точкових хмар забезпечує ефективну підготовку інформації для навчання та оцінювання працездатності нейронної мережі *Graph Point Completion Network*. Використання випадкового поділу точкових хмар та розпаралелювання операції генерації уможливлюють репрезентативну множину варіантів неповних точкових хмар, що сприяє підвищенню якості навчання моделі.

Розглянемо метод навчання запропонованої графової нейронної мережі для завдання реконструкції 3D-моделей точкових хмар. Процес навчання передбачає послідовне виконання таких етапів:

– ініціалізація моделі нейронної мережі (а також параметрів оптимізатора, функції втрат і планувальника швидкості навчання);

– завантаження наборів навчальних і валідаційних даних, поданих у вигляді точкових хмар;

– аналіз наявних результатів навчання аналогічної моделі в минулому (якщо збережені результати існують, відбувається відновлення найкращих значень функції втрат і відповідної епохи навчання; завантажуються історичні значення функцій втрат для навчального й валідаційного наборів даних; відновлюється стан моделі, оптимізатора та планувальника навчання способом завантаження збережених ваг).

Навчання моделі здійснюється протягом визначеної кількості епох, кожна з яких передбачає такі основні етапи:

– прохід по навчальному набору даних, згідно з яким визначаються грубі прогнози з подальшою деталізацією;

– обчислення функцій втрат для грубого й деталізованого прогнозів, а також загальної функції втрат;

– обчислення градієнтів (на основі значення загальної функції втрат) з подальшим оновленням коефіцієнтів моделі та параметрів оптимізатора;

– прохід по валідаційному набору даних, згідно з яким (за аналогією з проходом по навчальному набору даних) обчислюються прогнозні значення та значення загальної функції втрат; на цьому етапі оновлення ваг не відбувається, оскільки метою є лише оцінювання якості моделі;

– обчислення та логування навчальних метрик для поточної епохи;

– збереження поточного стану навчання через задані проміжки часу (зокрема станів моделі, оптимізатора, планувальника швидкості навчання, а також історії значень функцій втрат);

– збереження прикладів реконструйованих точкових хмар для подальшої візуалізації навчання;

– передача збережених результатів (реконструйованих хмар точок і навчальних метрик) до відповідної бази даних;

– завершення навчання: після досягнення заданої кількості епох зберігається найкращий стан моделі для подальшого використання у завдання доповнення точкових хмар.

Такий підхід забезпечує ефективне налаштування параметрів графової нейронної мережі для доповнення точкових хмар, даючи змогу відстежувати динаміку навчання, зберігати проміжні результати та візуалізувати реконструйовані хмари. Використання

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

планувальника швидкості навчання (наприклад, *StepLR*) додатково підвищує ефективність навчального процесу: на початкових етапах застосовується більш висока швидкість навчання для швидкої оптимізації параметрів; на подальших етапах швидкість навчання поступово знижується, що дає змогу моделі точніше налаштуватися поблизу глобального оптимуму та уникнути небажаних коливань.

Такий механізм сприяє кращій збіжності та допомагає уникнути зупинок алгоритму навчання в локальних оптимумах. Окрім цього, можливість використання попередніх результатів навчального процесу забезпечує економію обчислювальних ресурсів, оскільки дає змогу продовжити навчання з найкращого проміжного стану, уникаючи необхідності повторювати з нуля. Це значно зменшує витрати часу та ресурсів на досягнення оптимальних результатів.

Програмна реалізація запропонованого методу

Для програмної реалізації запропонованого методу використано такі основні технології: *Python, PyTorch* та *PyTorch Geometric*. Застосування цих інструментів зумовлено їх ефективністю, гнучкістю та поширенням у сфері глибокого навчання та оброблення інформації. *Python* – одна з найбільш популярних мов програмування, активно впроваджена в наукових обчисленнях, аналізі даних та завданнях машинного навчання завдяки простому та зрозумілому синтаксису, а також наявності значної кількості спеціалізованих бібліотек.

PyTorch – це відкрита бібліотека, яка забезпечує гнучкий та інтуїтивно зрозумілий інтерфейс для побудови й навчання нейронних мереж, підтримує автоматичне диференціювання та ефективно використовує обчислювальні ресурси, зокрема GPU та TPU. Завдяки можливості динамічного визначення обчислювального графа *PyTorch* є зручним інструментом для дослідницьких і експериментальних проєктів, а його велика спільнота користувачів сприяє активному розвитку та підтримці бібліотеки.

PyTorch Geometric – це спеціалізована бібліотека для роботи з графовими структурами в середовищі *PyTorch*, яка містить набір оптимізованих інструментів для завантаження, оброблення й трансляції графових даних у нейронні мережі, що дає змогу реалізовувати різні архітектури графових нейронних мереж (GNNs). *PyTorch Geometric* широко застосовується в завданнях оброблення графових структур, зокрема в 3D комп'ютерному зорі, моделюванні фізичних процесів, аналізі соціальних і біологічних мереж. Використання зазначених технологій допомагає ефективно реалізувати запропонований метод, забезпечуючи високу продуктивність, гнучкість у налаштуванні моделей та можливість масштабування обчислень.

Розглянемо особливості програмної реалізації основних компонентів графової мережі, що дають змогу виконувати окремі етапи запропонованого методу.

Одним з основних будівельних блоків кодувальної частини мережі для реконструкції неповних 3D точкових хмар є шар *EdgeConvLayer*, який реалізує операцію *EdgeConv*. Ця операція, запропонована в роботі [1], є локальним перетворенням на графових моделях, що дає змогу агрегувати інформацію з сусідніх вершин для оновлення ознак поточної вершини. Шар *EdgeConvLayer* приймає два параметри: *in_channels*, що визначає кількість ознак вершин, та *out_channels*, що задає розмірність ознак після застосування шару *EdgeConv*. У межах цього шару створюється екземпляр класу *EdgeConv* з бібліотеки *torch_geometric*. Для нелінійного перетворення h_Θ використовується лінійне відтворення, реалізоване за допомогою шару *nn.Linear*, що застосовується до ознак кожної вершини та її сусідів.

Упровадження методу *forward* дає змогу реалізувати операцію *EdgeConv*, що на основі вхідних даних формує новий тензор ознак вершин розміру *(num_nodes, out_channels)*, у якому кожна вершина містить оновлену інформацію, отриману від її сусідів.

Шар *EdgeConvLayer* є ключовим компонентом кодувальної частини графової нейронної мережі, оскільки забезпечує агрегацію локальної інформації із сусідніх вершин і оновлення їх ознак з огляду на структуру графа. Використання кількох послідовних шарів *EdgeConvLayer* дає змогу збільшувати рецептивне поле й отримувати більш виразні ознаки для кожної вершини, що підвищує ефективність навчання моделі.

Кодувальник *Graph Encoder* використовується для доповнення неповних 3D точкових хмар і отримання глобального подання вхідного графа. Його структура передбачає послідовності шарів *EdgeConvLayer*, що поетапно обробляють вхідні дані, збільшуючи рецептивне поле й агрегуючи інформацію із сусідніх вершин.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

Graph Encoder приймає один параметр *blocks*, що є списком цілих чисел, які визначають кількість каналів (розмірність ознак) на кожному рівні *EdgeConvLayer*.

Під час ініціалізації кодувальника *Graph Encoder* зберігається список *blocks* як атрибут класу, а також викликається метод *_build_encoder*, що створює послідовність шарів *EdgeConvLayer* згідно із заданою архітектурою. Метод *_build_encoder* приймає список *blocks*, де формується екземпляр класу *EdgeConvLayer* з відповідною кількістю вхідних і вихідних каналів, забезпечуючи ефективну побудову глибокої графової нейронної мережі.

Метод *forward* призначений для реалізації проходу крізь кодувальну мережу графа. Він використовує такі аргументи: вхідний тензор ознак вершин розміру; тензор індексів ребер розміру; тензор, що містить інформацію про належність вершин до різних графів у разі оброблення пакетів графів. Відповідно до цього методу вхідний тензор послідовно проходить крізь шари *EdgeConvLayer* кодувальника. У цьому разі шари оновлюють ознаки вершин, зважаючи на інформацію від їх сусідів. Потім застосовується операція об'єднання інформації з усіх вершин графа й формування глобального подання графа.

Подання вихідного графа, отримане за допомогою кодувальника *Graph Encoder*, є компактним і репрезентативним. Надалі це подання використовується для генерації доповненої точкової хмари іншими компонентами графової нейронної мережі, зокрема мережею відтворення *Mapping Network* та декодувальником *Decoder*. Кодувальник *Graph Encoder* з набором шарів *EdgeConvLayer* дає змогу ефективно обробляти неструктуровані дані графової моделі, що робить його зручним інструментом для аналізу та оброблення 3D точкових хмар.

У компоненті *Mapping Network*, призначеному для побудови грубих точкових хмар, використовуються такі аргументи: розмірність вхідного латентного подання графа; кількість повнозв'язних шарів у мережі; кількість точок у грубій точковій хмарі.

Ініціалізація *Mapping Network* передбачає виклик методу *_build_mappings* для створення послідовності повнозв'язних шарів на основі значень розмірності вхідного подання графа й кількості повнозв'язних шарів у мережі. Метод *_build_mappings* працює з послідовністю повнозв'язних шарів та функціями активації *ReLU*. Останній із шарів дає змогу генерувати координати точок грубої точкової хмари.

Груба точкова хмара, згенерована з використанням *Mapping Network*, має суттєво редуковану кількість точок порівняно з повною точковою хмарою, що сприяє зменшенню обчислювальної складності подальшого оброблення. Водночас глобальне подання графа, отримане згідно з кодувальником *Graph Encoder*, дає змогу грубій точковій хмарі зберігати загальну структуру та форму 3D-об'єкта. Сформована груба точкова хмара передається до компонента *Decoder*, що збільшує кількість точок для формування остаточної доповненої точкової хмари. Операція *Mapping Network* дає змогу перейти від початкового подання графа до просторового подання у вигляді точкової хмари. Це забезпечує ефективність і якість доповнення точкових хмар на подальших етапах оброблення інформації. Функції декодувальника для нейронної мережі реалізує компонент *Decoder*, що безпосередньо доповнює неповну 3D точкову хмару. Він використовує такі атрибути: список каналів у кожному блоці; розмір сітки *grid_size* (за замовчуванням 4) та масштаб сітки *grid_scale* (за замовчуванням 0.05). Ці атрибути ініціалізуються конструктором класу *init*, що також викликає метод *_build_decoder*, який створює необхідний список блокових модулів. Кожен блок списку містить згортковий шар, шар нормалізації батча й активаційної функції *ReLU*. Процес проходження даних крізь декодувальник реалізується з використанням методу *forward*, що приймає вхідну точкову хмару, а потім формує фрагменти доповненого подання точкової хмари, які застосовуються для остаточного доповнення неповних 3D точкових хмар. Тобто він відповідає за перетворення грубої точкової хмари на більш детальну точкову хмару.

Ініціалізація повної версії запропонованої графової мережі *Graph Point Completion Network*, що використовується для доповнення неповних 3D точкових хмар. Ключовими компонентами цієї мережі є кодувальна мережа *Graph Encoder*, мережа відтворення *Mapping Network* і декодувальна мережа *Decoder*.

Ініціалізація передбачає необхідність налаштування компонентів *Graph Encoder*, *Mapping Network*, *Decoder* та обчислення тензору *folding_seed*, що надалі використовується для згортання грубої точкової хмари до остаточної точкової хмари. Програмний код ініціалізації повної версії запропонованої мережі наведено на рис. 7.

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

```
class GraphPointCompletionNetwork(nn.Module):
   def __init__(self, cfg: dict):
      super(GraphPointCompletionNetwork, self).__init__()
   self.num_dense: int = cfg.get('num_dense', 16384)
   self.grid_size: int = cfg.get('grid_size', 4)
   self.grid_scale: int = cfg.get('grid_scale', 0.05)
   self.num_coarse = self.num_dense//(self.grid_size**2)
   self.encoder = GraphEncoder(**cfg['encoder'])
   self.mapping_network = MappingNetwork( **cfg ['mapping_network'], num_coarse = self.num_coarse)
   self.decoder = Decoder(**cfg ['decoder'], grid_size = self.grid_size, grid_scale = self.grid_scale)
   a = (torch.linspace(-self.grid_scale, self.grid_scale, steps = self.grid_size, dtype=torch.float).view(1, self.grid_size).expand(self.grid_size, self.grid_size) .reshape(1, -1))
   b = (torch.linspace(-self.grid_scale, self.grid_scale, steps=self.grid_size, dtype=torch.float).view(self.grid_size, 1).expand(self.grid_size, self.grid_size).reshape(1, -1))
      self.folding_seed = torch.cat([a, b], dim=0).view(1, 2, self.grid_size ** 2)  # (1, 2, S)
```

Рис. 7. Програмний код ініціалізації повної версії запропонованої мережі

Операція ініціалізації забезпечує налаштування функціональності ключових компонентів *Graph Encoder*, *Mapping Network* і *Decoder* для отримання остаточної доповненої точкової хмари на основі вхідної неповної точкової хмари. Для формування тестувального набору даних з метою налаштування розробленої мережі застосовуватимемо клас *GraphShapeNet*, який є підкласом *Dataset* і містить набір даних *ShapeNet* у графовому поданні. Клас *GraphShapeNet* призначений для тестування завдань реконструкції зображень, пов'язаних з обробленням неповних 3D точкових хмар з використанням графових нейронних мереж.

Клас *GraphShapeNet*, що є сумісним із функціональністю класу *ShapeNet*, додатково будує за методом KNN граф k найближчих сусідів $(k=15)$ для наявної точкової хмари, що зберігається в атрибуті об'єкта *Data*. Повна точкова хмара цього об'єкта містить відповідний атрибут y. До основних фрагментів коду, який реалізує запропонований підхід, належить механізм навчання побудованої нейромережі. Цей механізм передбачає реалізацію таких кроків: ініціалізація даних із використанням відповідних обчислювальних засобів; формування фрагментів відтворень 3D-моделей; оброблення нейромережею поданих фрагментів відтворень 3D-моделей; порівняння грубих і повних відтворень 3D-моделей; поєднання відтворень 3D-моделей (за результатами порівняння в єдину доповнену модель). Програмний код механізму навчання графової мережі наведено на рис. 8.

Цей механізм навчання оптимізується із застосуванням оптимізатора *Adam*.

```
ground_truth = data.y.to(device)
ground_truth = ground_truth.view(batch_size, -1, 3)
coarse_predicted, dense_predicted = model(data.to(device))
num_coarse_points = coarse_predicted.shape[1]
coarse_predicted = coarse_predicted.view(batch_size,-1, 3)
dense_predicted = dense_predicted.view(batch_size, -1, 3)
coarse_loss = loss_fn(coarse_predicted, ground_truth[:,:num_coarse_points, :] )
dense_loss = loss_fn(dense_predicted, ground_truth)
loss = coarse_loss + loss_multiplier * dense_loss
```

Рис. 8. Програмний код механізму навчання графової мережі

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

Моделювання запропонованого методу для завдань доповнення та 3D-реконструкції моделей точкових хмар

Моделювання запропонованого методу здійснювалося для реконструкції хмар точок на прикладі класу об'єктів "літак" (*airplane*) з набору даних *ShapeNet* [42]. Навчання моделі здійснювалось протягом 340 епох із використанням таких налаштувань:

– оптимізатор *Adam* з параметрами $l_r = 0.0001$, $weight\_decay = 0.0001$, $betas = [0.9,\ 0.999]$;

– планувальник швидкості навчання *StepLR* з параметрами $step\_size = 40$, $gamma = 0.8$;

– функція втрат *Chamfer Loss*.

Для тренування було використано хмарне середовище з графічним процесором *Nvidia RTX* 3090 *Ti* (24 GB) та 48 GB RAM. Планований час тренування становив 116 год, загальний час тренування – 30 год. На рис. 9 наведено логарифмічно масштабовані значення функцій втрат для навчальної та валідаційної вибірок залежно від кількості епох.

Результати навчання продемонстрували значення функції втрат 25.11 на навчальній вибірці та 30.21 на валідаційній вибірці. Різниця між значеннями функції втрат на навчальній та валідаційній вибірках не є дуже великою, що свідчить про відсутність значного перенавчання моделі. Відповідно до графіка (рис. 9) можна стверджувати, що, починаючи з 200-ї епохи, нейромережна мережа досягла своїх можливостей навчання на застосованому наборі даних.

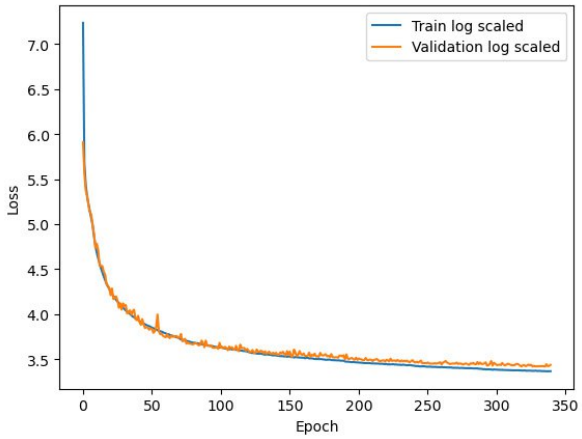
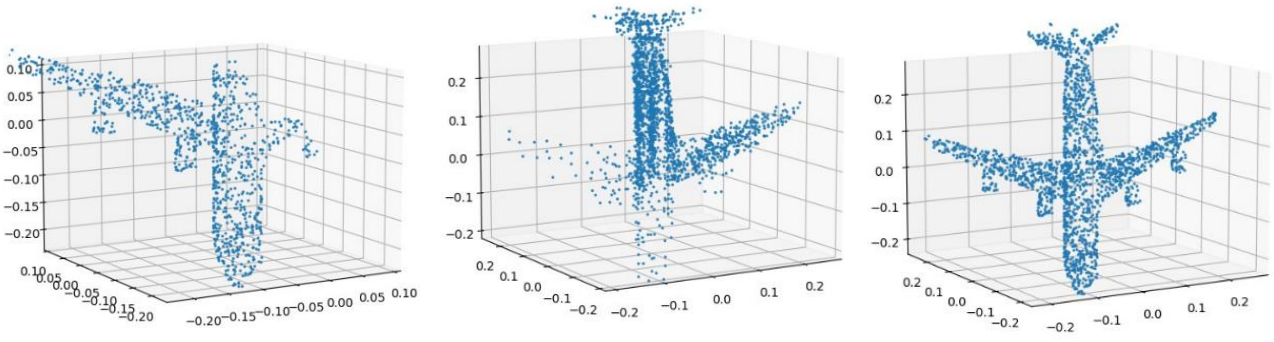


Рис. 9. Логарифмічно масштабовані значення функцій втрат навчальної та валідаційної вибірок

Стабільному процесу навчання сприяло використання оптимізатора *Adam*. Крім того, застосування планувальника швидкості навчання *StepLR* дало змогу регулювати швидкість навчання для покращення збіжності моделі. На рис. 10 наведено приклади реконструйованих хмар точок, вхідних даних та еталонних хмар точок.



a *в* *с*

Рис. 10. Візуалізація вхідних даних (а), відновлених фрагментів моделі (в) та повної реконструкції моделі (с)

Відновлена 3D-модель літака демонструє задовільну якість доповнення порівняно з початковою неповною моделлю. Основні структурні елементи, зокрема крила та хвостова частина, були відновлені. Водночас відновлена модель має певні недоліки в деталях. Наприклад, бажаною є більш висока деталізація невеликих або дрібних елементів об'єкта. Крім того, спроба відновити менш поширені моделі літальних апаратів (з відомих датасетів) інколи призводила до неякісних результатів доповнення (рис. 11).

Якщо порівняти отримані результати доповнення з результатами деяких інших робіт [41], [42], можна помітити відсутність чіткої деталізації дрібних елементів за рівних умов навчання та формування даних (рис. 12).

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

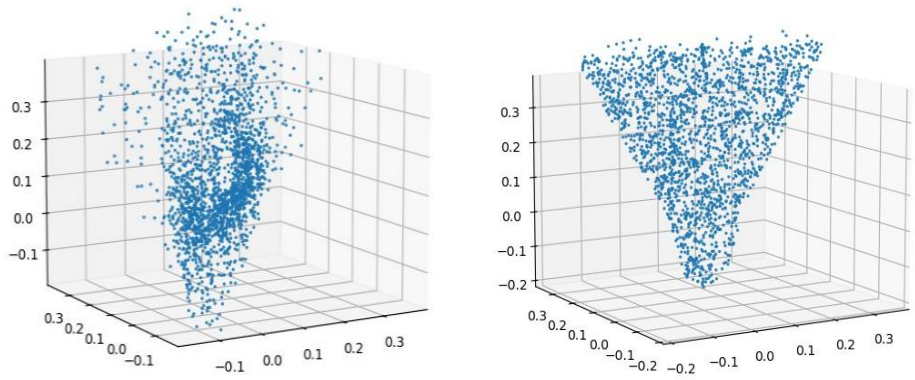
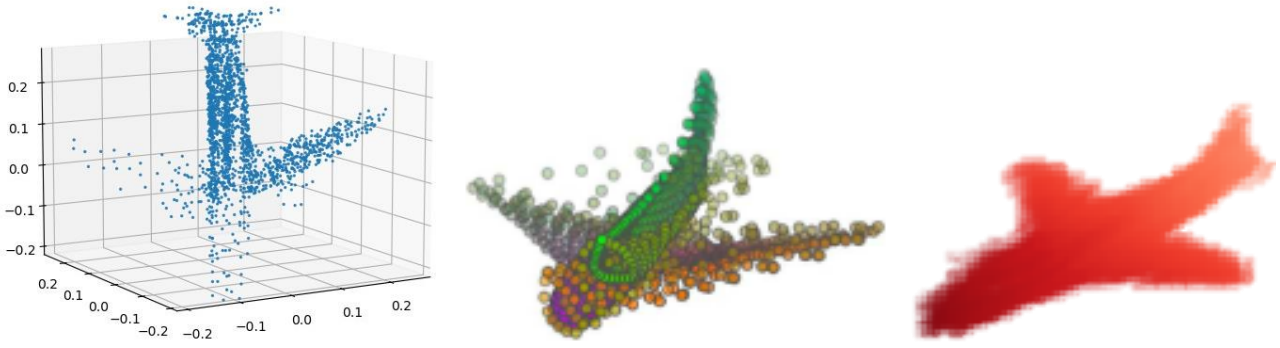


Рис. 11. Візуалізація ефекту недостатньої деталізації на малопоширеній 3D-моделі літака



a *b* *c*

Рис. 12. Візуальне порівняння отриманих результатів для пропонованої мережі (*a*), мережі *FoldingNet* (*b*) та мережі *Point Completion Network* (*c*)

Важливо наголосити, що деякі сучасні модифікації мереж (зокрема *SnowflakeNet*) дають змогу інколи відновлювати об’єкти з поліпшеною деталізацію графічного подання об’єкта (рис. 13).

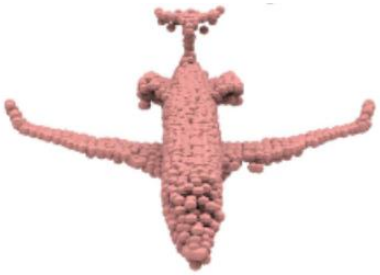


Рис. 13. Відновлення моделі літака за допомогою мережі *SnowflakeNet*

Утім, розроблений метод є першим, що призначений для розв’язання проблеми доповнення неповних 3D точкових хмар з безпосереднім використанням графових нейронних мереж без попереднього оброблення іншими засобами. Зважаючи на це, отримані результати можна вважати перспективними для досягнення поставленої в роботі мети. Зазначимо, що графове подання точкової хмари може покращити здатність моделі виявляти та вивчати структурні залежності між точками (це може бути корисним для завдання доповнення неповних точкових хмар).

З метою покращення подальших результатів доцільним є дослідження можливості інших архітектур кодувальних мереж, які дали б змогу ефективніше частково доповнити запропоновану архітектуру графової моделі. До того ж перспективним є поліпшення алгоритмів генерації самих точкових хмар (наприклад, алгоритмів подвійного спуску [43]) для підвищення якості відтвореної сцени й реконструкції дрібних деталізованих компонентів.

Більш об’єктивно оцінити узагальнювальну здатність запропонованого методу можна за результатами проведення додаткових експериментів з іншими класами об’єктів і наборів даних. Крім того, використання потужних обчислювальних ресурсів (зокрема GPU з більшим обсягом пам’яті) може сприяти навчанню більш глибоких графових нейронних моделей, що потенційно може поліпшити якість доповнення 3D-моделей хмар точок.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)

Висновки

У роботі запропоновано метод застосування графових нейронних мереж для доповнення 3D-моделі хмар точок.

Подане рішення для відновлення неповних 3D точкових хмар визначається науковою новизною та поєднує потужність графових нейронних мереж (GNN) з архітектурою мережі *Point Completion Network* (PCN). Це є важливим для багатьох застосувань, таких як 3D-реконструкція, навігація роботів тощо.

Автоматичне доповнення даних хмари точок надає потенціал для підвищення безпеки, надійності та продуктивності критично важливих систем, що покладаються на достовірну 3D-інформацію про навколишнє середовище.

Практичну значущість роботи підтверджують результати моделювання розробленого методу для класичних датасетів та їх порівняння з деякими наявними підходами до розв'язання досліджуваної проблеми.

Результати моделювання продемонстрували прийнятні значення функції втрат на навчальній та валідаційній вибірках, що свідчить про відсутність суттєвого ефекту перенавчання моделі.

Аналіз графіків функцій втрат і візуальне оцінювання реконструйованих хмар точок показали, що модель змогла відновити відсутні структурні елементи. Водночас спостерігалась недостатньо якісна деталізація дрібних елементів, а також низька спроможність доповнення для менш поширених моделей літаків.

Порівняння з іншими роботами в галузі доповнення хмар точок виявило подібні недоліки за аналогічних умов навчання та генерації даних. Це визначає необхідність подальшого вдосконалення архітектур графових нейронних мереж і методів навчання для підвищення якості відновлення 3D-моделей хмар точок.

Подальші дослідження доцільно зосередити на аналізі можливостей інших архітектур графових нейронних мереж, налаштуванні гіперпараметрів і застосуванні більш потужних обчислювальних ресурсів для навчання аналізованих моделей. Додаткові експерименти з іншими класами об'єктів та наборів даних також можуть допомогти більш об'єктивно оцінити узагальнювальну здатність запропонованого методу.

Загалом отримані результати демонструють потенціал графових нейронних мереж для ефективного розв'язання завдання доповнення та відновлення 3D-моделей хмар точок.

Список літератури

1. Taylor T. S. Introduction to Laser Science and Engineering. First edition. Boca Raton, FL: CRC Press/Taylor & Francis Group, 2020: *CRC Press*, 2019. DOI: https://doi.org/10.1201/b22159
2. Data-driven structural priors for shape completion / M. Sung et al. *ACM Transactions on Graphics*. 2015. Vol. 34, No. 6. P. 1–11. DOI: https://doi.org/10.1145/2816795.2818094
3. "State of the Art in Surface Reconstruction from Point Clouds / M. Berger et al. DSpace Repository". URL: https://diglib.eg.org/items/d9bfd0be-9d3e-4ced-8d25-ef522dc454f3 (дата звернення: 14.05.2024).
4. A Field Model for Repairing 3D Shapes / D. T. Nguyen et al. 2016 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016. DOI: https://doi.org/10.1109/cvpr.2016.612
5. Fu Z., Hu W., Guo Z. Point Cloud Inpainting on Graphs from Non-Local Self-Similarity. 2018 25th *IEEE International Conference on Image Processing (ICIP)*, Athens, 7–10 October 2018. DOI: https://doi.org/10.1109/icip.2018.8451550
6. Mitra N. J., Guibas L. J., Pauly M. Partial and approximate symmetry detection for 3D geometry. *ACM Transactions on Graphics*. 2006. Vol. 25, No. 3. P. 560–568. DOI: https://doi.org/10.1145/1141911.1141924
7. Discovering structural regularity in 3D geometry / M. Pauly et al. *ACM Transactions on Graphics*. 2008. Vol. 27, No. 3. P. 1–11. DOI: https://doi.org/10.1145/1360612.1360642
8. An interactive approach to semantic modeling of indoor scenes with an RGBD camera / T. Shao et al. ACM Transactions on Graphics. 2012. Vol. 31, No. 6. P. 1–11. DOI: https://doi.org/10.1145/2366145.2366155
9. Structure recovery by part assembly / C.-H. Shen et al. ACM Transactions on Graphics. 2012. Vol. 31, No. 6. P. 1–11. DOI: https://doi.org/10.1145/2366145.2366199
10. A probabilistic model for component-based shape synthesis / E. Kalogerakis et al. *ACM Transactions on Graphics*. 2012. Vol. 31, No. 4. P. 1–11. DOI: https://doi.org/10.1145/2185520.2185551
11. Chauve A.-L., Labatut P., Pons J.-P. Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. 2010 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, 13–18 June 2010. DOI: https://doi.org/10.1109/cvpr.2010.5539824

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

12. Completing 3D object shape from one depth image / J. Rock et al. 2015 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 7–12 June 2015. DOI: https://doi.org/10.1109/cvpr.2015.7298863
13. Vinyals O., Fortunato M., Jaitly N. Pointer Networks. *Advances in Neural Information Processing Systems*. 2015. DOI: https://doi.org/10.48550/arXiv.1506.03134
14. A Papier-Mache Approach to Learning 3D Surface Generation / T. Groueix et al. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–23 June 2018. DOI: https://doi.org/10.1109/cvpr.2018.00030
15. Dai A., Qi C. R., NieBner M. Shape Completion Using 3D-Encoder-Predictor CNNs and Shape Synthesis. 2017 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 21–26 July 2017. DOI: https://doi.org/10.1109/cvpr.2017.693
16. Wang X., Ang M. H., Hee Lee G. Voxel-based Network for Shape Completion by Leveraging Edge Generation. 2021 *IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 10–17 October 2021. 2021. DOI: https://doi.org/10.1109/iccv48922.2021.01294
17. PCN: Point Completion Network / W. Yuan et al. 2018 *International Conference on 3D Vision (3DV)*, Verona, 5–8 September 2018. DOI: https://doi.org/10.1109/3dv.2018.00088
18. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation / R. Q. Charles et al. 2017 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 21–26 July 2017. DOI: https://doi.org/10.1109/cvpr.2017.16
19. Morphing and Sampling Network for Dense Point Cloud Completion / M. Liu et al. *Proceedings of the AAAI Conference on Artificial Intelligence.* 2020. Vol. 34, No. 07. P. 11596–11603. DOI: https://doi.org/10.1609/aaai.v34i07.6827
20. Lawrance N. R. J., Chung J. J., Hollinger G. A. Fast Marching Adaptive Sampling. *IEEE Robotics and Automation Letters*. 2017. Vol. 2, No. 2. P. 696–703. DOI: https://doi.org/10.1109/lra.2017.2651148
21. Chen X., Chen B., Mitra N. J. Unpaired Point Cloud Completion on Real Scans using Adversarial Training. *International Conference On Learning Representations* (ICLR 2020). DOI: https://doi.org/10.48550/arXiv.1904.00069
22. "Generative Adversarial Networks / I. J. Goodfellow et al. Advances in Neural Information Processing Systems". 2014. URL: https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf. (дата звернення: 15.05.2024).
23. PF-Net: Point Fractal Network for 3D Point Cloud Completion / Z. Huang et al. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020. DOI: https://doi.org/10.1109/cvpr42600.2020.00768
24. Refinement of Predicted Missing Parts Enhance Point Cloud Completion / A. Mendoza et al. *Computer Vision and Pattern Recognition.* 11 p. 2020. DOI: https://doi.org/10.48550/arXiv.2010.04278
25. HyperPocket: Generative Point Cloud Completion / P. Spurek et al. 2022 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, Japan, 23–27 October 2022. 2022. DOI: https://doi.org/10.1109/iros47612.2022.9981829
26. Ha D., Dai A., Le Q. V. Hypernetworks. International Conference on Learning Representa- tions. *Machine Learning.* 2022. DOI: https://doi.org/10.48550/arXiv.1609.09106
27. SnowflakeNet: Point Cloud Completion by Snowflake Point Deconvolution with Skip-Transformer / P. Xiang et al. 2021 *IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 10–17 October 2021. 2021. DOI: https://doi.org/10.1109/iccv48922.2021.00545
28. Pan L. ECG: Edge-aware Point Cloud Completion with Graph Convolution. *IEEE Robotics and Automation Letters.* 2020. Vol. 5, No. 3. P. 4392–4398. DOI: https://doi.org/10.1109/lra.2020.2994483
29. Kullback S., Leibler R. A. On Information and Sufficiency. *The Annals of Mathematical Statistics.* 1951. Vol. 22, No. 1. P. 79–86. DOI: https://doi.org/10.1214/aoms/1177729694
30. Graph Convolutional Networks with Markov Random Field Reasoning for Social Spammer Detection / Y. Wu et al. *Proceedings of the AAAI Conference on Artificial Intelligence.* 2020. Vol. 34, No. 01. P. 1054–1061. DOI: https://doi.org/10.1609/aaai.v34i01.5455
31. Gradient-based learning applied to document recognition / Y. Lecun et al. *Proceedings of the IEEE.* 1998. Vol. 86, No. 11. P. 2278–2324. DOI: https://doi.org/10.1109/5.726791
32. LeCun Y., Bengio Y., Hinton G. Deep learning. Nature. 2015. Vol. 521, No. 7553. *PubMed*. P. 436–444. DOI: https://doi.org/10.1038/nature14539
33. What is a good medical decision? A research agenda guided by perspectives from multiple stakeholders / J. G. Hamilton et al. *Journal of Behavioral Medicine.* 2016. Vol. 40, No. 1. P. 52–68. DOI: https://doi.org/10.1007/s10865-016-9785-z
34. Efficient Estimation of Word Representations in Vector Space / T. Mikolov et al. *International Conference on Learning Representations*. 2013. DOI: https://doi.org/10.48550/arXiv.1301.3781
35. Perozzi B., Al-Rfou R., Skiena S. S. DeepWalk: online learning of social representations. KDD '14: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 20th ed. 2014. DOI: https://doi.org/10.1145/2623330.2623732
36. Mallat S. Wavelet Tour of Signal Processing. Elsevier Science & Technology Books, 1999.
37. Hammond D. K., Vandergheynst P., Gribonval R. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis.* 2011. Vol. 30, no. 2. P. 129–150. DOI: https://doi.org/10.1016/j.acha.2010.04.005

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)*

38. Kipf T. N., Welling M. Semi-Supervised Classification with Graph Convolutional Networks. *Machine Learning.* 2016. DOI: https://doi.org/10.48550/arXiv.1609.02907
39. Hamilton W. L., Ying R., Leskovec J. Inductive Representation Learning on Large Graphs. *Social and Information Networks.* 2017. DOI: https://doi.org/10.48550/arXiv.1706.02216
40. Dynamic Graph CNN for Learning on Point Clouds / Y. Wang et al. *ACM Transactions on Graphics.* 2019. Vol. 38, No. 5. P. 1–12. DOI: https://doi.org/10.1145/3326362
41. FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation / Y. Yang et al. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, 18–23 June 2018. DOI: https://doi.org/10.1109/cvpr.2018.00029
42. Orthogonal Dictionary Guided Shape Completion Network for Point Cloud / P. Cai et al. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2024. Vol. 38, No. 2. P. 864–872. DOI: https://doi.org/10.1609/aaai.v38i2.27845
43. Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. Deep double descent: Where bigger models and more data hurt. *Machine Learning*. 2019. DOI: https://doi.org/10.48550/arXiv.1912.02292

References

1. Taylor, T. S. (2020), "Introduction to Laser Science and Engineering". First edition. Boca Raton, FL: CRC Press/Taylor & Francis Group. DOI: https://doi.org/10.1201/b22159
2. Sung, M. (2015), "Data-driven structural priors for shape completion" / M. Sung et al. *ACM Transactions on Graphics*. 2015. Vol. 34, No. 6. P. 1–11. DOI: https://doi.org/10.1145/2816795.2818094
3. "State of the Art in Surface Reconstruction from Point Clouds" / M. Berger et al. DSpace Repository". URL: https://diglib.eg.org/items/d9bfd0be-9d3e-4ced-8d25-ef522dc454f3 (last accessed: 14.05.2024).
4. Nguyen, D. T. (2016), "A Field Model for Repairing 3D Shapes" / D. T. Nguyen et al. 2016 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016. DOI: https://doi.org/10.1109/cvpr.2016.612
5. Fu, Z., Hu, W., Guo, Z. (2018), "Point Cloud Inpainting on Graphs from Non-Local Self-Similarity". 2018 25th *IEEE International Conference on Image Processing (ICIP)*, Athens, 7–10 October 2018. DOI: https://doi.org/10.1109/icip.2018.8451550
6. Mitra, N. J., Guibas, L. J., Pauly, M. (2006), "Partial and approximate symmetry detection for 3D geometry". *ACM Transactions on Graphics*. 2006. Vol. 25, No. 3. P. 560–568. DOI: https://doi.org/10.1145/1141911.1141924
7. Pauly, M. (2008), "Discovering structural regularity in 3D geometry" / M. Pauly et al. *ACM Transactions on Graphics.* 2008. Vol. 27, No. 3. P. 1–11. DOI: https://doi.org/10.1145/1360612.1360642
8. Shao, T. (2012), "An interactive approach to semantic modeling of indoor scenes with an RGBD camera" / T. Shao et al. ACM Transactions on Graphics. 2012. Vol. 31, No. 6. P. 1–11. DOI: https://doi.org/10.1145/2366145.2366155
9. Shen, C.H. (2012), "Structure recovery by part assembly" / C.-H. Shen et al. ACM Transactions on Graphics. 2012. Vol. 31, No. 6. P. 1–11. DOI: https://doi.org/10.1145/2366145.2366199
10. Kalogerakis, E. (2012), "A probabilistic model for component-based shape synthesis" / E. Kalogerakis et al. *ACM Transactions on Graphics*. 2012. Vol. 31, No. 4. P. 1–11. DOI: https://doi.org/10.1145/2185520.2185551
11. Chauve, A.L., Labatut, P., Pons, J.P. (2010), "Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data". 2010 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, 13–18 June 2010. DOI: https://doi.org/10.1109/cvpr.2010.5539824
12. Rock, J. (2015), "Completing 3D object shape from one depth image" / J. Rock et al. 2015 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 7–12 June 2015. DOI: https://doi.org/10.1109/cvpr.2015.7298863
13. Vinyals, O., Fortunato, M., Jaitly,N. (2015), "Pointer Networks". *Advances in Neural Information Processing Systems*. 2015. DOI: https://doi.org/10.48550/arXiv.1506.03134
14. Groueix, T. (2018), "A Papier-Mache Approach to Learning 3D Surface Generation" / T. Groueix et al. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–23 June 2018. DOI: https://doi.org/10.1109/cvpr.2018.00030
15. Dai, A., Qi, C. R., NieBner, M. (2017), "Shape Completion Using 3D-Encoder-Predictor CNNs and Shape Synthesis". 2017 I*EEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 21–26 July 2017. DOI: https://doi.org/10.1109/cvpr.2017.693
16. Wang, X., Ang, M. H., Hee Lee, G. (2021), "Voxel-based Network for Shape Completion by Leveraging Edge Generation". 2021 *IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 10–17 October 2021. 2021. DOI: https://doi.org/10.1109/iccv48922.2021.01294
17. Yuan, W. (2018), "PCN: Point Completion Network" / W. Yuan et al. 2018 *International Conference on 3D Vision (3DV)*, Verona, 5–8 September 2018. DOI: https://doi.org/10.1109/3dv.2018.00088
18. Charles, R. Q. (2017), "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation" / R. Q. Charles et al. 2017 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 21–26 July 2017. DOI: https://doi.org/10.1109/cvpr.2017.16
19. Liu, M. (2020), "Morphing and Sampling Network for Dense Point Cloud Completion" / M. Liu et al. *Proceedings of the AAAI Conference on Artificial Intelligence.* 2020. Vol. 34, No. 07. P. 11596–11603. DOI: https://doi.org/10.1609/aaai.v34i07.6827

Сучасний стан наукових досліджень та технологій в промисловості. 2025. № 2 (32)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

20. Lawrance, N. R. J., Chung J. J., Hollinger G. A. (2017), "Fast Marching Adaptive Sampling". *IEEE Robotics and Automation Letters*. 2017. Vol. 2, No. 2. P. 696–703. DOI: https://doi.org/10.1109/lra.2017.2651148
21. Chen, X., Chen, B., Mitra, N. J. (2020),"Unpaired Point Cloud Completion on Real Scans using Adversarial Training". *International Conference On Learning Representations* (ICLR 2020). DOI: https://doi.org/10.48550/arXiv.1904.00069
22. "Generative Adversarial Networks / I. J. Goodfellow et al. Advances in Neural Information Processing Systems". 2014. URL: https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf. (last accessed: 15.05.2024).
23. Huang, Z. (2020), "PF-Net: Point Fractal Network for 3D Point Cloud Completion" / Z. Huang et al. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020. DOI: https://doi.org/10.1109/cvpr42600.2020.00768
24. Mendoza, A. (2020), "Refinement of Predicted Missing Parts Enhance Point Cloud Completion" / A. Mendoza et al. *Computer Vision and Pattern Recognition*. 11 p. DOI: https://doi.org/10.48550/arXiv.2010.04278
25. Spurek, P. (2022), "HyperPocket: Generative Point Cloud Completion" / P. Spurek et al. 2022 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, Japan, 23–27 October 2022. DOI: https://doi.org/10.1109/iros47612.2022.9981829
26. Ha, D., Dai, A., Le, Q. V. (2022), "Hypernetworks. International Conference on Learning Representa- tions". *Machine Learning*. 2022. DOI: https://doi.org/10.48550/arXiv.1609.09106
27. Xiang, P. (2021), "SnowflakeNet: Point Cloud Completion by Snowflake Point Deconvolution with Skip-Transformer" / P. Xiang et al. 2021 *IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 10–17 October 2021. DOI: https://doi.org/10.1109/iccv48922.2021.00545
28. Pan, L. (2020), "ECG: Edge-aware Point Cloud Completion with Graph Convolution". *IEEE Robotics and Automation Letters*. 2020. Vol. 5, No. 3. P. 4392–4398. DOI: https://doi.org/10.1109/lra.2020.2994483
29. Kullback, S., Leibler, R. A.(1951), "On Information and Sufficiency". *The Annals of Mathematical Statistics*. 1951. Vol. 22, No. 1. P. 79–86. DOI: https://doi.org/10.1214/aoms/1177729694
30. Wu, Y. (2020), "Graph Convolutional Networks with Markov Random Field Reasoning for Social Spammer Detection" / Y. Wu et al. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2020. Vol. 34, No. 01. P. 1054–1061. DOI: https://doi.org/10.1609/aaai.v34i01.5455
31. Lecun, Y. (1998), "Gradient-based learning applied to document recognition" / Y. Lecun et al. *Proceedings of the IEEE*. 1998. Vol. 86, No. 11. P. 2278–2324. DOI: https://doi.org/10.1109/5.726791
32. LeCun, Y., Bengio, Y., Hinton, G. (2015), "Deep learning. Nature". Vol. 521, No. 7553. *PubMed*. P. 436–444. DOI: https://doi.org/10.1038/nature14539
33. Hamilton, J. G. (2016), "What is a good medical decision? A research agenda guided by perspectives from multiple stakeholders" / J. G. Hamilton et al. *Journal of Behavioral Medicine*. 2016. Vol. 40, No. 1. P. 52–68. DOI: https://doi.org/10.1007/s10865-016-9785-z
34. Mikolov, T. (2013), "Efficient Estimation of Word Representations in Vector Space" / T. Mikolov et al. *International Conference on Learning Representations*. 2013. DOI: https://doi.org/10.48550/arXiv.1301.3781
35. Perozzi, B., Al-Rfou, R., Skiena, S. S. (2014), "DeepWalk: online learning of social representations". KDD '14: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 20th ed. 2014. DOI: https://doi.org/10.1145/2623330.2623732
36. Mallat, S. Wavelet Tour of Signal Processing. Elsevier Science & Technology Books, 1999.
37. Hammond, D. K., Vandergheynst, P., Gribonval, R. (2011), "Wavelets on graphs via spectral graph theory". *Applied and Computational Harmonic Analysis*. 2011. Vol. 30, No. 2. P. 129–150. DOI: https://doi.org/10.1016/j.acha.2010.04.005
38. Kipf, T. N., Welling, M. (2016), "Semi-Supervised Classification with Graph Convolutional Networks". *Machine Learning*. 2016. DOI: https://doi.org/10.48550/arXiv.1609.02907
39. Hamilton, W. L., Ying, R., Leskovec, J. (2017), "Inductive Representation Learning on Large Graphs". *Social and Information Networks*. 2017. DOI: https://doi.org/10.48550/arXiv.1706.02216
40. Wang, Y. (2019), "Dynamic Graph CNN for Learning on Point Clouds" / Y. Wang et al. *ACM Transactions on Graphics*. 2019. Vol. 38, No. 5. P. 1–12. DOI: https://doi.org/10.1145/3326362
41. Yang, Y. (2018), "FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation" / Y. Yang et al. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, 18–23 June 2018. DOI: https://doi.org/10.1109/cvpr.2018.00029
42. Cai, P. (2024), "Orthogonal Dictionary Guided Shape Completion Network for Point Cloud" / P. Cai et al. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2024. Vol. 38, No. 2. P. 864–872. DOI: https://doi.org/10.1609/aaai.v38i2.27845
43. Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. (2019), "Deep double descent: Where bigger models and more data hurt". *Machine Learning*. DOI: https://doi.org/10.48550/arXiv.1912.02292

Надійшла (Received) 05.04.2025

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2025. No. 2 (32)*

Відомості про авторів / About the Authors

Чухран Іван Дмитрович – Харківський національний університет радіоелектроніки, студент кафедри штучного інтелекту, Харків, Україна; e-mail: ivan.chukhran@nure.ua; ORCID ID: https://orcid.org/0009-0004-7720-6979

Удовенко Сергій Григорович – доктор технічних наук, професор, Харківський національний економічний університет ім. С. Кузнеця, завідувач кафедри інформатики та обчислювальної техніки, e-mail: serhiy.udovenko@hneu.net; ORCID ID: https://orcid.org/ ORCID: 0000-0001-5945-8647

Шергін Вадим Леонідович – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри штучного інтелекту, Харків, Україна; e-mail: vadim.shergin@nure.ua; ORCID ID: https://orcid.org/0000-0002-4388-8180

Чала Лариса Ернестівна – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри штучного інтелекту, Харків, Україна; e-mail: larysa.chala@nure.ua; ORCID ID: https://orcid.org/ 0000-0002-9890-4790

Chukhran Ivan – Kharkiv National University of Radio Electronics, Student of the Artificial Department, Kharkiv, Ukraine.

Udovenko Serhii – Doctor of Sciences (Engineering), Professor, Head at the Department of Informatics and Computer Technics of S. Kuznets Kharkiv National University of Economics, Kharkiv, Ukraine.

Shergin Vadim – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor of the Artificial Department, Kharkiv, Ukraine.

Chala Larysa – PhD (Engineering Sciences), Associate Professor, Kharkiv National University of Radio Electronics, Associate Professor of the Artificial Department, Kharkiv, Ukraine.

METHOD FOR AUGMENTING 3D POINT CLOUD MODELS USING GRAPH NEURAL NETWORKS

Constructing models representing three-dimensional objects as a set of unstructured points (3D point cloud models) in space is becoming increasingly widespread across various domains, including autonomous navigation, robotics, virtual/augmented reality, and 3D reconstruction. Accurately capturing and processing 3D point cloud data is critical for applications requiring a comprehensive understanding of the surrounding environment, such as obstacle avoidance, path planning, and scene modelling. However, due to various reasons, point clouds often contain missing regions, posing significant challenges for subsequent data processing. Incomplete point cloud data can have serious consequences, for instance, in autonomous navigation systems, where errors may lead to collisions or other hazardous situations. Addressing this issue is crucial for the reliable processing of 3D data. This work aims to develop and investigate a method for automatically completing and reconstructing point clouds using graph neural networks. The study's primary objectives include analysing existing approaches to constructing and restoring three-dimensional graph models, developing and implementing a method for automatically completing point clouds using graph neural networks and modelling the proposed method for tasks related to the completion and 3D reconstruction of point cloud models. In this work, a conceptual model for point cloud completion was developed using graph neural networks, enabling the efficient encoding of incomplete point clouds as graphs and the prediction of missing points. The proposed solution for completing incomplete 3D point clouds offers scientific novelty and combines the power of graph neural networks (GNN) with the Point Completion Network (PCN) architecture. The suggested approach allows for high-quality restoration of incomplete 3D data, essential for numerous applications, such as 3D reconstruction, robot navigation, and more. The practical significance of the work's results is validated by the modelling outcomes of the developed method on classical datasets and their comparison with existing approaches to solving the studied problem. A promising direction for further research on this topic includes testing various architectures of graph neural networks, tuning hyperparameters, applying alternative loss functions, and leveraging more powerful computational resources to train the constructed neural network models.

Keywords: point cloud; deep learning; graph neural network; automatic point cloud completion; graph encoding; 3D reconstruction of graph models.

Бібліографічні описи / Bibliographic descriptions

Чухран І. Д., Удовенко С. Г., Шергін В. Л., Чала Л. Е. Метод доповнення 3D-моделей точкових хмар з використанням графових нейронних мереж. *Сучасний стан наукових досліджень та технологій в промисловості*. 2025. № 2 (32). С. 129–150. DOI: https://doi.org/10.30837/2522-9818.2025.2.129

Chukhran, I., Udovenko, S., Shergin, V., Chala, L. (2025), "Method for augmenting 3D point cloud models using graph neural networks", *Innovative Technologies and Scientific Solutions for Industries*, No. 2 (32), P. 129–150. DOI: https://doi.org/10.30837/2522-9818.2025.2.129