Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра математики

Магістерська робота

освітній ступінь – магістр

на тему: «**ДИСТИЛЯЦІЯ ДАНИХ У КОНТЕКСТІ ЗАДАЧІ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ / DATA DISTILLATION IN THE CONTEXT OF IMAGE CLASSIFICATION**»

Виконав: студент 2-го року навчання,
освітньо-наукової програми
«Прикладна математика», 113

Мокрий Михайло Вікторович

Керівник _Швай Н.О,_____

ст. викладач

Рецензент_____
(прізвище та ініціали)

Магістерська робота захищена
з оцінкою _____

Секретар ЕК _____
«____» _____ 20____ р.

Київ – 2023

ЗМІСТ

ВСТУП

Сучасний світ наповнений величезною кількістю інформації, яку потрібно вміти швидко та якісно опрацьовувати, аби не відставати від важливих новин сьогодення, наукового прогресу, а також просто розуміти, що відбувається навколо. Але співмірно з такою великою кількістю інформації перед людиною з'являються не менш серйозні виклики, тому що майже неможливо обробити такі обсяги новин, які пропонують новинні ресурси, і зробити їх якісний аналіз, побудувавши логічні зв'язки між цими даними. Саме тому люди користуються різними сервісами для того, щоб отримати стиснені дані, які коротко та зрозуміло доносять основну ідею та узагальнений опис тієї чи іншої події, але не обтяжують споживачів повним обсягом інформації з незначними деталями, які суттєво не впливають ні на що. Це дозволяє людям отримувати бажану інформацію набагато швидше, не втрачаючи при цьому основні ідеї, які були закладені в її основу, та побудувати в своїй голові логічні зв'язки між отраминими даними.

Можна провести певні паралелі з областю штучних нейронних мереж та машинного навчання, які набули шаленої популярності особливо в останні роки. Розв'язання різноманітних задач методами машинного навчання залежить від доступності наборів даних, деякі з яких можуть бути дійсно величезними, та від складності самої моделі навчання. Це суттєво обмежує в своїх можливостях звичайних користувачів, які не мають доступу до великих та потужних обчислювальних машин. Через це вони змушені або відмовитись від подальшого використання наборів даних та багатошарових й складних штучних нейронних мереж з великою кількістю параметрів, або змушені миритись з надзвичайно довгим часом тренування моделі. Тобто навчання такої моделі потребує використання надмірних обчислювальних ресурсів. З точки зору логіки, проблему витратності навчання не можна було ігнорувати, тому що вона обмежувала використання

наборів даних великого розміру, доступность цих даних та розмір моделі нейронної мережі. Не менш важливим чинником є те, що обмеження використання ресурсів сповільнювало науковий прогрес і подальший розвиток машинного навчання. З’являлись різні задачі, які не могли бути вирішені поточними методами через брак наявних ресурсів.

Великі набори даних, що містять мільйони тренувальних даних, стають цілком природніми для отримання сучасних моделей машинного навчання в різних галузях, включаючи комп'ютерний зір, класифікацію зображень, обробку природної мови та розпізнавання мовлення. В таких масштабах навіть зберігання та попередня обробка даних стає обтяжливою, а навчання моделей машинного навчання на них вимагає спеціалізованого обладнання та інфраструктури. Також варто зазначити ефективність алгоритмів машинного навчання зумовлена ще й здатністю виокремлювати корисні ознаки з великих обсягів даних. Зі збільшенням розмірів моделей і наборів даних методи дистиляції даних, які стискають великі набори даних до значно менших, але високопродуктивних, стануть цінними з точки зору ефективності навчання та вилучення корисних ознак.

Саме тому перед науковою спільнотою та фахівцями з машинного навчання постало питання, яким чином можна пришвидшити час навчання моделі та покращити ситуацію з надмірним навантаженням на обчислювальні ресурси шляхом зменшення кількості параметрів або об'єму даних, на яких модель навчалась, а головне – без негативних впливів на результат навчання. Отже, актуальність цієї проблеми є доволі високою.

Метою даної наукової роботи є аналіз різних методів дистиляції даних для вирішення задачі класифікації зображень та дослідження різних способів початкової ініціалізації зображень для методу дистиляції даних DC [1].

Основними завданнями даного дослідження є:

1. Зробити огляд основних проблем використання великих наборів даних та способів зменшення складності моделей;
2. Розглянути різні методи дистиляції даних та виконати порівняльний аналіз результатів використаних методів;
3. Дослідити вплив різних способів початкової ініціалізації зображень для методу дистиляції даних DC та проаналізувати їх результати.

В якості інструментів дослідження було використане середовище розробки Google Colab з графічним апаратним прискорювачем T4, оперативною пам’яттю Standart RAM та бібліотекою машинного навчання PyTorch, яка базується на мові програмування Python. Для практичної частини була взята згорткова нейронна мережа ConvNet, а також набір даних CIFAR-10 [2].

Сама робота складається з трьох розділів. Перший розділ присвячений опису основних проблем використання великих наборів даних та способів зменшення складності моделі нейронної мережі. В другому розділі розглядаються основні існуючі методи дистиляції даних, які поділені на два типи: методи вибору підмножин та градієнтні методи дистиляції даних. Третій розділ присвячено аналізу результатів існуючих методів дистиляції даних, дослідженню різних способів ініціалізації зображень для методу DC, а також порівняльному аналізу результатів з оригінальним методом.

У результаті наукової роботи було розглянуто різні методи дистиляції даних та досліджено вплив різних способів ініціалізації зображень на результати методу DC.

РОЗДІЛ 1: Проблеми використання великих наборів даних та складності моделей в контексті класифікації зображень

1.1 Класифікація зображень

Класифікація зображень – це різновид задач в машинному навчанні, метою яких є визначення класу, до якого належить зображення. Тобто модель нейронної мережі під час навчання вчиться розпізнавати, до якого конкретного класу належить зображення. Класифікація зображень застосовується в різноманітних сферах життя, таких як: діагностика захворювань в медицині шляхом аналізу медичних знімків, розпізнавання облич у сфері безпеки, визначення об'єктів на дорозі для водіння з автопілотом та багатьох інших.

В основі класифікації зображень лежить принцип навчання "з вчителем", тобто кожен елемент з набору навчальних даних має свою мітку або клас. При вирішенні задачі за допомогою класифікації зображень навчальний та тестувальний набір даних містить набір розмічених зображень, де кожне зображення асоціюється з певним класом, або іншими словами – має свою мітку.

В процесі навчання модель вчиться розрізняти, до якого класу належить те чи інше зображення, тобто створює набір ознак для кожного класу. В машинному навчанні існує багато різноманітних метрик для оцінки моделі штучної нейронної мережі, але найпопулярнішою з них у сфері класифікації є метрика точності (англ. Accuracy).

Точність класифікації визначається як кількість правильних прогнозів поділена на загальну кількість прогнозів. Тобто результатом успішного навчання є навчання моделі з високою точністю визначення, до якого відповідного класу належить вхідне зображення.

Звісно, в ідеальних умовах ця точність повинна бути на рівні ста відсотків, але на практиці, використовуючи зображення з реального світу, досягти такого результату просто неможливо.

Головною умовою є те, що для тестування не можна брати ті зображення, які вже були задіяні в навчанні, набір тестувальних даних повинен містити абсолютно нові для моделі нейронної мережі дані, в даному випадку — зображення. Саме тому кожен набір даних, який використовується для вирішення задачі класифікації зображень, містить як навчальні, так і тестувальні дані.

Існує декілька підходів вирішення задачі класифікації зображень. Однак найбільш поширеним з них є використання згорткових нейронних мереж для навчання.

Класифікація зображень є дуже популярним типом задач у машинному навчанні. Використовуючи можливості згорткових нейронних мереж можна отримати високий рівень точності моделі, який дозволяє конкурувати з можливостями людського ока. Подальший розвиток класифікації зображень та згорткових нейронних мереж значно розширить галузь практичного використання даної задачі в недалекому майбутньому. Якщо використовувати все складніші і складніші моделі, то набори даних повинні також бути більш великими та об'ємними, що знову таки призводить до потреби в застосуванні методів дистиляції зображень задля зменшення розмірів вхідних даних. Саме тому вирішення проблеми класифікації зображень було взяте за основу в даній науковій роботі.

1.2 Проблеми великих наборів даних

1.2.1 Загальний опис

На перший погляд здається, що чим більший обсяг даних наявний в розпорядженні, тим кращий результат можна отримати, використавши ці знання для тренерування моделі нейронної мережі. Але не все так просто.

Використання великих наборів даних може спричинити декілька проблем. Деякі з них, а саме проблема складності обчислень та масштабування нейронних мереж, є загальними проблемами, які часто зустрічаються в багатьох сферах машинного навчання, наприклад, глибокого навчання, а деякі найчастіше можуть виникати саме на великих наборах даних [3].

Найбільш поширеними проблемами, які характерні при використанні великих наборів даних, є проблеми:

1. Складності обчислень
2. Збереження набору даних
3. Масштабування моделі
4. Упередженості вибору
5. Якості даних
6. Перенасищення моделі

Більш детальний опис цих проблем наведено нижче.

1.2.2 Складність обчислень

Чим складніша модель, тобто чим більше в неї параметрів, тим більшого набору даних вона потребує для успішного навчання. Від цього залежить обсяг обчислювальних ресурсів, необхідних для того, щоб навчити модель нейронної мережі. А отже використання великих наборів даних потребує великої обчислювальної потужності та є дуже часозатратним. Навчання нейронної мережі

на великому наборі даних іноді може займати години, або навіть цілі дні, а зберігання такого великого набору даних може стати величезною проблемою, особливо коли наявні обчислювальні ресурси є обмежені певними рамками.

1.2.3 Збереження набору даних

Ще однією проблемою, пов'язаною з ресурсами, є проблема використання пам'яті для збереження набору даних. Ні для кого не секрет, що набір даних повинен бути збереженим на фізичному накопичувачі. Але якщо цей набір даних містить сотні тисяч або мільйони елементів, то відповідно пам'яті для зберігання такого обсягу даних повинно бути в рази більше. В контексті вирішення задачі класифікації зображень елементами набору даних, відповідно є зображення, які в залежності від їх розмірів займають в рази більше місця ніж наприклад текстові дані. Тому і з'являється проблема зберігання такого великого набору даних. Звісно, її можна вирішити за допомогою використання розподілених систем зберігання даних або хмарних сховищ, але це не звільняє від наявності самого факту існування даної проблеми при використанні великих наборів даних.

Так як набори даних продовжують збільшуватись дослідники та інженери повинні продовжувати розробляти нові методи їх обробки, зберігання та аналізу, одним з цих методів і є дистиляція даних.

1.2.4 Масштабування моделі

Існує ще одна проблема, яка з великою вероґідністю може виникнути з масштабуванням моделі нейронної мережі. При збільшенні кількості параметрів та складності моделі виростають відповідно й обсяги необхідних для успішного навчання даних, а загальне навантаження на обчислювальні ресурси може досягти непомірно великих значень через те, що масштабувати модель до нескінченності просто неможливо. Це, безумовно, негативно позначиться на продуктивності даної

моделі нейронної мережі, а що головне – на її результатах. Вона просто стане надто складною і не зможе в повній мірі виконувати задачі, які перед нею ставились. Це особливо є критичним для моделей, які вирішують поставлені перед ними задачі в режимі реального часу, де швидкість роботи є надважливим чинником, бо від нього залежить загальний успіх. Наприклад, припустимо, що існує система розпізнавання облич в аеропорті: модель, яка використовувалась в основі алгоритму розпізнавання цієї системи, розвивалась, ставала все складнішою. Врешті-решт, ця модель зіткнулася з проблемою масштабування, швидкість роботи даного алгоритму суттєво знизився, а отже модель не змогла відповідати стандартам якості та виконувати роботу, для якої її й було створено.

1.2.5 Упередженість вибору

Упередженість вибору є доволі поширеною проблемою, яка залежить від набору вхідних даних. Тобто, якщо набір вхідних даних не є репрезентативним у тій галузі, в якій він має бути представленим, це може призвести до отримання неточних результатів. Найчастіше дана проблема виникає, коли команда розробки моделі нейронної мережі фокусується на вирішенні конкретної задачі, не думаючи, яке саме вирішення буде використовуватись в майбутньому і як дані датасету можуть призвести до неправильних висновків або прогнозів. Величезні набори даних дуже важко контролювати на цілісність даних і їх негативний вплив на результати навчання моделі.

1.2.6 Якість даних

Ще однією проблемою є проблема якості даних. Чим більший набір даних, тим вища вірогідність, що такий набір даних має більше проблем з самими даними. Загальні проблеми можуть бути наступними:

1. Неповність даних

   Неповні дані відносяться до даних, які містять порожні значення.

Причини цієї помилки є різноманітними, а саме: некоректне збирання даних, людський фактор або системна помилка. Проблема неповних даних може призвести до виникнення упереджених моделей та отримання неточних результатів, тобто невдачі в тренуванні моделі.

2. Непослідовність даних

Непослідовні дані містять суперечливі значення для того ж самого класу чи атрибуту. Наявність цих даних може призвести до невдалого тренування моделі.

3. Некоректність даних

Некоректності дані – це дані, які містять помилкові дані, а саме дані з невірними значеннями, неправильними метриками або невірними форматами. Поява некоректних даних може бути пов’язана з помилками при введенні даних, проблемами з обробкою цих даних або системному збої. Це може призвести до неправильного тренування моделі або взагалі виникненню помилок при запуску самого процесу.

4. Дублювання даних

Проблема може виникати при некоректній роботі алгоритму витяганням даних з якогось джерела або проблеми з записом даних в набір. Ці дані зазвичай містять однакові значення та можуть призвести до виникнення проблеми упередженості моделі.

1.2.7 Перенасичення моделі

Проблема перенасичення часто виникає при навчанні моделі штучної нейронної мережі, коли модель засвоює нові знання занадто добре, що в свою чергу призводить до поганого узагальнення нових даних.

Модель стає занадто складною та нездатною аналізувати нові дані. Це призводить до низької продуктивності, а у випадку вирішення задачі класифікації зображень – до низької точності результатів на нових даних, навіть незважаючи на

високу точність при використанні тренувальних даних. Існує декілька різних методів для того, щоб вирішити проблему перенасичення моделі. Найбільш поширеним з цих методів є метод регуляризації, який використовується в багатьох моделях нейронної мережі.

1.3 Способи зменшення складності навчання моделі нейронної мережі

1.3.1 Загальний опис

Існує декілька різних підходів для вирішення проблеми використання надмірної кількості ресурсів для навчання моделі штучної нейронної мережі та зменшення її складності. Всі ці способи мають різну специфіку використання, а також містять різні шляхи вирішення цієї проблеми. Найбільш популярними та ефективними з них є:

1. Навчання з перенесенням (англ. Transfer learning).
2. Активне навчання (англ. Active learning)
3. Попереднє навчання (англ. Pre-trained leaning)
4. Дистиляція знань (англ. Knowlege distillation)
5. Дистиляція даних (англ. Data distillation)

Загальний опис кожного з цих способів наведено нижче.

1.3.2 Навчання з перенесенням

В основі навчанні з перенесенням лежить ідея використання знань моделі, навченній на одній задачі для виконання іншої [4]. Цей метод дозволяє використовувати попередні знання, які отримані при навчанні моделі, для покращення результатів нової моделі та пришвидшення її навчання. Тобто використовуючи ваги вже навченої моделі можна вирішити нову задачу більш ефективно, тому що нова модель вже буде мати знання попередньої. Цей підхід зазвичай використовується, коли навчальні дані є обмеженими для нової моделі.

1.3.3 Активне навчання

Ще одним методом вирішення проблеми використання надмірних обсягів ресурсів є метод активного навчання [5]. Його основним підходом є залучення користувача в процес навчання моделі, який надає експертні оцінки для деяких

наборів даних, якщо є сумніви щодо правильності визначення результатів. Особливо добре даний метод працює для вирішення задачі класифікації зображень, тому це значно спрощує експертну оцінку в силу використання візуальних зображень, які можна класифікувати за допомогою людського ока. Цей метод дозволяє покращити якість навчання моделі, що в свою чергу впливає на збільшення точності результатів та скорочення загального часу навчання.

1.3.4 Попереднє навчання

Попереднє навчання — це метод навчання нейронної мережі, який використовує попередньо навчену модель, зазвичай великої складності, а також використовує великий набір тренувальних даних [6]. У даному підході модель, яка була попередньо навчена і досягла високої точності в вирішенні певної задачі, використовується в якості початкової точки в навчанні нової моделі на значно меншому наборі даних. Тобто в новій моделі використовуються ваги попередньо навченної моделі, як початкові ваги ініціалізації. Таким чином можна створювати не лише глибокі мережі для вирішення тих чи інших задач, а й створювати більш точні моделі для задач з обмеженою кількістю даних.

Сам процес попереднього навчання вимагає наявність часу для навчання моделі на великому обсязі даних, а також великих обчислювальних потужностей. Але зазвичай вже натреновані моделі попереднього навчання, які мають високу продуктивність, є у відкритому доступі і їх початкові ваги можна використати для початкових вагів ініціалізації в новій нейроній мережі.

Також не можна не зазначити той факт, що сам підхід використання набутих раніше знань позитивно відображається на складності нової моделі, тому що з його допомогою можна зменшити часові й обчислювальні витрати на навчання моделі, а також покращити якість навчання, і відповідно — результат цього навчання.

Можна припустити, що попереднє навчання і навчання з перенесенням є однаковими методами, але це не вірно. Основна відмінність між ними полягає в тому, що при навчанні з перенесенням модель навчається на схожій задачі, в той час як при попередньому навчанні модель навчається на великому наборі даних, а потім допрацьовується на новій задачі. Навчання з перенесенням застосовується, коли цільова задача має обмежену кількість даних з мітками, в той час як навчання з попереднім навчанням використовується, коли цільова задача вимагає великої кількості даних з мітками.

1.3.5 Дистиляція знань

В основі методу дистиляції знань нейронної мережі закладено доволі простий принцип – зменшення складності моделі шляхом її спрощення, або іншими словами, передачі набутих знать від більш складної моделі до менш складної. Даний метод показав неабияку ефективність у створенні спрощених моделей з використанням знань складних моделей. Також цей підхід дозволяє покращити точність нової мережі, яка має меншу кількість параметрів, використовуючи знання більш складної нейронної мережі.

1.3.6 Дистиляція даних

Дистиляція даних – це метод, який використовується в машинному навчанні для покращення продуктивності та узагальнення моделі шляхом вибору більш інформативного та компактного набору даних. Він включає в себе процес вибору підмножини репрезентативних та інформативних елементів множини почтакового набору даних.

В контексті класифікації зображень, загальний алгоритм дистиляції даних можна виразити математично. Припустимо, що існує набір тренувальних даних, який позначається як $D^{tr} = \{(x_i, y_i)\}/_{i=1}^{K}$, де $x \in \chi$ та $y \in \{0, \ldots, C-1\}$, χ – це

множина вхідних даних, C – кількість класів в наборі тренувальних даних D^{tr} та K – кількість вхідних даних в D^{tr}. Задачею класифікації даних є знаходження функції $f \in F$, такої що $f(x) = y$ для $x \in \chi$. Алгоритмом дистиляції даних є створення нового набору тренувальних даних $\widehat{D}^{tr} = \{(x_i, y_i)\}/_{i=1}^{\widehat{K}}$, в якому $\widehat{K} \ll K$. Вирішення задачі класифікації зображень, що виражається за допомогою функції $\hat{f}$, отриманої з набору даних $\widehat{D}^{tr}$, повинна мати схожу результативність.

$$perf(f(D^{val})) \approx perf(\hat{f}(D^{val}))$$

Для задачі класифікації зображень вхідний простір χ стає простором зображень. Вирішуючи цю задачу за допомогою глибокого навчання можна припустити, що множина $F = F_\theta$ є родиною параметричних функцій, які представляють деяку нейронну архітектуру та її ваги θ.

Сам процес розв'язання задачі класифікації зображень, який пов'язаний з навчанням нейронної мережі, тобто знаходження вагів θ, які мінімізують емпіричну функцію втрат цієї мережі, можна виразити за допомогою формули, наведеної нижче.

$$\theta^* = arg_\theta min\, L(D^{tr}, \theta)$$

В останні роки метод дистиляції даних став популярним підходом в галузі машинного навчання. Він дозволяє суттєво зменшити розмір набору даних й вирішити проблему використання надмірних обчислювальних та часових ресурсів, тобто зменшити складність навчання. Також варто зазначити, що тема дистиляції даних активно досліджується, але наразі поки що не було запропоновано універсального методу для дистиляції набору даних, який міг би бути застосований для вирішення будь-яких задач. Саме завдяки актуальності і наявності великого простору для подальших досліджень перед цією роботою була поставлена задача – дослідження методів дистиляції даних у контексті класифікації зображень.

Найбільш поширеними методами дистиляції даних є методи вибору підмножини даних та градієнтні методи дистиляції даних. Більш детальний опис даних методів представлений у наступному розділі.

РОЗДІЛ 2: Методи дистиляції даних

2.1 Методи вибору підмножини

2.1.1 Загальний опис

Метод вибору підмножини даних полягає в скороченні обсягів даних шляхом вибору підмножини даних з великого набору даних, не втрачаючи при цьому інформаційну цінність. Застосування методу вибору підмножини даних може бути найбільш корисним у випадках наявності великої колекції даних в умовах обмежених часових або обчислювальних ресурсів, коли обробка всього набору даних не є практичною.

За лінійний або майже лінійний час великі набори даних можуть бути стиснуті в менший датасет, який може використовуватись у алгоритмі та мати схожі результати в порівнянні з використанням усього набору даних. Вибір підмножини даних грає важливу роль для зменшення розміру набору даних.

Підмножина даних – це своєрідний зменшений набір даних, який представляє репрезентацію повного набору даних [7]. Підмножину даних можна використати для такого самого алгоритму, що й з повним набором даних. Результат виконання алгоритму з використанням підмножини даних повинен бути наближеним до результату на повному наборі даних для того, щоб вважати вибрану підмножину даних успішною.

В контексті класифіцації зображень вибір підмножини даних грає надзвичайно важливу роль, тому що від нього залежить якість вибраних елементів набору даних, а також їх можливість в повній мірі репрезентувати початковий набір даних. Також методи вибору підмножини даних можуть бути розглянуті в якості

методів дистиляції даних, бо вони також виконують основну функцію, а саме зменшення розміру початкового набору даних без втрати якості навчання моделі.

Метод вибору підмножини є ефективним способом зменшення кількості початкових навчальних даних для тренування моделі, що дозволяє зменшити час, необхідний для тренування моделі, та зменшити кількість обчислювальних ресурсів.

Існує багато різних способів вибору підмножини даних [8], але в даній роботі було вирішено зосередитись на декількох з них, а саме: Random Sampling, Herding, K-Center та Forgetting. Ці методи вибору підмножини даних є найбільш популярними в сфері дистиляції даних і кожен з них має власний алгоритм вирішення задачі. Саме тому було вирішено розглянути дані методи вибору підмножини даних в рамках даного дослідження.

2.1.2 Forgetting

У машинному навчанні метод Forgetting [9] відноситься до процесу динамічного оновлення та зміни навчальних даних, які модель використовує для вибору підмножини даних. При виборі даних для моделей традиційні методи часто зосереджуються на пошуку інформативних прикладів на початку процесу навчання. Однак ці підходи не враховують, як змінюється розподіл даних або як змінюються потреби моделі.

Метод Forgetting долає обмеження безперервного навчання та адаптації даних, виключаючи нерелевантні або застарілі дані, та прагнучи знайти і вибрати підмножину даних, які є найбільш релевантними для поточної моделі. Модель може пристосовуватися до змін у розподілі даних, відбираючи забуті або вилучаючи певні приклади з процесу навчання, що підвищує загальну продуктивність та ефективність моделі нейронної мережі.

Методи вибору підмножин даних на основі методу Forgetting виявилися перспективними в ряді областей, включаючи обробку природної мови і класифікацію зображень. Вони надають моделям можливість постійно навчатися на релевантних і поточних даних, покращуючи продуктивність, адаптивність і навички узагальнення.

Багато моделей машинного навчання, зокрема нейронні мережі, не здатні до безперервного навчання. Вони мають тенденцію забувати раніше вивчену інформацію під час навчання на нових завданнях, що зазвичай називають катастрофічним забуванням. Однією з причин катастрофічного забування в нейронних мережах є зсув у розподілі вхідних даних між різними завданнями, наприклад, відсутність спільних факторів або структури у вхідних даних різних завдань, що в майбутньому може призвести до того, що стандартні методи оптимізації будуть отримувати радикально різні рішення щоразу, коли постане нове завдання.

Даний метод можна більш детально описати, використовуючи математичну модель. Уявимо, що в нас є певний набір даних $\mathcal{D} = (x_i, y_i)_i$, який складається з кортежу даних, а саме елементу з вхідної множини даних, та класу, який класифікує його. Маючи цей набір даних можна дізнатися умовний розподіл ймовірностей $p(y/x; \theta)$, використовуючи параметри глибокої нейронної мережі θ. Модель нейронної мережі навчена мінімізувати кросс-ентропічні втрати $R = \frac{1}{|\mathcal{D}|}\Sigma_i L(p(y_i/x_i; \theta), y_i)$, де $y_i \in 1, \dots k$. Власне мінімізація виконується за допомогою стохастичного градієнтного спуску з початковими випадковими параметрами θ^0 шляхом вибору випадкових елементів з множини набору даних $\mathcal{D}$.

Події забування (англ. Forgetting) та навчання можна визначити як $\hat{y}_i^t = arg\ max_k\ p(y_{ik}/x_i; \theta^t)$, де клас, до якого належить елемент навчального набору даних x_i, отриманий після t кроків стохастичного градієнтного спуску. Також

точність чи елемент набору даних i є правильно визначений на кроці t можна позначити як $acc_i^t = \mathbb{1}_{\hat{y}_i^t = y_i}$. Даний елемент стає “забутим” тоді, коли точність acc_i^t зменшується між двома послідовними оновленнями $acc_i^t > acc_i^{t+1}$.

Елементи набору даних називаються незабутими (англ. Unforgettable), якщо вони вивчені в певний момент та не зазнають жодних подій забування під час навчання моделі нейронної мережі.

Загальний алгоритм методу Forgetting можна виразити схематично.

Алгоритм Forgetting:

1: Ініціалізувати $prev\_acc_i = 0,\ i \in \mathcal{D}$
2: Ініціалізувати забування $T[i] = 0,\ i \in \mathcal{D}$
3: **while** навчання не завершено **do**:
4: Створити підмножину $B \sim \mathcal{D}$
5: **for** елементу $i \in B$ **do**:
6: Порахувати acc_i
7: **if** $prev\_acc_i > acc_i$ **then:**
8: $T[i] = T[i] + 1$
9: $prev\_acc_i = acc_i$
10: Оновити градієнти класифікатора на B
Вихідні дані: T

Отже, метод Forgetting є одним з основних методів вибору підмножин, тому було вирішено розглянути його в даній науковій роботі.

2.1.3 Herding

Метод вибору підмножини, який має назву “стадо” (англ. Herding) [10], використовуються для вибору невеликої підмножини даних, для того щоб

відобразити різноманітність і репрезентативність загального набору даних. Такий підхід є дуже корисним, коли пам'ять або обчислювальні можливості не дозволяють використовувати весь набір даних.

Метод стадного аналізу натхненний поведінкою пастуха, який пасе отару овець і має на меті спрямувати отару таким чином, щоб вона відображала загальні характеристики всієї популяції. Аналогічно, в контексті вибору підмножини даних, метою є вибір підмножини прикладів, яка ефективно представляє різноманітність і розподіл повного набору даних.

Метод Herding гарантує, що відібрані екземпляри представляють різні регіони простору даних, фіксуючи ключові закономірності та різноманітність, що присутні в повному наборі даних. Це допомагає зменшити надмірність і мінімізувати втрату інформації, яка може виникнути при використанні меншої підмножини.

Даний метод успішно застосовується в різних завданнях машинного навчання, включаючи кластеризацію, класифікацію та зменшення розмірності. Він дозволяє створювати компактні, але інформативні підмножини, які можна використовувати для навчання, оцінки моделей або дослідження даних. Важливо зазначити, що ефективність методу “стада” сильно залежить від вибору міри подібності або відмінності, а також від конкретної проблемної області та характеристик набору даних. Крім того, ефективність алгоритму може бути важливим фактором при роботі з великими наборами даних.

Фундаментальним для методу Herding є використання функції, яка має назву “функція Тіпі”:

$$\ell_0(w) = \sum_{\alpha} w_\alpha \, \bar{g}_\alpha - \left[\sum_{\alpha} w_\alpha g_\alpha(s_\alpha) \right]$$

Вона визначає релевантність кожного елемента на основі його “вкладу” у наближення центру мас невибраних елементів.

За допомогою неї можна сформувати динаміку стада:

$$s_t^* = arg_s max \sum_{\alpha} w_{\alpha t} g_\alpha(s_\alpha)$$

$$w_{\alpha,t+1} = w_{\alpha,t} + \underline{g}_\alpha - g_\alpha(s_{\alpha t}^*)$$

$$f_\beta^*(y_\beta) \leftarrow (\frac{t-1}{t})\, f_\beta^*(y_\beta) + (\frac{1}{t})\, f_\beta(s_{\alpha t}^*) \| [y_\alpha = s_{\alpha t}^*]$$

Це набір детермінованих нелінійних рівнянь оновлення.

Перше оновлення визначає стан s необхідний для того, щоб обчислити градієнт функції Тіпі, тобто визначити локальну плоску область функції, в якій алгоритм зараз знаходиться. Враховуючи цей факт, друге оновлення робить градієнтний крок вперед на $w_{\alpha,t+1}$. Цей процес повторюється, а під час вибору прикладів станів s_t обчислюються середні значення інтересів, використовуючи третє оновлення.

Оновлення “стада” містять дві фази задач мінімуму-максимуму, s, яка мінімізує задачу ℓ_0 та w яка максимізує її. Також в даному методі застосовується версія сходження по локальній координаті (англ. Local Coordinate Ascent), або скорочено LCA. Це своєрідне розширення стандартного алгоритму стадної обробки, яке має на меті покращити якість обраного базового набору шляхом врахування локальних структур у розподілі даних. LCA в методі Herding включає крок локального пошуку для ітеративного уточнення вибраних елементів шляхом оптимізації їхніх позицій у базовому наборі даних.

Метод Herding було вирішено розглянути в межах даної наукової роботи, тому що він є досить популярним методом вибору підмножини.

2.1.4 K-Center

Метод K-Center [11] — це метод вибору підмножини даних у машинному навчанні, який визначає компактну та репрезентативну підмножину екземплярів з великого набору даних. Його мета полягає в тому, щоб мінімізувати надлишковість, одночасно максимізуючи охоплення вибраних екземплярів, гарантуючи, що вони охоплюють основні характеристики всього набору даних.

Метод K-Center спрямований на те, щоб вибрані екземпляри ефективно охоплювали весь набір даних, залучаючи різні регіони простору даних. Він балансує між відбором різноманітних екземплярів і мінімізацією кількості відібраних екземплярів для створення компактної підмножини.

K-Center можна застосовувати в різних задачах машинного навчання, включаючи кластеризацію, класифікацію та виявлення викидів. Він дозволяє створити репрезентативну підмножину, яка може бути використана для навчання, оцінки моделей або дослідницького аналізу даних.

Ефективність методу K-Center залежить від кількох факторів, таких як вибір метрики відстані, вибір початкових точок і конкретної проблемної області. Крім того, ефективність алгоритму може бути важливим фактором при роботі з великими наборами даних, оскільки обчислення відстаней може вимагати значних обчислювальних витрат.

Метою алгоритму є вибрати b центральних точок таким чином, щоб найбільша відстань між точкою даних та її найближчим центром була мінімальна. Тобто, потрібно розв’язати проблему:

$$\min_{s^1 : |s^1| \le b} \max_{i} \min_{j \in s^1 \cup s^0} \Delta(x_i, x_j)$$

Ця проблема має складність NP. Однак можна отримати ефективний розв'язок, використовуючи "жадібний" алгоритм (англ. K-Center-Greedy), який схематично описаний нижче.

Алгоритм K-Center-Greedy:

***Вхідні дані*:** елемент набору даних x_i, існуючий початковий набір даних s^0, та кількість центральних точок b

1: Ініціалізація $s = s^0$

2: **do:**

3: $u = arg\ maxi \in [n] \setminus_s\ min_{j \in s} \Delta(x_i, x_j)$

4: $s = s \cup \{u\}$

5: **while** $|s| = b + |s^0|$

Вихідні дані: $s \setminus s^0$

Даний алгоритм широко використовується в якості методу вибору підмножин, тому його було вирішено розглянуті в рамках даної наукової роботи.

2.2 Градієнтні методи дистиляції даних

2.2.1 Загальний опис

Метою градієнтних методів дистиляції даних є зменшення розміру початкового набору даних і може здаватись, що вони схожі з методами вибору підмножини даних, проте це припущення хибне.

В градієнтних методах дистиляції даних основний акцент робиться саме на створенні синтетичних зображень і прямій взаємодії з їх ознаками, виокремлюючи найбільш корисні з них. Це допомагає правильно оновлювати ваги нейронної мережі, використовуючи ці високоточні синтетичні дані, а також методи взаємодії з градієнтом. Також на відміну від методів вибору підмножини даних, синтезовані дані, в даному випадку — синтетичні зображення, безпосередньо оптимізуються для наступної задачі, і таким чином успіх методу не залежить від наявності репрезентативних вибірок даних.

Окрім зменшення розміру навчальних даних, метод дистиляції може допомогти виявити або виправити помилки в початкових навчальних даних, а також допомогти зрозуміти, які особливості даних є найбільш важливими для найбільш ефективного навчання моделі. В межах даної наукової роботи було вирішено розглянути наступні градієнтні методи дистиляції даних, а саме: Dataset Condensation (DC), Kernel Inducting Points (KIP) та Label Solving (LS).

2.2.2 Dataset Condensation (DC)

Дослідження проблеми скорочення часу навчання моделі та зменшення обчислювальних потужностей було вперше розглянуто в науковій роботі “Дистиляція знать в нейронній мережі” Джеффрі Хінтона та інших [12]. В ній був запропонований спосіб покращення ефективності виконання алгоритмів машинного навчання шляхом зменшення складності моделі, тобто її дистиляції. А

головним показником було те, що даний спосіб був не таким ресурсозатратним, ніж інші способи, які використовувались раніше.

Термін дистиляції даних (англ. Dataset distillation), або скорочено DD, був вперше застосований в науковій роботі під назвою “Dataset distillation”, опублікованій в 2018 році науковцем Вангом та іншими [13].

На відміну від запропонованого раніше методу дистиляції знань нейронної мережі задля спрощення її структури, метод дистиляції даних має в своїй основі зовсім інший підхід.

В основі дистиляції даних лежить зменшення розміру набору даних, шляхом перетворення цих даних в синтетичні зображення, які мають ті самі ознаки, що і зображення початкового датасету, але вони є комбінацією усіх класів цього набору даних.

Це дозволяє набагато ефективніше навчати модель, порівняно з традиційним навчанням, яке часто використовує десятки тисяч кроків градієнтного спуску, і отримати результати близькі до стандартних.

Основні ідеї методу DD були суттєво покращені в більш новому методі, опис якого наведений нижче, тому даний метод було вирішено розглянути лише в якості ретроспективи.

Новий метод дистиляції даних під назвою метод стиснення даних з підбором градієнта (англ. Dataset Condensation with Gradient Matching), або скорочено DC, був опублікований в березні 2021 року науковцем Жао та іншими. Основною метою даного методу було покращення реалізації методу дистиляції даних 2018 року. Незважаючи на те, що основне натхнення при розробці методу DC надходило від ідей методу Ванга DD, новий становить самостійну одиницю. Він переважає метод

Ванга і по продуктивності і по точності результатів, які отримуються під час навчання моделі на дистильованих даних.

Як і в методі DD, науковці сфокусувалися на створені синтетичних зображень, які оптимізовані для навчання нейроної мережі, а також поставили собі ціль отримати найкрашу продуктивність моделі, навченій на маленькому наборі синтетичних зображень.

Припустимо, що існує великий початковий набір даних, який має $|\mathcal{T}|$ пар тренувальних даних $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{T}|}$, який складається з закодованих даних $x \in \chi \subset \mathbb{R}^d$ та класів $y \in \{0, \ldots, C-1\}$. У випадку набору даних CIFAR-10 кожна пара містить закодоване зображення x розмірністю $d = (32, 32, 3)$ з вхідної множини χ, а також клас, до якого воно належить y, де загальна кількість класів становить $C = 10$. Нехай існує функція ϕ з параметром θ, яка правильно прогнозує, до якого класу належить те чи інше зображення $y = \phi_\theta(x)$. Параметри цієї функції можна дізнатися, якщо мінімізувати емпіричну функцію втрат на тренувальному наборі даних, та виразити за допомогою формули:

$$\theta^{\mathcal{T}} = arg_\theta min\, \mathcal{L}^{\mathcal{T}}(\theta)$$

де функція втрат є крос-ентропічною і виражається формулою $\mathcal{L}^{\mathcal{T}}(\theta) = \frac{1}{|\mathcal{T}|} \sum_{(x,y)\in\mathcal{T}} \ell(\phi_\theta(x), y), \ell(\cdot,\cdot)$, в якій параметри $\theta^{\mathcal{T}}$ є мінімізованими.

Ціллю методу дистиляції даних DC є створення невеликого набору синтетичних зображень, кожне з яких відноситься до одного з C класів. Цей набір даних можна позначити аналогічно до початкового набору даних, а саме $S = \{(s_i, y_i)\}_{i=1}^{|S|}$, де $s \in \mathbb{R}^d$ є синтетичним зображенням, а y класом до якого це зображення відноситься. До того ж початковий набір даних є набагато більшим ніж

набір синтетичних зображень, тобто $|S| \ll |\mathcal{T}|$. Подібно до $\theta^{\mathcal{T}}$ формулу знаходження параметрів можна виразити наступним чином:

$$\theta^S = arg_\theta min\, \mathcal{L}^S(\theta)$$

де $\mathcal{L}^S(\theta) = \frac{1}{|S|}\Sigma_{(s,y)\in S}\ell(\phi_\theta(s), y)$ — кросс-ентропічна функція втрат, в якій параметри θ^S є мінімізованими.

Метою методу дистиляції даних DC є пошук оптимального набору синтетичних зображень $S^* = arg_S min\, \mathcal{L}^{\mathcal{T}}(\theta^S(S))$ за умови, що $\theta^S(S) = arg_\theta min\, \mathcal{L}^S(\theta)$, для якого модель ϕ_{θ^S}, навчена на цих синтетичних зображень, мінімізує втрати при навчанні порівняно з початковими вхідними даними.

Метою є навчити модель ϕ_{θ^S} на наборі синтетичних даних S, таким чином, щоб дана модель досягала не лише подібної до $\phi_{\theta^{\mathcal{T}}}$ продуктивності, а й також збігалися до подібного розв'зку в параметричному просторі, тобто $\theta^S \approx \theta^{\mathcal{T}}$. Власне алгоритм мінімізації різниці значень двух параметрів θ^S та $\theta^{\mathcal{T}}$ можна сформулювати таким чином:

$$min_S D(\theta^S, \theta^{\mathcal{T}})$$

де D — це функція відстані між двома параметрами.

В min_S ціллю є знаходження оптимального набору синтетичних зображень для однієї моделі ϕ_{θ^S} з початковою ініціалізацією вагів θ_0, однак справжньою метою є створення таких синтетичних зображень, які можуть працювати з розподілом випадкових ініціалізацій P_{θ_0}. Отже, рівняння можна виразити наступним чином:

$$min_S\, \mathbb{E}_{\theta_0 \sim P_{\theta_0}}[D(\theta^S(\theta_0), \theta^{\mathcal{T}}(\theta_0))]$$

в якому $\theta^S(S) = arg_\theta min\, \mathcal{L}^S(\theta(\theta_0))$ та $\theta^{\mathcal{T}} = arg\, min_\theta\, \mathcal{L}^{\mathcal{T}}(\theta(\theta_0))$.

А власне процесс неповної оптимізації параметрів θ^S можна виразити формулою:

$$\theta^S(S) = opt\text{-}alg_\theta(\mathcal{L}^S(\theta), \varsigma)$$

в якій $opt\text{-}alg$ є певним оптимізаційним процесом з фіксованим числом кроків ς.

Але при застосуванні даного підходу можуть з'явитися дві можливі проблеми, а саме відстань між θ^T та проміжними значеннями параметрів θ^S може бути завеликою в просторі параметрів, що може зумовлювати складність її досягнення, а також обмеженої кількості кроків може бути недостатньо для того, щоб досягти оптимального результату.

Для того, щоб позбутися цих проблем, було запропонований новий підхід, що базується на навчальній програмі. В цьому підході основна ідея полягала в тому, що параметри θ^S повинні бути близькими за значеннями до остаточних параметрів θ^T, але й також слідувало аналогічним шляхом до параметрів θ^T на кожній ітерації t, що дозволяє зробити процес оптимізації більш керованим та ефективніше використовувати неповний оптимізатор. Цей процес можна виразити за допомогою формули:

$$min_S \, \mathbb{E}_{\theta_0 \sim P_{\theta_0}} \left[\sum_{t=0}^{T-1} D(\theta_t^S, \theta_t^T) \right]$$

за умови, що $\theta_{t+1}^S(S) = opt\text{-}alg_\theta(\mathcal{L}^S(\theta_t^S), \varsigma^S)$, а також $\theta_{t+1}^T = opt\text{-}alg_\theta(\mathcal{L}^T(\theta_t^T), \varsigma^T)$, де T — це загальна кількість ітерацій, а ς^S та ς^T це номери оптимізаційного кроку для θ^S та θ^T відповідно. Параметри θ_{t+1}^S можуть успішно відслідковувати θ_{t+1}^T, оновлюючи набір синтетичних зображень S, а мінімізувати відстань між $D(\theta_t^S, \theta_t^T)$ майже до нуля тобто $D(\theta_t^S, \theta_t^T) \approx 0$.

У випадку одного кроку оптимізації градієнтного спуску для алгоритму $opt\text{-}alg$ правило оновлення параметрів є наступним:

$$\theta_{t+1}^{S} \leftarrow \theta_t^{S} - \eta_\theta \nabla_\theta \mathcal{L}^{S}(\theta_t^{S})$$
$$\theta_{t+1}^{\mathcal{T}} \leftarrow \theta_t^{\mathcal{T}} - \eta_\theta \nabla_\theta \mathcal{L}^{\mathcal{T}}(\theta_t^{\mathcal{T}})$$

де η_θ — це параметр швидкості навчання.

Процес мінімізації можна скоротити, грунтуючись на тому, що $D(\theta_t^{S}, \theta_t^{\mathcal{T}}) \approx 0$ і він матиме наступний вигляд:

$$min_S \, \mathbb{E}_{\theta_0 \sim P_{\theta_0}} \left[\sum_{t=0}^{T-1} D(\nabla_\theta \mathcal{L}^{S}(\theta_t), \nabla_\theta \mathcal{L}^{\mathcal{T}}(\theta_t)) \right]$$

Отже, глибока нейрона мережа з параметрами θ, яка навчена на синтетичному наборі даних S, оптимізована таким чином, що відстань між градієнтами врат на початковому тренувальному наборі даних та синтетичному, тобто $\mathcal{L}^{\mathcal{T}}$ та $\mathcal{L}^{S}$ відповідно, була мінімальною. Іншими словами, робота алгоритму зводиться до узгодження градієнтів реальних та синтетичних втрат під час навчання шляхом оновлення синтетичних зображень набору даних S. Цей підхід має суттєву перевагу над методом Ванга DD, тому що він не вимагає розгортання рекурсивного обчислювального графа для попередніх параметрів θ, що зменшує навантаження на обчислювальні ресурси. Наслідком цього є пришвидшена оптимізація, яка більш ефективно використовує пам’ять обчислювальної машини, та дозволяє алгоритму краще масштабуватись до сучасних глибоких нейроних мереж.

Обчислення мінімізації втрат D вимірюється шляхом обчислення відстані між градієнтами θ для функцій втрат $\mathcal{L}^{\mathcal{T}}$ та $\mathcal{L}^{S}$. Коли нейрона мережа ϕ_θ є багатошаровою, градієнти відповідають набору двовимірних $out \times in$ та чотирьохвимірних вагів $out \times in \times h \times w$ для кожного об’єднувального шару та шару згортки, в якому out, in, h, w — кількість вихідних та вхідних каналів, висота та ширина ядра відповідно. Втрати можна розкласти в суму втрат для кожного шару і виразити за допомогою формули:

$$D(\nabla_\theta \mathcal{L}^S, \nabla_\theta \mathcal{L}^\mathcal{T}) = \Sigma_{l=1}^{L} d(\nabla_{\theta^{(l)}} \mathcal{L}^S, \nabla_{\theta^{(l)}} \mathcal{L}^\mathcal{T})$$

в якій l — це індекс шару, L — номер шару з вагами, а також відстаню:

$$d(A, B) = \sum_{i=1}^{out} (1 - \frac{A_{i.} \cdot B_{i.}}{||A_{i.}|| ||B_{i.}||})$$

де $A_{i.}$ та $B_{i.}$ — це вектори градієнтів, які відповідають кожному вихідному вузлу i, а також мають розмірність in для вагів об'єднувального шару, і $in \times h \times w$ для вагів згорткового шару.

Власне алгоритм дистиляції даних DC можна зобразити в текстовому форматі. Спочатку визначається кількість ітерацій алгоритму K, і для кожної з них створюється початкова випадкова ініціалізація вагів θ, після якої використовується нейрона мережа ϕ_θ для того, щоб обчислити втрати для початкового навчального набору даних $\mathcal{L}^\mathcal{T}$ і синтетичного набору даних $\mathcal{L}^S$, а також їх оновлені градієнти θ. Після цього застосовується алгоритм оптимізації $opt-alg_S$ синтетичного набору даних S з метою наближення значень градієнтів $\nabla_\theta \mathcal{L}^\mathcal{T}$ та $\nabla_\theta \mathcal{L}^S$, використовуючи ς_s кроків градієнтного спуску з коефіцієнтом швидкості навчання η_S. Далі застосовується оптимізаційний алгоритм $opt-alg_\theta$ для оновлення ваг θ на оновленому в попередньому кроці наборі синтетичних даних S, зі швидкістю навчання η_θ, за ς_θ кроків градієнтного спуску, при цьому мінімізуючи втрати $\mathcal{L}^S$. Також варто зауважити, що під час роботи алгоритму створюється початкова і синтетична пара підмножин з наборів даних $\mathcal{T}$ та S відповідно, які містять дані лише з одного класу, що дозволяє зменшити використання пам'яті під час навчання моделі та не призводить до додаткових обчислювальних витрат.

Нижче представлений більш стислий варіант, записаний з використанням псевдокоду.

Алгоритм DC:

***Вхідні дані*:** початковий набір даних $\mathcal{T}$, нейрона мережа ϕ_θ, кількість кроків зовнішнього циклу K, кількість кроків внутрішнього циклу T, кількість кроків для оновлення ваг ς_θ та синтетичних зображень ς_s.

1: **for** $k = 0, \ldots, K-1$ **do**:

2: Створити набір синтетичних зображень S для C класів

3: Виконати ініціалізацію початкових вагів θ_0, де $\theta_0 \sim P_{\theta_0}$

4: **for** $t = 0, \ldots, T-1$ **do**:

5: **for** $c = 0, \ldots, C-1$ **do**:

6: Створити пару підмножин $B_c^{\mathcal{T}} \sim \mathcal{T}$ та $B_c^S \sim S$,
які належать одному класу c.

7: Обчислити функції втрат:
$\mathcal{L}_c^{\mathcal{T}} = \frac{1}{|B_c^{\mathcal{T}}|} \Sigma_{(x,y)\in B_c^{\mathcal{T}}} \ell(\phi_\theta(x), y)$ $\mathcal{L}_c^S = \frac{1}{|B_c^S|} \Sigma_{(x,y)\in B_c^S} \ell(\phi_\theta(s), y)$

8: Оновити синтетичні зображення:
$S_c \leftarrow opt\text{-}alg_S D(\nabla_\theta \mathcal{L}_c^S(\theta_t), \nabla_\theta \mathcal{L}_c^S(\theta_t)), \varsigma_S, \eta_S)$

9: Оновити параметри $\theta_{t+1} \leftarrow opt\text{-}alg_\theta(\mathcal{L}^S(\theta_t), \varsigma_\theta, \eta_\theta)$

***Вихідні дані*:** набір синтетичних зображень S

Підсумовуючи загальний опис алгоритму DC, можна сказати, що даний метод дозволяє ефективно зменшувати початковий набор даних, забезпечує репрезентатність даних, а також покращує продуктивність навчання моделі. Він є одним з фундаментальних методів градієнтної дистиляції даних, саме тому було вирішено використати його в подальшому дослідженні, запропонувавши його модифікацію.

2.2.3 Kernel Inductiong Points (KIP)

Наступний метод був запропонований науковцем Нгуєном та іншими, який отримав назву індукційні точки ядра (англ. Kernel Inducing Points), або скорочено KIP [14].

Цей метод використовував градієнтний спуск для того, щоб мінімізувати значення функції втрат регресії відносно навчальних даних під час кожної ітерації за допомогою випадкових ядер та цільових частин тренувального набору даних.

Поштовхом до даного створення даного методу були проблеми в існуючих методах навчання, а саме проблема потреби мати великі набори даних задля того, щоб досягти високої точності моделі, а також проблеми масштабування моделі.

Метою методу дистиляції даних KIP є пошук невеликого набору даних $\widehat{D}$, маючи початковий навчальний набір даних D з розподілу P. Набір даних $\widehat{D}$ є ε-наближеним до D відносно $\left(A, \hat{A}, l, P\right)$, тобто наближенням з точністю ε. Де A і $\hat{A}$ є алгоритмами навчання, які отримують на вхід набори даних D та $\widehat{D}$ відповідно, l – – це функція втрат.

Самі алгоритми навчання є алгоритмами, які побудовані на основі хребтової регресії ядра.

Хребтова регресія ядра (англ. Kernel ridge regression) — це алгоритм хребтової регресії [15], метою якого є знаходження лінійних функцій моделі, яка моделює залежності між неперервними коваріантами змінними та змінними відгуку. В контексті машинного навчання він використовується для зменшення кількості ваг моделі та зниження ризику перенавчання.

Припускаючи, що $A = \hat{A}$ виражається певна формула наближення:

$$\mathbb{E}_{(x,y)\in\mathcal{P}}\ell(\tilde{A}_{\tilde{D}}(x),y) \approx \mathbb{E}_{(x,y)\in D}\ell(\tilde{A}_{\tilde{D}}(x),y)$$

У загальних алгоритмів $\hat{A}$ зовнішня оптимізація для набору даних $\widehat{D}$ потребує багато ресурсів для обчислення, тому що вона пов'язана з похідними другого порядку.

Алгоритм KIP використовує хребтову регресію ядра для навчання ε-наближених наборів даних. Для ядра K функція втрат для методу хребтової регресії хребта, навчена на опорному наборі даних (X_s, y_s) та перевірена на цільовому наборі даних (X_t, y_t) має вигляд:

$$L(X_s, y_s) = \frac{1}{2}\left|\left|y_t - K_{X_tX_s}(K_{X_sX_s} + \lambda I)^{-1}y_s\right|\right|^2$$

де $\lambda > 0$ та є фіксованим параметром регуляризації. Даний алгоритм складається з оптимізації функції втрат L, яка залежить від опорного набору даних (X_s, y_s).

Саму ідею алгоритму можна схематично зобразити за допомогою послідовності кроків:

Алгоритм KIP:

***Вхідні дані*:** Цільовий набір даних (X_t, y_t) включно з ядром або родиною ядер.

1: Ініціалізувати допоміжний набір даних (X_s, y_s)

2: **while** не збігаються **do**:

3: Вибрати випадкове ядро. Вибрати випадкову підмножину $(\bar{X}_s, \bar{y}_s)$ з допоміжного набору даних. Вибрати випадкову підмножину $(\bar{X}_t, \bar{y}_t)$ з цільового набору даних.

4: Порахувати втрати при хребтовій регресії ядра $L(X_s, y_s)$, використовуючи вибране ядро та вибраний допоміжний і цільовий набори даних.

5: Зворотнє розповсюдження через $\bar{X}_s$ та оновлення допоміжного набору даних (X_s, y_s) шляхом оновлення підмножини $(\bar{X}_s, \bar{y}_s)$.

***Вихідні дані*:** Натренований допоміжний набір (X_s, y_s)

2.2.4 Label solving (LS)

Метод дистиляції даних під назвою метод розв'язання міток (англ. Label solving) , або скорочено LS, був також запропонований Нгуєном та іншими в тій само науковій роботі, що і метод KIP. На відміну від KIP, де опорний набір даних навчається за допомогою градієнтного спуску, метод розв'язання міток використовує трохи інший підхід. Даний метод індукційних точок базується на знаходженні мінімальної функції втрат $L(X_s, y_s)$ відносно опорних міток y_s при фіксованих навчальних даних X_s. Так як функція втрат є квадратичною в y_s, є можливість знайти отримані мітки за допомогою формули:

$$y_s^* = \Phi_0(K_{X_t X_s}(K_{X_s X_s} + \lambda I)^{-1})y_t$$

де y_s^* є мінімально-нормальним розв'язком серед $L(X_s, y_s)$, а Φ_0 є певною псевдо-інверсною операцією. Отримані мітки називають розв'язаними мітками, що і дало назву даному методу дистиляції даних.

Варто зазначити, що обидва методи, а саме LS та KIP, є доволі ресурсо залежними. Для того, щоб виконати процес дистиляції даних, їм потрібні великі обчислювальні потужності. Наукова робота була продовжена Нгуєном та іншими в 2022 році та була спрямована на покращення методів дистиляції даних KIP та LS за допомогою використання нової системи мета-навчання на основі ядра з використанням нескінченно широких згорткових нейронних мереж для вирішення проблеми дистиляції даних [16].

Однак в ній науковцям так і не вдалося вирішити проблему складності обчислень, тому вони визнають, що виконання даних алгоритмів потребує тисячі годин обчислювальних ресурсів. Методи дистиляції даних LS та KIP є надзвичайно

важкими навіть для “іграшкових” тестових задач. Тому було вирішено розглянути дані методи лише для загального ознайомлення.

Вони безумовно внесли значний вклад в розвиток розділу дистиляції даних, але в межах даної наукової роботи було вирішено сфокусуватись саме на градієнтному методі дистиляції даних DC.

РОЗДІЛ 3: Дослідження різних способів ініціалізації зображень для методу DC

3.1 Нейронна мережа та набір даних

3.1.1 Згорткова нейронна мережа ConvNet

В межах даної наукової роботи було вирішено взяти глибоку згорткову нейрону мережу ConvNet [17], в якості інструменту для дослідження дистиляції даних, при вирішенні проблеми класифікації зображень.

Дана згорткова нейронна мережа містить модульну архітектуру малопоширеного навчання з блоками, які дублюються кількості D, кожен з яких має згортковий шар з W фільтром розмірності 3×3, шар нормалізації N, шар активації A, а також об'єднувальний шар P, які можна позначити як $[W, N, A, P] \times D$.

Стандартна нейрона мережа ConvNet складається з трьох повторюваних блоків, кожен з яких має шар згортки з ста двадцятьма восьми фільтрами розмірності 3×3, за якими йдуть шар нормалізації InstanceNorm, модуль з функцією активації ReLU, а також об'єднувальний шар AvgPooling. Ця послідовність блоків завершується щільним шаром. Під час навчання для початкової ініціалізації вагів використовується випадкова ініціалізація вагів.

Повна схема моделі нейронної мережі ConvNet зображена в Додатку А.

Як було зазначено раніше, ця згорткова нейронна мережа широко використовується в задачах малопоширеного навчання, тобто галуззю, яка займається проблемою навчання розпізнаванню нових класів на основі невеликої кількості прикладів з мітками. В традиційному машинному навчанні модель навчається на великому наборі даних з прикладами з мітками, а потім використовується для прогнозування нових даних, які не зустрічались у навчальному наборі даних. Однак метою малопоширеного навчання є навчання

моделі розпізнавати нові класи за допомогою дуже обмеженої кількості прикладів з мітками. Саме тому згорткова нейронна мережа ConvNet підходить для використання при застосуванні методу дистиляції зображень в контексті класифікації зображень, бо кількість дистильованих даних іноді є досить обмеженою, задля того, щоб не витрачати дуже багато часу на кожен з експериментів.

Під час навчання нейроної мережі було вирішено використовувати саме випадкову ініціалізацію початкових вагів, а також алгоритм стохастичного градієнтного спуску (англ. Stochastic gradient descent) або скорочено SGD [18] задля оптимізації штучної нейронної мережі ConvNet.

3.1.2 Набір даних CIFAR-10

У наші дні набір даних, який має назву CIFAR-10, є ледь не одним з найбільш популярних наборів даних у відкритому доступі, що використовується для навчання нейронних мереж, особливо для задач, пов’язаних з класифікацією зображень. Основною метою датасету CIFAR-10 є його використання в якості своєрідного критерію оцінки успішності навчання моделей в навчальних та науково-дослідницьких проектів. Цей датасет складається з набору з шістдесяти тисяч квадратних кольорових зображень, що мають розмір тридцять два в довжину і тридцять два пікселя в ширину. Тобто кожне зображення є своєрідною квадратною сіткою розмірами тридцять два пікселя, кожен з яких має три кольорові канали RGB.

Саме RGB розшифровується як "Red Green Blue" (червоний, зелений, синій) і відноситься до моделі кольору, що використовується для відображення кольорів на електронних дисплеях, таких як екрани комп'ютерів і телевізорів. У цій моделі кожен колір представлений значенням від 0 до 255, де 0 — відсутність кольору, а 255 — максимальна інтенсивність. Комбінуючи різні рівні червоного, зеленого та синього, можна створити широкий спектр кольорів. Ця модель кольорів широко

використовується в цифровій обробці зображень, графічному дизайніб в машиному навчанні та інших галузях.

З загальних шістдесяти тисяч зображень п’ятьдесят тисяч є тренувальними зображеннями, а десять тисяч – тестовими. Зображення розділені на десять різних класів, пронумерованих від нуля до дев’яти. Ці класи є доволі стандартними і вони характеризують певні живі та неживі предмети, а саме: птах, літак, кіт, автомобіль, олень, жаба, пес, кінь, вантажівка та корабель.

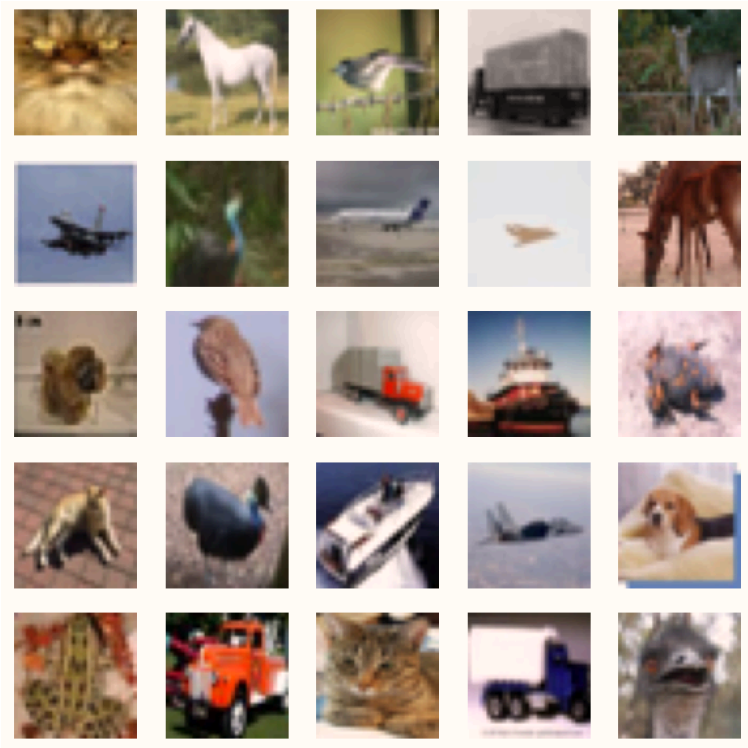


Рисунок 3.1.2.1 Приклад зображень набору даних CIFAR-10

Оцінюючи даний перелік, можна одразу дійти до висновку, що цей датасет не зовсім підходить для вирішення проблеми класифікації зображень в справжніх, не тестових проектах, де кількість класів повинна бути набагато більша. Проте для експериментальних цілей даного наукового дослідження, цей набір даних є майже ідеальний.

Його простота дозволяє за обмежений час робити більше експериментів, використовуючи різні методи дистиляції даних.

Саме завдяки цим особливостям набір даних CIFAR-10 був обраний в якості основного набору даних для цього дослідження, він чудово підходить для вирішення задачі класифікації зображень та повністю підтримується архітектурою згорткової нейроної мережі ConvNet.

3.2 Аналіз результатів вибраних методів дистиляції даних

В межах даної наукової роботи було вирішено перевірити, які саме результати можна отримати, використовуючи методи дистиляції даних, що описані в науковій роботі. За основу були взяті чотири класичні методи вибору підмножин, а саме Random, Herding, K-Center та Forgetting, а також градієнтний метод дистиляції даних DC.

Було вирішено проводити есперименти саме з такою кількістю зображень: 10, 100, 500. Тобто в першому експерименті на кожен клас припадає по одному зображенню, а частка загального набору даних складає лише 0.02 відсотка, в другому експерименті на кожен клас припадає по 10 зображень, що в загальному становить 0.2 відсотки, а в третьому — 50 зображень з загальною часткою в 1 відсоток.

Для кожного з вибраних методів, беручи до уваги кількість дистильованих зображень, було проведено по п’ять експериментів, в кожному з яких модель ConvNet навчалась на дистильованих даних набору CIFAR-10 протягом трьохста епох та однієї тисячі кроків ітерації алгоритму.

Кількість зображень	Відсоток зображень	Random	Herding	K-Center	Forgetting	DC	Цілий датасет
10	0.02	14.5±2.0	21.5±1.2	21.5±1.3	13.5±1.2	28.3±0.5	84.8±0.1
100	0.2	26.0±1.2	31.6±0.7	24.7±0.9	23.3±1.0	44.9±0.5	
500	1	43.4±1.0	40.4±0.6	27.0±1.4	23.3±1.1	53.9±0.5	

Як видно з отриманих результатів, метод DC значно переважає інші методи дистиляції даних, експериментальним шляхом доводячи, що за допомогою градієнтних методів дистиляції даних можна отримати кращі результати ніж з використанням методів вибору підмножини.

3.3 Метод DC з використанням різних способів початкової ініціалізації зображень

В оригінальному методі DC використовується випадкова початкова ініціалізація зображень на етапі створення набору синтетичних зображень, тому було вирішено спробувати різні способи початкової ініціалізації зображень, для того щоб перевірити, яким чином це впливає на загальний результат.

Для дослідження різних способів початкової ініціалізації зображень були взяті методи вибору підмножини, а саме Herding, Forgetting та K-Center. На відміну від ініціалізації зображень в оригінальнгому методі DC, в якому випадковим чином вибираються зображення з набору даних, обрані методи вибору підмножини, вибирають найбільш релевантні зразки, базуючись на власних алгоритмах. Тому набори синтеничних зображень, які отримуються під час початкової ініціалізації зображень, суттєво відрізняються від наборів синтетичних зображень, отриманих в результаті випадкової ініціалізації зображень, представленій в оригінальному методі DC.

Оновлену версію алгоритму DC, яка містить різні методи початкової ініціалізації зображень, представлено схематично нижче.

Алгоритм DC з різною ініціалізацією зображень:

***Вхідні дані*:** початковий набір даних $\mathcal{T}$, нейрона мережа ϕ_θ, кількість кроків зовнішнього циклу K, кількість кроків внутрішнього циклу T, кількість кроків для оновлення ваг ς_θ та синтетичних зображень ς_s.

1: **for** $k = 0, \ldots, K-1$ **do**:

2: Створити початковий набір синтетичних зображень S для C класів, використовуючи один з методів вибору підмножин (K-Center, Forgetting, Herding) або випадкову ініціалізацію зображень

3: Виконати ініціалізацію початкових вагів θ_0, де $\theta_0 \sim P_{\theta_0}$

4: **for** $t = 0, \ldots, T-1$ **do**:

5: **for** $c = 0, \ldots, C-1$ **do**:

6: Створити пару підмножин $B_c^{\mathcal{T}} \sim \mathcal{T}$ та $B_c^S \sim S$,
які належать одному класу c.

7: Обчислити функції втрат:
$\mathcal{L}_c^{\mathcal{T}} = \frac{1}{|B_c^{\mathcal{T}}|} \Sigma_{(x,y)\in B_c^{\mathcal{T}}} \ell(\phi_\theta(x), y)$ $\mathcal{L}_c^S = \frac{1}{|B_c^S|} \Sigma_{(x,y)\in B_c^S} \ell(\phi_\theta(s), y)$

8: Оновити синтетичні зображення
$S_c \leftarrow opt\text{-}alg_S D(\nabla_\theta \mathcal{L}_c^S(\theta_t), \nabla_\theta \mathcal{L}_c^S(\theta_t)), \varsigma_S, \eta_S)$

9: Оновити параметри $\theta_{t+1} \leftarrow opt\text{-}alg_\theta(\mathcal{L}^S(\theta_t), \varsigma_\theta, \eta_\theta)$

***Вихідні дані*:** набір синтетичних зображень S

Питання ініціалізації зображень в методі DC є доволі важливим, тому що в момент ініціалізації зображень відбувається відбір початкових зображень для їх подальшої трансформації в завершений набір даних синтетичний. У Додатку Б. на Рис. 1 можна побачити 10 відібраних за допомогою методу Herding зображень для кожного класу набору даних CIFAR-10.

Під час роботи алгоритму DC зображення постійно модифікуються, і в результаті утворюється набір дистильованих синтетичних зображень, які можна використовувати для навчання моделі (Додаток В. Рис. 2).

Отже, в межах даного дослідження були запропоновані різні варіанти початкової ініціалізації для зображень, таким чином модифікувавши оригінальний метод DC, а також було зроблено аналіз отриманих результатів, який більш детально описаний в наступному розділі.

3.4 Аналіз використання різних способів ініціалізації зображень в методі DC

3.4.1 Результати різних способів ініціалізації зображень

Для проведення експериментів з використанням різних способів ініціалізації зображень було вирішено взяти таку кількість дистильованих зображень, а саме: 100, 250 та 500.

Тобто в першому експерименті на кожний клас припадає по 10 зображнь, а частка загального набору даних складає 0.2 відсотка, в другому експерименті на кожен клас припадає по 25 зображень, що в загальному становить 0.5 відсотки, а в третьому — 50 зображень з загальною часткою в 1 відсоток.

В експериментах з використанням різних способів ініціалізації зображень було проведено по п’ять експериментів, в кожному з яких модель ConvNet навчалась на дистильованих даних набору CIFAR-10 протягом трьохста епох та п'ятиста кроків ітерації алгоритму, що в два рази менше ніж в попередніх експериментах, в яких досліджувались результати різних методів дистиляції даних. Цей вимушений крок був зроблений з метою зменшення часу навчання.

Кількість зображень	Відсоток зображень	Random Init	Herding Init	K-Center Init	Forgetting Init
100	0.2	44.07±0.27	44.66±0.26	44.18±0.69	43.67±0.69
250	0.5	50.89±0.37	50.46±0.13	50.55±0.12	50.44±0.32
500	1	53.35±0.35	53.19±0.43	52.76±0.44	53.6±0.3

Як видно по результатам, отриманим після навчання моделі на дистильованих даних, використовуючи різні способи ініціалізації зображень, важко провести аналіз з визначення найкращого способа ініціалізації. Тому було вирішено додатково провести t-тест, тобто використати метод для порівняння середніх значень двох вибірок і визначення, чи є ці різниці статистично значущими. Тест базується на

обчисленні t-статистики, яка враховує різницю між середніми значеннями вибірок і їхніми стандартними відхиленнями.

Також в межах дослідження було вирішено порівняти часові витрати методу DC при використанні різних способів ініціалізації зображень і записати їх в таблицю.

Кількість зображень	Random Init	Herding Init	K-Center Init	Forgetting Init
100	94 хв 24 с	100 хв 38 с	95 хв 7 с	100 хв 9 с
250	171 хв 10 с	182 хв 52 с	178 хв 7 с	182 хв 11 с
500	329 хв 6 с	328 хв 12 с	329 хв 11 с	323 хв 27 с

Згідно з часовими результатами для експериментів перших двох експериментів беззаперечним лідером є метод випадкової ініціалізації зображень. При його використанні алгоритм дистиляції даних DC працює 94 хв 24 с та 171 хв 10 с, для ста та двохсот п’ятдесяти зображень відповідно. Для п’ятиста зображень не все так однозначно. Виходячи з наявних даних можна побачити, що методи ініціалізації зображень Herding, K-Center та Forgetting по часовим результатам зрівнюються з випадковою ініціалізацією, в той час, як метод Forgetting взагалі отримує найкращий часовий результат — 323 хв 27 с. Можна припустити, що зі збільшенням кількості зображень, методи ініціалізації зображень, які використовують вибір підмножин, будуть отримувати кращі результати. Але для спростування або підтвердження цього припущення потрібно більше тестових даних.

3.4.2 T-тест результатів різних способів ініціалізації зображень

За допомогою t-тесту [19] було проведено порівняння між різними методами ініціалізації зображень для різної кількості дистильованих зображень. Нижче наведені відповідні значення t-статистики та p-значення для кожного порівняння.

Для 100 дистильованих зображень були отримані наступні результати порівняння різних методів ініціалізації зображень:

Перший метод	Другий метод	t-статистика (t- statistic)	p-значення (p-value)
Random	Herding	-2.73	0.053
Random	K-Center	-0.257	0.810
Random	Forgetting	0.935	0.403
Herding	K-Center	1.127	0.323
Herding	Forgetting	2.325	0.081
K-Center	Forgetting	0.905	0.417

Для 250 дистильованих зображень були отримані наступні результати:

Перший метод	Другий метод	t-статистика (t- statistic)	p-значення (p-value)
Random	Herding	1.90	0.130
Random	K-Center	1.51	0.205
Random	Forgetting	1.593	0.186
Herding	K-Center	-0.881	0.428
Herding	Forgetting	0.1	0.925
K-Center	Forgetting	0.557	0.607

Для 500 дистильованих зображень були отримані наступні результати:

Перший метод	Другий метод	t-статистика (t- statistic)	p-значення (p-value)
Random	Herding	0.500	0.643
Random	K-Center	1.82	0.143
Random	Forgetting	-0.939	0.4
Herding	K-Center	1.21	0.293
Herding	Forgetting	-1.355	0.247
K-Center	Forgetting	-2.7	0.052

На основі отриманих результатів можна зробити наступні висновки для ста зображень: порівняння не показують статистично значущих відмінностей результатів, тобто нульову гіпотезу про рівність середніх значень різних методів ініціалізації зображень відкинути не можна.

Для двохсот п’ятдесяти зображень усі порівняння методів ініціалізації зображень не показують статистично значущих різниць між ними. Отже, для цієї вибірки нульову гіпотезу також неможливо заперечити.

Для п’ятиста зображень усі порівняння не показують статистично значущих відмінностей між результатами, тобто не можна заперечити нульову гіпотезу для них.

В підсумку можна сказати, що на основі проведених тестів не можливо встановити статистично значущих відмінностей між досліджуваними методами ініціалізації зображень для тестових вибірок. Отже, можна дійти до висновку, що нульову гіпотезу про рівність середніх значень за наявними вибірками не можна відкинути. Це означає, що немає достатніх доказів, щоб стверджувати, що середні значення результатів експериментів відрізняються між собою.

ВИСНОВКИ

Дистиляція даних є важливим напрямком в машинному навчанні, оскільки вона допомагає зменшити часові й обчислювальні витрати на навчання моделі. Також вона відкриває нові можливості для ефективного використання моделей у різних сценаріях, сприяє подальшому розвитку та вдосконаленню алгоритмів машинного навчання.

В результаті виконаної роботи були розглянуті основні проблеми, з якими стикаються великі набори даних, та способи зменшення складності моделі нейронної мережі. Також в науковій роботі було зроблено огляд різних методів дистиляції даних для вирішення проблеми класифікації зображень, а саме: Forgetting, Herding, K-Center, DC, KIP, LS, та проведено аналіз ефективності вибраних методів. В межах даної роботи було досліджено вплив різних способів ініціалізації зображень на результати методу DC та проаналізовано отримані результати експериментів.

Під час аналізу вибраних методів дистиляції даних виявилось, що градієнтні методи дистиляції даних переважають по продуктивності методи вибору підмножин та показують кращі результати в усіх проведених експериментах.

Експериментальним шляхом було доведено, що ефективність методу DC з обраними способами ініціалізації зображень, а саме: Random, Forgetting, Herding та K-Center відрізняється в межах статистичної похибки, виходячі з наявних тестових даних. Було зроблено t-тест отриманих результатів, який підтвердив, що нульову гіпотезу про рівність середніх значень цих результатів відкинути не можна.

З урахуванням отриманих результатів у майбутньому можливими напрямками розширення даної наукової роботи є продовження дослідження впливу різних способів початкової ініціалізації на результативність методу DC з метою покращення результатів методу в порівнянні з оригінальною випадковою ініціалізацією зображень.

Список літератури

[1] Zhao, B., Mopuri, K.R. and Bilen, H., 2020. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*.

[2] The CIFAR-10 dataset [Електронний ресурс] – Режим доступу до ресурсу: https://www.cs.toronto.edu/~kriz/cifar.html.

[3] Sagiroglu, S. and Sinanc, D., 2013, May. Big data: A review. In *2013 international conference on collaboration technologies and systems (CTS)* (pp. 42-47). IEEE.

[4] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H. and He, Q., 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, *109*(1), pp.43-76.

[5] Collins, B., Deng, J., Li, K. and Fei-Fei, L., 2008. Towards scalable dataset construction: An active learning approach. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part I 10* (pp. 86-98). Springer Berlin Heidelberg.

[6] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[7] Phillips, J.M., 2016. Coresets and sketches. *arXiv preprint arXiv:1601.00617*.

[8] Guo, C., Zhao, B. and Bai, Y., 2022, July. Deepcore: A comprehensive library for coreset selection in deep learning. In *Database and Expert Systems Applications: 33rd International Conference, DEXA 2022, Vienna, Austria, August 22–24, 2022, Proceedings, Part I* (pp. 181-195). Cham: Springer International Publishing.

[9] Toneva, M., Sordoni, A., Combes, R.T.D., Trischler, A., Bengio, Y. and Gordon, G.J., 2018. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*.

[10] Welling, M., 2009, June. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning* (pp. 1121-1128).

[11] Sener, O. and Savarese, S., 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

[12] Hinton, G., Vinyals, O. and Dean, J., 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

[13] Wang, T., Zhu, J.Y., Torralba, A. and Efros, A.A., 2018. Dataset distillation. *arXiv preprint arXiv:1811.10959*.

[14] Nguyen, T., Chen, Z. and Lee, J., 2020. Dataset meta-learning from kernel ridge-regression. *arXiv preprint arXiv:2011.00050*.

[15] Welling, M., 2013. Kernel ridge regression. *Max Welling's classnotes in machine learning*, pp.1-3.

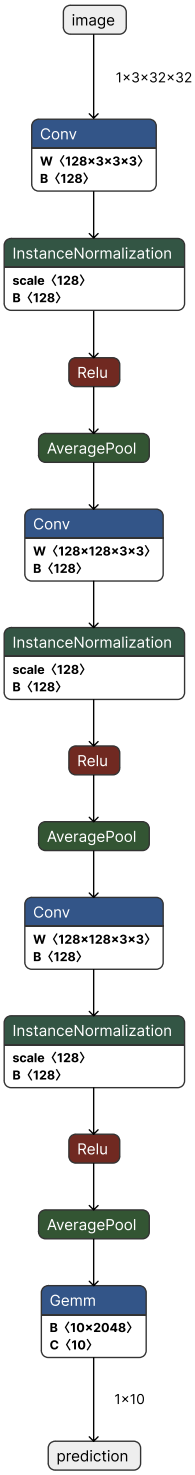
[16] Nguyen, T., Novak, R., Xiao, L. and Lee, J., 2021. Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, *34*, pp.5186-5198.

[17] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[18] Ruder, S., 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

[19] Sendecor, G.W. and Cochran, W.G., 1989. Statistical methods.

Додаток А. Схема моделі ConvNet



Додаток Б. 10 синтетичних зображень методу DC

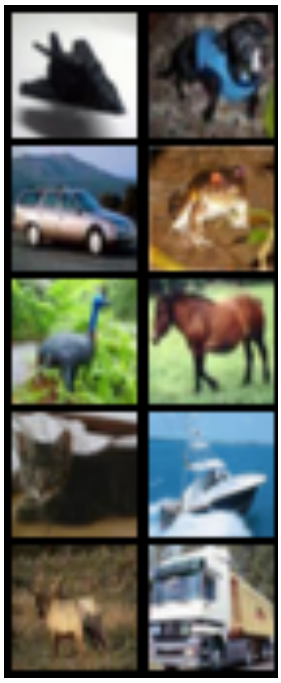


Рисунок 1 Випадкова ініціалізація зображень

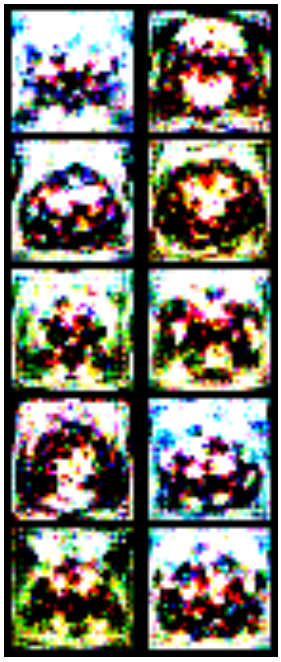


Рисунок 2 Синтетичні зображення після п’ятиста ітерацій

Додаток В. 100 синтетичних зображень методу DC

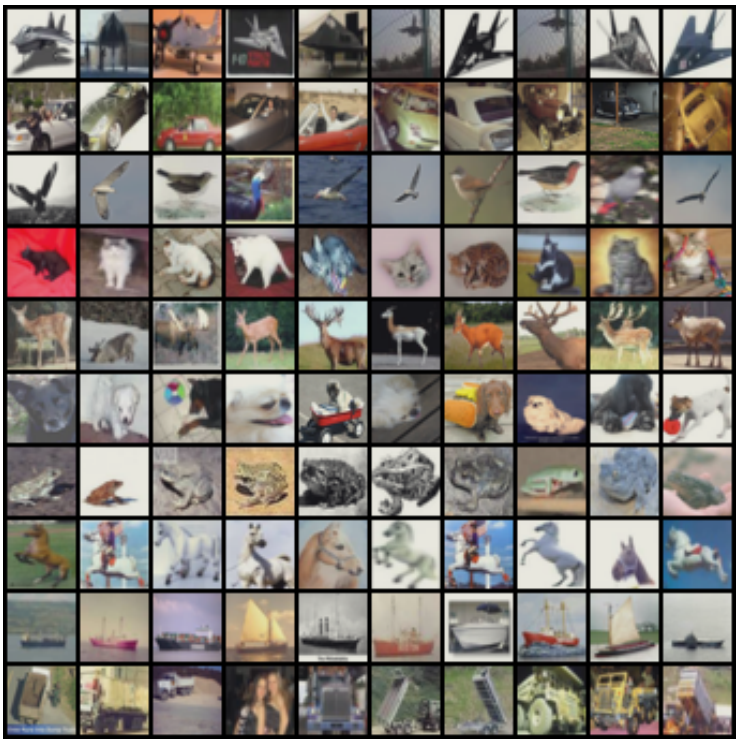


Рисунок 1 Ініціалізація зображень за допомогою методу Herding

Рисунок 2 Синтетичні зображення після п’ятиста ітерацій

Додаток Г. 250 синтетичних зображень методу DC

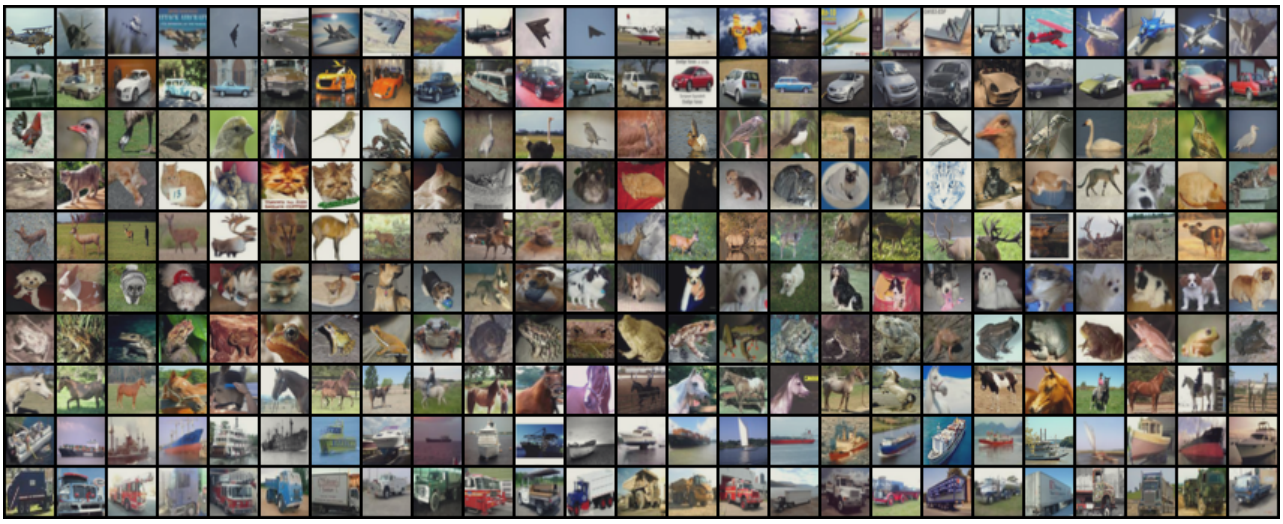


Рисунок 1 Ініціалізація зображень за допомогою методу Forgetting

Рисунок 2 Синтетичні зображення після п’ятиста ітерацій