НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«На правах рукопису»
УДК 004.056

«До захисту допущено»
Науковий керівник кафедри
_____ Іван ДИЧКА
«___»_____2020 р.

Магістерська дисертація

на здобуття ступеня магістра

освітньо-науковою програмою
«Інженерія програмного забезпечення комп'ютерних та інформаційно-пошукових систем»

зі спеціальності 121 Інженерія програмного забезпечення

на тему: «Алгоритмічно-програмний метод шифрування даних з використанням нейронних мереж»

Виконав:
студент ІІ курсу, групи КП-81мн
Северін Андрій Іванович _____

Керівник:
Доцент кафедри ПЗКС, к.т.н., доцент,
Онай Микола Володимирович _____

Консультант з нормоконтролю:
Доцент кафедри ПЗКС, к.т.н., доцент,
Онай Микола Володимирович _____

Рецензент:
Доцент кафедри СПіСКС, к.т.н., доцент,
Клятченко Ярослав Михайлович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2020 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – другий (магістерський)

Спеціальність (спеціалізація) – 121 «Інженерія програмного забезпечення»

Освітньо-наукова програма «Інженерія програмного забезпечення комп'ютерних та інформаційно-пошукових систем»

ЗАТВЕРДЖУЮ

Науковий керівник кафедри

_____ Іван ДИЧКА

«___»_____2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Северіну Андрію Івановичу

1. Тема дисертації «Алгоритмічно-програмний метод шифрування даних з використанням нейронних мереж», науковий керівник дисертації Онай Микола Володимирович, к.т.н., доцент, затверджена наказом по університету від «07» квітня 2020 р. №964-С

2. Термін подання студентом дисертації «15» травня 2020 р.

3. Об'єкт дослідження: процеси шифрування, дешифрування та генерації інформації з використанням нейронних мереж.

4. Предмет дослідження: методи, способи та моделі захисту наборів даних з використанням нейронних мереж.

5. Перелік завдань, які потрібно вирішити:
   - дослідити методи захисту приватних наборів даних;
   - провести аналіз методів, способів та моделей шифрування та генерації наборів даних;
   - розробити та дослідити модель шифрування даних з використанням нейронних мереж;
   - запропонувати метод шифрування приватних наборів даних;
   - вирішити задачу класифікації оригінальних та зашифрованих даних;
   - провести аналіз отриманих результатів.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:
   - схема роботи генеративної конкуруючої мережі;
   - модель шифрування даних з використанням генеративних конкуруючих нейронних мереж;
   - схема етапу формування нового набору даних запропонованим методом;
   - діаграма класів розробленої системи;

- експериментальні результати тривалості навчання модифікованої моделі шифрування даних;
- експериментальні результати точності вирішення задачі класифікації оригінальних та зашифрованих даних.

7. Орієнтовний перелік публікацій:
   - Стаття “Метод шифрування даних з використанням нейронних мереж”
   - Тези доповіді “Метод шифрування даних з використанням нейронних мереж”

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Онай М.В., к.т.н., доцент		

9. Дата видачі завдання «11» жовтня 2018 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Ґрунтовне ознайомлення з предметною галуззю	17.12.2018	
2.	Визначення структури магістерської дисертації; вивчення літератури, пошук додаткової літератури, патентний пошук	04.03.2019	
3.	Робота над першим розділом магістерської дисертації; проведення наукового дослідження	16.05.2019	
4.	Проведення наукового дослідження; робота над другим розділом магістерської дисертації; розроблення програмного забезпечення; підготовка матеріалів доповіді на конференції ПМК-2019	14.10.2019	
5.	Проведення наукового дослідження; робота над статтею за результатами наукового дослідження	15.12.2019	
6.	Проведення наукового дослідження; робота над третім та четвертим розділами магістерської дисертації	20.02.2020	
7.	Завершення роботи над основною частиною магістерської дисертації; підготовка ілюстративного матеріалу	16.04.2020	
8.	Оформлення текстової і графічної частини магістерської дисертації	13.05.2020	

Студент Андрій СЕВЕРІН

Науковий керівник Микола ОНАЙ

РЕФЕРАТ

Актуальність теми. Системи аналізу даних і засобів штучного інтелекту широко застосовуються у багатьох сферах людської діяльності. З кожним роком клас задач, де можна й доцільно їх застосовувати, постійно розширюється. Важливим елементом для побудови надійної і точної системи є наявність достатньої кількості даних для навчання. Однак, далеко не всі зібрані дані можуть бути використані для навчання у рішеннях з використанням штучного інтелекту, оскільки вони можуть містити значну кількість приватної інформації: секретної (наприклад, фінансові, військові дані), конфіденційної (ідентифікуючі дані: паспортні дані, реєстраційний номер облікової картки платника податків) або чутливої (медичні дані, які містять діагнози пацієнтів). За таких умов, розроблення методів захисту приватних наборів даних, що дозволять їх використання для навчання, є актуальним, оскільки кількість загальнодоступних даних є обмеженою.

Об’єктом дослідження є процеси шифрування, дешифрування та генерації інформації з використанням нейронних мереж.

Предметом дослідження є методи, способи та моделі захисту наборів даних з використанням нейронних мереж.

Мета роботи: розроблення алгоритмічно-програмного методу шифрування даних на основі генеративних конкуруючих нейронних мереж для використання зашифрованих приватних наборів даних в системах аналізу даних і штучного інтелекту.

Методи дослідження. В роботі використовуються методи математичного та програмного моделювання, методи оптимізації, методи штучного інтелекту.

Наукова новизна роботи полягає в наступному:

1. Запропоновано метод шифрування наборів даних, що дозволяє використовувати приватні дані в системах штучного інтелекту й аналізу даних, а саме при вирішенні задачі класифікації зображень, точність отриманих результатів на зашифрованих даних з відомим ключем шифрування лише на 2-4 % нижча, ніж точність класифікації оригінальних даних.

2. Модифіковано модель шифрування даних, шляхом використання двовимірних згорткових нейронних мереж, що дозволяє пришвидшити у 1,5 рази шифрування зображень, порівняно з класичною моделлю, яка призначена для шифрування окремих біт.

3. Розроблено архітектуру програмного забезпечення, особливістю якої є інкапсуляція алгоритму попередньої обробки вхідного набору даних, що дозволяє на основі шаблону «Стратегія» замінювати конкретну реалізацію в екземплярі класу класифікатор без застосування наслідування.

Практична цінність отриманих в роботі результатів полягає в тому, що запропонований метод шифрування даних дає можливість використовувати приватні набори даних у загальнодоступних системах штучного інтелекту шляхом їх шифрування. Модифікована модель шифрування дозволяє прискорити її навчання, а розроблене програмне забезпечення для шифрування й класифікації даних дозволяє користувачеві проводити аналіз даних, як на оригінальних даних, так і на зашифрованих.

Апробація роботи. Основні положення і результати роботи були представлені та обговорювались на XII науковій конференції магістрантів та аспірантів «Прикладна математика та комп’ютинг» ПМК-2019 (Київ, 13-15 листопада 2019 р.). За результатами роботи була опублікована наукова стаття у міжнародному науковому журналі «Радіоелектроніка, інформатика, управління» 2020 (випуск №2), що входить до наукометричної бази даних Web of Science.

Структура та обсяг роботи. Магістерська дисертація складається з вступу, чотирьох розділів, висновків та додатків.

У вступі надано загальну характеристику роботи, виконано оцінку сучасного стану проблеми, обґрунтовано актуальність напрямку досліджень, сформульовано мету дослідження й завдання, що включає відповідна науково-практична задача.

У першому розділі розглянуто основні методи захисту приватних наборів даних, які можуть бути використані при побудові систем аналізу даних і штучного інтелекту.

У другому розділі розглянуто й проаналізовано модель шифрування даних, яка використовує генеративні конкуруючі мережі; досліджено структуру згорткових нейронних мереж, які є основою для побудови генеруючої й дискримінуючої моделей.

У третьому розділі запропоновано метод шифрування даних з використанням нейронних мереж; докладно розглянуто його кроки; запропоновано модифікацію моделі шифрування, що використовує генеративні конкуруючі мережі; визначено метрики для оцінки запропонованого методу й модифікації моделі.

У четвертому розділі проаналізовано розроблену програмну систему, яка реалізує метод шифрування даних: обґрунтовано вибір технологій, розглянуто загальну архітектуру системи. Розглянуто проведене тестування системи, а також проаналізовано експериментальні результати роботи запропонованого методу й модифікації моделі.

У висновках проаналізовано отримані результати роботи.

У додатках наведено схему роботи генеративної конкуруючої мережі, модель шифрування даних з використанням генеративних конкуруючих нейронних мереж, схему етапу формування нового набору даних запропонованим методом, діаграму класів розробленої системи, експериментальні результати тривалості навчання модифікованої моделі шифрування даних, експериментальні результати точності вирішення

задачі класифікації оригінальних та зашифрованих даних, фрагменти тексту програми та копію презентації.

Робота виконана на 70 аркушах, містить 3 додатки та посилання на список використаних літературних джерел з 34 найменувань. У роботі наведено 17 рисунків, 7 таблиць та 1 лістинг.

Ключові слова: захист приватних наборів даних, конкуруюча нейронна криптографія, шифрування, генеративна конкуруюча мережа, задача класифікації.

ABSTRACT

Theme urgency. Data analysis systems and artificial intelligence tools are widely used in many areas of human activity. Every year the class of tasks, where it is possible and expedient to apply them, is constantly expanding. An important element for building a reliable and accurate system is the availability of sufficient data for training. However, not all collected data can be used for training in decisions using artificial intelligence, as they may contain a significant amount of private information: confidential (eg, financial, military data), confidential (identifying data: passport data, registration number) taxpayer) or sensitive (medical data containing patient diagnoses). Under these circumstances, the development of methods to protect private datasets that allow their use for training is urgency, as the amount of publicly available data is limited.

Object of research is the processes of encryption, decryption and generation of information using neural networks.

Subject of research is the methods, approaches and models of dataset protection using neural networks.

Research objective: the development of an algorithmic-software method of data encryption based on generative adversarial neural networks for the use of encrypted private datasets in data analysis and artificial intelligence systems.

Research methods. Methods of mathematical and software modelling, optimization methods and artificial intelligence methods are used in the research.

Scientific novelty consists in the following:

1. The method of datasets encryption is proposed, which allows the use of private data in artificial intelligence and data analysis systems. In particular, when solving the problem of image classification, the accuracy of the results

obtained on encrypted data with a known encryption key is only 2-4% lower than the accuracy of the classification of the original data.

2. The modification of the data encryption model by using two-dimensional convolutional neural networks is proposed, which allows to speed up 1.5 times image encryption, compared with the classical model, which is designed to encrypt individual bits.

3. The software architecture, the feature of which is the encapsulation of the pre-processing algorithm of the input data set, which allows based on the template "Strategy" to replace a specific implementation in the instance of the classifier class without the use of inheritance, is developed.

Practical value of the obtained results consists in the following: the proposed method of data encryption allows the use of private datasets in public artificial intelligence systems by encrypting them. The modified model of encryption allows to accelerate its training, and the developed software for encryption and classification of the data allows the user to perform the analysis of the data, both on the original data, and on the encrypted.

Approbation. The basic points and outcomes of the research have been presented and discussed at the XII scientific conference for students and postgraduates "Applied mathematics and computing" PMK-2019 (Kyiv, November 13-15, 2019). Based on the results of the work, a scientific article was published in the international scientific journal “Radio Electronics, Computer Science, Control” 2020 (issue №2), which is included in the scientometric base Web of Science.

Structure and content of the thesis. The master thesis consists of the introduction, four chapters, conclusions and appendixes.

The introduction presents the general description of the research, gives the overview on a current state of the scientific problem, explains the research topicality, the purpose of research and the subtasks that are included in the corresponding scientific and practical task are formulated.

In the first chapter the main methods of protection of private data sets that can be used to build a system of data analysis and artificial intelligence are discussed.

In the second chapter the data encryption model that uses generative adversarial networks is considered and analyzed; the structure of convolutional neural networks, which are the basis for constructing generating and discriminating models, is investigated.

In the third chapter the data encryption method using neural networks is proposed; his steps are considered in detail; a modification of the encryption model using generative adversarial networks is proposed; metrics for evaluation of the proposed method and modification of model are defined.

In the fourth chapter the developed software system which implements the data encryption method is analyzed: the choice of technologies is substantiated, the general architecture of the system is considered. The system testing is considered, as well as the experimental results of the proposed method and model modification are analyzed.

In the conclusions the general conclusions on the presented thesis are given; the obtained results are analyzed.

In the appendixes the diagram of the generative adversarial network, the data encryption model using generative adversarial neural networks, the diagram of the stage of a new dataset formation by the proposed method, the class diagram of the developed system, experimental results of the modified data encryption model, experimental results of the accuracy of solving the classification problem of original and encrypted datasets, fragments of the program text and a copy of the presentation.

The thesis is presented in 70 pages, it contains 3 appendixes and 34 references to the used information sources. 17 figures, 7 tables, and 1 listing are given in the thesis.

Key words: protection of private datasets, adversarial neural cryptography, encryption, generative adversarial network, classification problem.

РЕФЕРАТ

Актуальность темы. Системы анализа данных и средств искусственного интеллекта широко применяются во многих сферах человеческой деятельности. С каждым годом класс задач, где можно и целесообразно их применять, постоянно расширяется. Важным элементом для построения надежной и точной системы является наличие достаточного количества данных для обучения. Однако, далеко не все собранные данные могут быть использованы для обучения в решениях с использованием искусственного интеллекта, поскольку они могут содержать значительное количество частной информации: секретной (например, финансовые, военные данные), конфиденциальной (идентифицирующие данные: паспортные данные, регистрационный номер учетной карточки налогоплательщика) или чувствительной (медицинские данные, содержащие диагнозы пациентов). При таких условиях, разработка методов защиты частных наборов данных, которые позволят их использования для обучения, является актуальным, поскольку количество общедоступных данных ограничено.

Объектом исследования являются процессы шифрования, дешифрования и генерации информации с использованием нейронных сетей.

Предметом исследования является методы, способы и модели защиты наборов данных с использованием нейронных сетей.

Цель работы: разработка алгоритмически-программного метода шифрования данных на основе генеративных конкурирующих нейронных сетей для использования зашифрованных частных наборов данных в системах анализа данных и искусственного интеллекта.

Методы исследования. В работе используются методы математического и программного моделирования, методы оптимизации, методы искусственного интеллекта.

Научная новизна работы состоит в следующем:

1. Предложен метод шифрования наборов данных, что позволяет использовать частные данные в системах искусственного интеллекта и анализа данных. В частности, при решении задачи классификации изображений, точность полученных результатов на зашифрованных данных с известным ключом шифрования лишь на 2-4% ниже, чем точность классификации оригинальных данных.

2. Модель шифрования данных была модифицирована, путем использования двумерных свёрточных нейронных сетей, что позволяет ускорить в 1,5 раза шифрование изображений, по сравнению с классической моделью, предназначенной для шифрования отдельных бит.

3. Разработана архитектура программного обеспечения, особенностью которой является инкапсуляция алгоритма предварительной обработки входного набора данных, позволяет на основе шаблона «Стратегия» заменять конкретную реализацию в экземпляре класса классификатор без применения наследования.

Практическая ценность полученных в работе результатов состоит в том, что предложенный метод шифрования данных позволяет использование частных наборов данных в общедоступных системах искусственного интеллекта путем их шифрования. Модифицированная модель шифрования позволяет ускорить ее обучение, а разработанное программное обеспечение для шифрования и классификации данных позволяет пользователю проводить анализ данных, как на оригинальных данных, так и на зашифрованных.

Апробация работы. Основные положения и результаты работы были представлены и обсуждались на XII научной конференции магистрантов и

аспирантов «Прикладная математика и компьютинг» ПМК-2019 (Киев, 13-15 листопада 2019 г.). По результатам работы была опубликована статья в международном научном журнале «Радиоэлектроника, информатика, управление» 2020 (выпуск №2), входящим в наукометрическую базу данных Web of Science.

Структура и объем работы. Магистерская диссертация состоит из введения, четырех глав, выводов и приложений.

Во вступлении представлена общая характеристика работы, сделана оценка текущего состояния проблемы, обоснована актуальность направления исследований, сформулированы цель исследования и задания, которые включает соответствующая научно-практическая задача.

В первом разделе рассмотрены основные методы защиты частных наборов данных, которые могут быть использованы при построении систем анализа данных и искусственного интеллекта.

Во втором разделе рассмотрена и проанализирована модель шифрования данных, которая использует генеративные конкурирующие сети; исследована структура свёрточных нейронных сетей, которые являются основой для построения генерирующей и дискриминирующей моделей.

В третьем разделе предложен метод шифрования данных с использованием нейронных сетей; подробно рассмотрены его шаги; предложено модификацию модели шифрования, использует генеративные конкурирующие сети; определены метрики для оценки предложенного метода и модификации модели.

В четвертом разделе разработанная программная система, которая реализует метод шифрования данных, была проанализирована: обоснован выбор технологий, рассмотрена общая архитектура системы. Было рассмотрено проведенное тестирование системы, а также проанализированы экспериментальные результаты предложенного метода и модификации модели.

В выводах сделаны общие выводы по работе; проанализированы полученные результаты.

В приложениях приведены схема работы генеративной конкурирующей сети, модель шифрования данных с использованием генеративных конкурирующих нейронных сетей, схема этапа формирования нового набора данных предложенным методом, диаграмма классов разработанной системы, экспериментальные результаты продолжительности обучения модифицированной модели шифрования данных, экспериментальные результаты точности решения задачи классификации оригинальных и зашифрованных данных, фрагменты текста программы и копия презентации.

Работа представлена на 70 страницах, содержит 3 приложения и ссылки на список использованных литературных источников из 34 наименований. В работе приведены 17 рисунков, 7 таблиц и 1 листинг.

Ключевые слова: защита частных наборов данных, конкурирующая нейронная криптография, шифрование, генеративная конкурирующая сеть, задача классификации.

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

Приватні дані – дані, що містять секретну, конфіденційну або чутливу інформацію.

AI (Artificial intelligence) – штучний інтелект.

GDPR – General Data Protection Regulation.

GAN (Конкуруючі нейронні мережі) – алгоритм машинного навчання без вчителя, побудований на комбінації двох нейронних мереж, одна з яких генерує зразки (генеративна), а інша намагається відрізнити правильні (оригінальні) зразки від неправильних.

Гомоморфне шифрування – форма шифрування, яка дозволяє проводити обчислення на шифротексті, дешифрований результат яких якого буде таким самим, як і результат виконання операцій над відкритим текстом.

Федеративне навчання – архітектурний підхід до навчання алгоритму штучного інтелекту на багатьох кінцевих пристроях чи серверах, що містять локальні набори даних, які в ході навчання залишаються на пристрої (не відбувається їх обмін між пристроями).

P (plaint text) – вхідний текст для шифрування.

K (key) – ключ шифрування.

C (cypher text) – зашифрований текст.

Функція втрат (L, loss function) – міра того, наскільки прогноз моделі далекий від її мітки (очікуваного результату).

Симетричне шифрування – це тип шифрування, в якому шифрування та дешифрування інформації здійснюється з використанням лише одного ключа (приватного).

Згорткова нейронна мережа (CNN – convolutional neural network) – клас глибинних нейронних мереж, які характеризуються прямим поширенням (інформація передається в одному напрямку) і наявністю щонайменше одного згорткового шару. Типова згорткова нейронна мережа

складається з певної комбінації наступних шарів: згорткових, агрегувальних, повноз'єднаних та шарів нормалізації.

Adversarial Neural Cryptography – конкуруюча нейронна криптографія; термін, що позначає галузь криптографії, що використовує конкуруючі нейронні мережі для шифрування даних.

Гіперпараметр моделі – це зовнішній параметр для моделі, який не можна оцінити за даними, його значення встановлюються перед початком навчання. Прикладами таких параметрів є коефіцієнт швидкості навчання (learning rate) та розмір міні партії даних, що обробляється моделлю (mini-batch size).

Епоха навчання – це період навчання, за який весь навчальний набір даних проходить через нейронну мережу лише 1 раз. Таким чином, епоха складається з ітерацій $I = \frac{N}{batch\_size}$, де N – кількість екземплярів даних навчального набору, *batch_size* – кількість екземплярів даних, що опрацьовується протягом однієї ітерації.

Навчальний набір даних – підмножина набору даних, що використовується для тренування моделі.

Тестовий набір даних – підмножина набору даних, яка використовується для тестування моделі після того, як модель пройшла первинну перевірку набором для затвердження (validation set).

Набір даних для затвердження (validation set) – підмножина набору даних, яка використовується для об’єктивної оцінки моделі на етапі навчання, що дозволяє регулювати гіперпараметри моделі під час навчання. Варто зауважити, що ці дані не використовуються для навчання моделі.

Метрика – критерій, за допомогою якого можна оцінити значення числових властивостей програмного забезпечення.

Точність позитивних передбачень або прецизійність (precision) – метрика, яка використовується для моделей класифікації. Вона визначає

частоту, з якою модель давала правильні результати при прогнозуванні позитивного класу.

ООП – об’єктно-орієнтоване програмування.

Шаблони проектування – повторювані архітектурні конструкції, які дозволяють ефективно вирішувати задачі, які часто зустрічаються в розробці архітектури ПЗ.

ВСТУП

Системи аналізу даних і засобів штучного інтелекту широко застосовуються у багатьох сферах людської діяльності. З кожним роком клас задач, де можна й доцільно їх застосовувати, постійно розширюється. Наприклад, такі рішення застосовуються у електронній комерції (рекомендації товарів, онлайн помічники), в управлінні людськими ресурсами (аналіз резюме кандидатів), у соціальній сфері (боротьба зі спамом, видалення образливого контенту), в медицині, гральній індустрії, і навіть у політиці. Це далеко не вичерпний перелік сфер застосування систем штучного інтелекту.

Важливим елементом для побудови надійної і точної системи є наявність достатньої кількості даних для навчання. Однак, далеко не всі зібрані дані можуть бути використані для навчання у рішеннях з використанням штучного інтелекту. Мова йде, перш за все, про дані, які можуть містити значну кількість приватної інформації: секретної (наприклад, фінансові, військові дані), конфіденційної (ідентифікуючі дані: паспортні дані, реєстраційний номер облікової картки платника податків) або чутливої (медичні дані, які містять діагнози пацієнтів). За таких умов, пошук набору даних для побудови системи штучного інтелекту є важливим завданням.

Кількість інформації, що може бути використана для аналізу, постійно зростає, а її наявність часто є конкурентною перевагою незначної кількості компаній у всьому світі, але навіть ця інформація потребує використання певних механізмів захисту інформації навіть у приватних системах штучного інтелекту.

Отже, розроблення методів захисту приватних наборів даних, що дозволять їх використання для навчання є актуальною, оскільки кількість загальнодоступних даних є обмеженою.

У даній дисертації розглядається алгоритмічно-програмний метод шифрування даних з використанням нейронних мереж для шифрування приватних наборів даних, що дозволить здійснювати навчання на зашифрованих даних.

Об’єктом дослідження є процеси шифрування, дешифрування та генерації інформації з використанням нейронних мереж.

Предметом дослідження є методи, способи та моделі захисту наборів даних з використанням нейронних мереж.

Метою магістерської дисертації є розроблення алгоритмічно-програмного методу шифрування даних з використанням генеративних конкуруючих нейронних мереж для використання зашифрованих приватних наборів даних в системах аналізу даних і штучного інтелекту.

Науково-практична задача, що розв’язується в даній дипломній роботі, включає наступні завдання:

1. Дослідження методів захисту приватних наборів даних.
2. Аналіз методів, способів, підходів та моделей шифрування та генерації наборів даних.
3. Розроблення та дослідження моделі шифрування даних з використанням нейронних мереж.
4. Розроблення алгоритмічно-програмного методу шифрування приватних наборів даних.
5. Вирішення задачі класифікації оригінальних та зашифрованих даних.
6. Аналіз отриманих результатів.

1. АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ЗАХИСТУ ПРИВАТНИХ НАБОРІВ ДАНИХ

1.1. Основні способи захисту приватної інформації в системах штучного інтелекту

Безпека і конфіденційність даних є однією з найбільших проблем сучасності, адже значна частина інформації зберігається в електронному вигляді й передається через різноманітні пристрої (смартфони, комп'ютери), які широко увійшли в суспільне життя. Підтвердженням цього є й посилення законодавства спрямованого на забезпечення захисту даних. Зокрема, у 2016 році у Європейському союзі було прийнято General Data Protection Regulation (GDPR) – регламент щодо захисту персональних даних, а у 2018 році в штаті Каліфорнія – California Consumer Privacy Act (CCPA). Вищезгадані правові акти спонукають компанії, розробників та дослідників розробляти інформаційні системи конфіденційні за дизайном (що реалізовують підхід «Privacy by Design»).

Конфіденційність даних є центральним питанням і в системах штучного інтелекту, зокрема в навчанні та тестування моделей AI, особливо тих, які використовують приватні дані. Для її забезпечення в цій сфері виділяють 4 ключові аспекти [1]:

- конфіденційність навчальних даних (зловмисник не повинен мати можливості відтворити навчальні дані, зокрема використовуючи методи зворотної розробки (reverse engineering));
- конфіденційність вхідних даних (дані, що вводить користувач не повинні бути доступні іншим сторонам, в тому числі власнику моделі);
- конфіденційність вихідних даних (вихідні дані моделі повинні бути доступні лише користувачу, який очікує передбачення);

- конфіденційність моделі (модель не може буди викраденою зловмисником).

Конфіденційність моделі стоїть осторонь, від решти аспектів, адже її метою є захист вже навченої моделі, а не безпека наборів даних і її повинен забезпечити власник моделі; у той час як інші аспекти, забезпечуються на етапі побудови, навчання й тестування моделі, за що значною мірою відповідає фахівець з розробки систем штучного інтелекту.

Основними способами забезпечення захисту приватних наборів даних є [2]:

- генерація синтетичних наборів даних (synthetic data generation);
- обробка приватних наборів даних (анонімізація даних, диференційна приватність, гомоморфне шифрування);
- федеративне навчання.

Розглянемо їх докладніше.

1.2. Аналіз підходів до генерації синтетичних наборів даних

Синтетичні дані – штучні дані, які згенеровані за певним алгоритмом на відміну від оригінальних даних, які базуються на реальній інформації. Ідея таких даних полягає в тому, що вони складаються з нових точок даних, що не є простою модифікацією існуючого набору даних. Однак, варто зауважити, що не всі синтетичні дані є анонімними.

Синтетичні дані все частіше застосовуються в системах машинного навчання. З їх допомогою навчають модель з наміром перенести результати навчання на реальні дані.

За умови успішної побудови генератора синтетичних даних, можна отримати такі переваги:

- швидко і дешево можна згенерувати стільки даних, скільки необхідно;

- згенеровані дані можуть мати ідеально точні мітки, включаючи мітки, які може бути складно отримати на реальних даних;
- синтетичні дані можна використати як заміну певних реальних сегментів даних, які містять, приватну інформацію.

Генератор синтетичних даних можна побудувати використовуючи різні підходи (зокрема, доменну рандомізацію чи комбінування реальних даних), шляхом побудови генеративних моделей серед яких [3]:

- прихована модель Маркова;
- Баєсівська мережа;
- агентне моделювання;
- варіаційний автокодувальник;
- генеративні конкуруючі нейронні мережі.

Розглянемо одну з перелічених генеративних моделей: генеративні конкуруючі мережі (GANs) – це архітектура глибокого навчання нейронних мереж, яка була запропонована Яном Гудфелоу та іншими дослідниками університету в Монреалі у 2014 році [4].

Основна ідея цієї архітектури пов’язана з конкуренцією між генеративними й дискримінаційними алгоритмами.

Дискримінаційні алгоритми намагаються класифікувати вхідні дані; тобто, вони передбачають мітку або категорію, до якої належать ці дані, враховуючи особливості екземпляра даних. Прикладом цього, є алгоритм розпізнавання спаму в електронній пошті, який враховуючи всі слова повідомлення , визначає чи містить воно мітку «спам». Математично це можна сформулювати як пошук умовної ймовірності $p(y|x)$, тобто ймовірність мітки y для заданих вхідних даних x.

Генеративні моделі вирішують складніше завдання, ніж аналогічні дискримінаційні. Один із способів їх описання є те, що вони працюють навпаки: замість того, щоб передбачити мітку з певними параметрами, вони намагаються передбачити параметри, що надаються певній мітці. В

контексті розглянутої задачі з розпізнаванням спаму, якщо припустити, що лист є спамом лист, то метою генеративного алгоритму є обчислення ймовірності того що такий лист існує. Генеративні моделі дозволяють фіксувати $p(x|y)$, ймовірність вхідних даних (їх ознак) x, для заданої мітки y. Зазначаючи це, генеративні алгоритми можуть також використовуватися як класифікатори, проте, часом, вони можуть зробити більше, ніж просто класифікувати вхідні дані.

Генеративна-конкуруюча мережа складається з двох частин:

- генератора, який навчається генерувати правдоподібні дані;
- дискримінатора, який навчається відрізняти підроблені (хибні) дані від реальних.

Схема навчання такої мережі представлена на рис. 1.1.

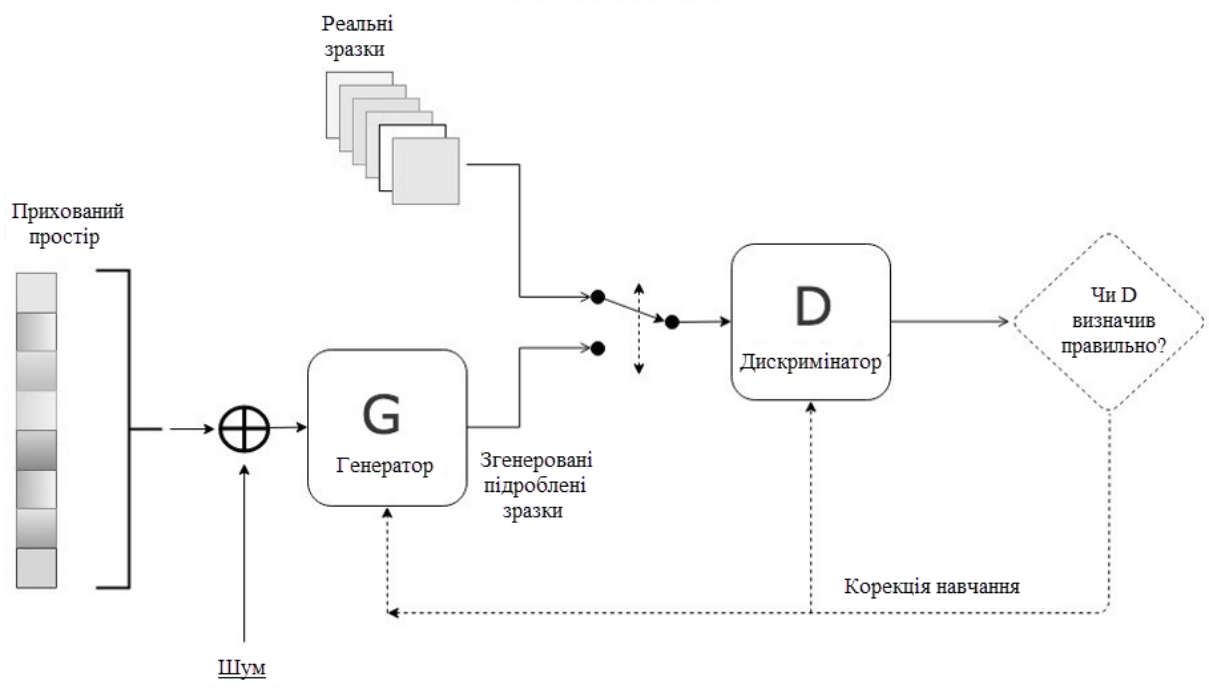


Рис. 1.1. Схема навчання генеративної конкуруючої мережі

Процес навчання нейронних мереж за класичною GAN архітектурою складається з наступних кроків:

1. Обираємо шум з випадкового розподілу й подаємо його на вхід мережі-генератора *G*, щоб створити екземпляр-підробку *x* (мітка *y* = 0) –> (*x*, *y*).
2. Подаємо отриману підроблену пару й справжню пару *x* (мітка *y* = 1) й альтернативно подаємо її до дискримінатора *D*.
3. Обчислюємо функцію втрат дискримінатор *D*, який є нейронною мережею бінарної класифікації, як для підроблених *x*, так і для реальних *x* і поєднуємо їх як кінцеву втрату.
4. Обчислюємо функції втрат генератор *G* від його шуму як втрати *G*, оскільки кожна мережа має різні цільові функції.
5. Коригуємо параметри нейронних мереж за допомогою функцій втрат.
6. Застосувати алгоритм оптимізації. Наприклад, алгоритм градієнтного спуску, ADAM, RMSprop тощо.
7. Повторити кроки 1-6 протягом певної кількості епох.

Розглянемо функції втрат для нейронних мереж побудованих за GAN архітектурою. Оскільки GAN мережі намагаються повторити розподіл ймовірностей, вони повинні використовувати функції втрат, які відображають відстань між розподілом даних, згенерованих за допомогою GAN, та розподілом реальних даних. Питання визначення відмінності між двома розподілами у функціях втрат для GAN є сферою активних досліджень, і було запропоновано багато підходів.

Одним з перших підходів до визначення функції втрат є мінімакс втрати, що був запропонований при формуванні GAN архітектури. Генератор намагається мінімізувати функцію подану за наступною формулою, тоді як дискримінатор намагається її максимізувати [4, 5]:

$$\min_G \max_D L(D,G) = E_x\left[\log\left(D(x)\right)\right] + E_z\left[\log\left(1 - D\left(G(z)\right)\right)\right],$$

де $D(x)$ – оцінка дискримінатора щодо ймовірності того, що реальний екземпляр даних x є реальним;

E_x – очікуване значення для всіх реальних екземплярів даних;

$G(z)$ – вихід генератора при заданому шумі z;

$D(G(z))$ – оцінка дискримінатора щодо ймовірності того, що фальшивий екземпляр є справжнім;

E_z – очікуване значення над усіма випадковими входами в генератор (фактично, очікуване значення над усіма створеними фальшивими екземплярами $G(z)$).

Ця формула випливає з перехресної ентропії між реальними та згенерованими розподілами. Генератор не може безпосередньо впливати на $\log(D(x))$ у функції, тому для генератора мінімізація втрат еквівалентна мінімізації $\log(1 - D(G(z)))$. Варто зауважити, що в оригінальному документі, що вводить термін GAN [4], зазначається, що вищезазначена функція втрат при мінімакс підході може призвести до того, що генеративна конкуруюча мережа застрягне на ранніх етапах навчання, коли робота дискримінатора дуже проста. У цьому документі пропонується змінити втрати генератора так, щоб генератор намагався максимізувати функцію $\log(D(G(z)))$.

Потенціал генеративних конкуруючих нейронних мереж є величезним, оскільки вони можуть навчитися імітувати будь-який розподіл даних. Тобто, GAN можна навчити створювати світи, подібні до нашого, у будь-якій галузі: образи, музика, промова, проза. Приклад таких згенерованих даних, зображено на рис. 1.2.

Рис. 1.2. Зображення, згенеровані конкуруючою нейронною мережею у 2017 році

Однак, конкуруючі генеративні мережі містять і ряд недоліків, серед яких складність навчання (незбіжність моделі, дисбаланс між генератором і дискримінатором, висока чутливість до гіперпараметрів) та реалістичність, а не реальність даних (генератор створює лише обмежений набір зразків, ця проблема особливо характерна для невеликих наборів тренувальних даних).

1.3. Аналіз методів попередньої обробки приватних наборів даних

Іншим способом захисту приватних наборів даних є попередня обробка набору даних, яка може здійснюється наступними методами:

- анонімізація даних;

- диференційна приватність (differential privacy);
- гомоморфне шифрування;

Розглянемо ці методи докладніше.

1.3.1. Анонімізація даних

Анонімізація даних – це процес захисту приватної або конфіденційної інформації шляхом видалення або зміни ідентифікаторів, які з'єднують особу із збереженими даними. Іноді анонімізацію даних також називають «обфускація», «маскування» або «деідентифікація» даних.

Анонімізація даних здійснюється багатьма галузями, які працюють з конфіденційною інформацією, зокрема, такими як охорона здоров'я, фінансова та цифрова індустрії. Це сприяє цілісності обміну даними. Наприклад, коли лікарні потрібно передати дані до медичної дослідницької лабораторії, вона захищає конфіденційні дані й зберігає анонімність своїх пацієнтів. Це можна зробити шляхом видалення імен, номерів соціального страхування, дати народження та адрес своїх пацієнтів з загальнодоступного списку, залишаючи лише важливі компоненти, які необхідні для медичних досліджень, такі як вік, хвороби, зріст, вага, стать, раса тощо.

Анонімність даних може забезпечуватись різними способами. Серед них базовими є [6, 7]:

- придушення атрибутів і/або записів;
- перестановка даних;
- підміна даних (частковим випадком якої є псевдонімізація);
- маскування символів;
- дисперсія чисел і дат;
- узагальнення.

Розглянемо базові способи анонімізації на прикладі структурованих даних. Нехай задано набір даних, що представлений у табл. 1.1.

Таблиця 1.1

Оригінальний набір вхідних даних

Користувач	Вік	E-mail	Поштовий код	Адреса	Кількість покупок
Олег	21	oleh@gmai.com	02222	вулиця Вишнева 1	3
Руслан	34	ruslan@i.ua	02217	вулиця Крайня 12б	8
Валентин	18	valentyn@outlook.com	01103	вулиця Вишнева 10	2
Степан	27	stepan@gmai.com	03061	вулиця Франка 7	5

Придушення атрибутів і/або записів полягає у їх видаленні з набору даних, якщо вони містять ідентифікуючу інформацію, але є несуттєвими для подальшого аналізу або не можуть бути анонімізовані іншими способами.

Перестановка даних полягає у перестановці окремих значень атрибутів, в результаті чого, вони залишаться присутніми в модифікованому наборі даних, але не відповідатимуть оригінальним засобам.

Підміна даних – це процес заміни даних зі стовпця-атрибуту випадковими значеннями зі списку фальшивих, але схожих на вигляд даних. Таким чином, номери кредитних карток можуть бути замінені випадковим рядком з 16 чисел. Частковим випадком підміни даних є

псевдонімізація – заміна ідентифікаційних даних кодованими даними (складеними за певним принципом). При цьому псевдоніми можуть бути як оборотними (якщо можна отримати оригінальні дані), так і необоротними.

Маскування символів – це заміна символів у значенні атрибута даних, наприклад, використовуючи сталий символ, такий як «*» або «x». Маскування символів зазвичай виконують частково, тобто застосовується лише до деяких символів в атрибуті.

Дисперсія дат і чисел полягає у зміні значень атрибута на певний значення з інтервалу. Дати і числа часто використовуються в якості параметрів пошукового запиту до набору даних, а також вони є важливими елементами медичної та фінансової статистики. Оскільки додавання/віднімання певного заданого однакового значення до атрибуту легко можна декодувати, то необхідно додавати/віднімати випадкове значення з певного довірчого інтервалу, що дозволить зберегти розподіл даних за цим атрибутом.

Узагальнення – заміна специфічних даних більш загальною, але ще корисною інформацією. За рахунок цього відбувається зниження точності даних. Цей спосіб зазвичай полягає у використанні діапазонів (перетворення віку людини у віковий діапазон) або менш точної інформації (перетворення точного місця розташування у менш точне). Цей прийом іноді також називають перекодуванням даних. Його доцільно використовувати для значень, що можуть бути узагальнені й ще будуть корисні в перетвореному вигляді.

Таким чином, застосувавши розглянуті вище способи придушення атрибутів (атрибут Адреса), перестановки даних (атрибут Користувач, маскування символів (атрибути Поштовий код та E-mail) і дисперсії чисел (атрибут Вік), можна отримати анонімізований набір, що представлений у табл. 1.2.

Таблиця 1.2

Набір даних, до якого застосовано способи анонімізації (придушення атрибутів, перестановка даних, маскування символів і дисперсія чисел)

Користувач	Вік	E-mail	Поштовий код	Кількість покупок
Степан	23	****@gmai.com	02xxx	3
Валентин	32	******@i.ua	02xxx	8
Олег	15	********@outlook.com	01xxx	2
Руслан	26	******@gmai.com	03xxx	5

Якщо ж застосувати способи розглянуті вище способи підміни даних (атрибут Користувач), узагальнення даних (атрибути Вік та Адреса), псевдонімізацію (атрибут Користувач) та маскування символів (атрибут E-mail), можна отримати анонімізований набір, що представлений у табл. 1.3.

Таблиця 1.3

Набір даних, до якого застосовано способи анонімізації (підміна даних, узагальнення даних, псевдонімізація, маскування символів)

Користувач	Вік	E-mail	Поштовий код	Адреса	Кількість покупок
User3	20-30	****@gmai.com	02222	вулиця Вишнева	3
User1	> 30	******@i.ua	02217	вулиця Крайня	8
User4	< 20	********@outlook.com	01103	вулиця Вишнева	2
User2	20-30	******@gmai.com	03061	вулиця Франка	5

На основі розглянутих прикладів, можна зробити висновок, що головна перевага анонімізації полягає в тому, що вона дозволяє приховати чутливі аспекти даних. Однак, навіть за умови очищення набору даних від ідентифікуючої інформації, анонімізовані дані можуть бути розшифровані і розкриті за допомогою методів деанонімізації (також називають повторна ідентифікація). Оскільки дані зазвичай зберігаються в декількох джерел (частина з яких доступна для широкого загалу, наприклад, через державні реєстри даних), методи деанонімізації можуть перехресно посилатися на джерела та виявляти особисту інформацію в анонімізованому наборі даних. У зв'язку з цим, критики вважають, що анонімізація надає помилкове відчуття безпеки.

1.3.2. Диференційна приватність

Диференційна приватність – метод захисту даних, який захищає конфіденційність користувача шляхом додавання випадкового шуму до даних. Мета цього методу – забезпечення жорстких статистичних гарантій того, що зловмисник не зможе зробити висновок про приватні дані, на основі результатів даних, що отримані за допомогою рандомізованого алгоритму. Іншими словами, абсолютна різниця ймовірності p того, що результатом запиту до оригінального набору даних D_1 є значення s, і ймовірності p_2 того, що результатом запиту до модифікованого набору даних D_2 є те саме значення, повинна бути менша певного значення ε (тобто така різниця ймовірностей має бути в межа похибки).

Математично метод диференційної приватності можна сформулювати наступним чином [8]:

Нехай $\varepsilon > 0$, A – рандомізований алгоритм, який приймає на вхід приватний набір даних D_1, а $E(A)$ – область значень цього алгоритму. Алгоритм A є ε-диференційно приватним, якщо для всіх записів наборів

даних D_1 і D_2, які відрізняються єдиним записом, для всіх підмножин $S \subseteq E(A)$ виконується наступна нерівність:

$$p\{A(D_1) \in S\} \leq e^{\varepsilon} \cdot p\{A(D_2) \in S\},$$

де p – ймовірність отримана з випадковості рандомізованого алгоритму.

Зазначеного результату можна досягти шляхом додавання випадкового шуму, зокрема шуму від розподілу Лапласа або Гауса.

Варто зауважити, що чим більше шуму додається до вхідних даних, тим менше цінності вони представляють, наслідком чого є. менша точність їх аналізу. Цю тезу чудово ілюструє приклад, зображений на рис. 1.3.

Рис. 1.3. Залежність цінності даних, від кількості доданого шуму

Застосувавши до зображення значну кількість шуму, можна досягти високого рівня захисту конфіденційності даних, при цьому екземпляр даних майже повністю втрачає свої властивості, що унеможливить здійснення подальшого аналізу цих даних. Зважаючи на це, під час застосування методу диференційної приватності необхідно знайти баланс між захистом приватності й збереженням характерних особливостей даних.

Іншим недоліком методу є необхідність достатньо великої множини даних (більшої, ніж для анонімізації) для його ефективного застосування.

Зараз цей метод використовують такі компанії, як Apple, Google, Facebook та Uber. Також його планують застосовувати США при здійсненні перепису населення у 2020 році.

1.3.3. Гомоморфне шифрування

Гомоморфне шифрування – форма шифрування, яка дозволяє проводити обчислення на шифротексті, дешифрований результат яких буде таким самим, як і результат виконання операцій над відкритим текстом [9].

Базовим поняттям цього методу захисту приватних даних є гомоморфізм. Математично це можна сформулювати наступним чином:

Функція f: $G \to H$ між двома групами G та H є гомоморфною, якщо:

$$f(xy) = f(x)\,f(y) \text{ для } \forall x, y \in G.$$

Найпростішими прикладами таких функцій є $c(x+y) = cx + cy$, $|xy| = |x||y|$, $(xy)^c = x^c y^c$.

Сформулювати гомоморфне шифрування можна наступним чином:

Нехай $a \cdot b \equiv 1$, тоді

$$E(x) = x^b \bmod n,$$

$$D(x) = y^a \bmod n,$$

$$E(x_1) \cdot E(x_2) = {x_1}^b \cdot {x_2}^b \bmod n = (x_1 \cdot x_2)^b \bmod n = E(x_1 \cdot x_2),$$

де x_1, x_2 – відкритий текст.

Комбінуючи дві операції (наприклад, складання і множення), можна побудувати будь-яку довільну функцію. Сучасні схеми шифрування, що підтримують гомоморфні обчислення в необмеженій кількості, називаються повністю гомоморфними схемами шифрування. Перша згадка про гомоморфне шифрування була ще в 1978 році, але повністю гомоморфна криптосистема була побудована Крейгом Джентрі лише в 2009 році. Саме з цього часу кількість досліджень у цій сфері значно збільшилась.

У 2015 році компанія Microsoft розробила бібліотеку SEAL (Simple Encrypted Arithmetic Library) з відкритим кодом, написану мовою C++, яка

реалізовує різні форми гомоморфного шифрування, зокрема схеми BFV, CKKS [2], яку намагаються застосовувати в системах штучного інтелекту. Однак, обчислювальна потужність і пам'ять все ще залишаються значними технічними перешкодами на шляху до цього, оскільки обробка зашифрованих даних вимагає значно більших ресурсів – насправді, в мільйон разів більше, ніж потрібно для обробки незашифрованих даних або відкритого тексту. Завдяки досягненням технологій за останні 10 років, гомоморфне шифрування тепер не тільки можливе, а й все більше використовується на практиці.

Отже, певні операції, можуть бути виконані безпосередньо на шифрованих текстах, так що при розшифруванні отримується та ж відповідь, що й при виконанні операцій на вихідних повідомленнях. Це і є основною перевагою цього методу. Зашифровані дані впливають на AI-рішення так само, як і оригінальні. Головним недоліком є обсяг зашифрованої інформації порівняно з оригінальними даним. Так, для 1 Мб інформації обсяг зашифрованих даних може сягати 10 Гб.

Одним з підходів до захисту інформації є протокол конфіденційного обчислення (secure multi-party computation, MPC), який забезпечує збереження конфіденційності розподілених даних. Цей підхід є гомоморфним [10]. Нехай множина користувачів (учасників) $P_1, \ldots, P_n$, кожен з яких має певні приватні дані x_i і прагне обчислити спільну функцію $y = f(x_1,...,x_n)$, що є результатом цих даних. При цьому комунікація повинна бути безпечною між суб'єктами, а дані – коректними. Залежно від вхідних даних і кількості учасників конфіденційний протокол можна побудувати різними способами.

Розглянемо приклад такого протоколу. Нехай треба знайти суму значень даних учасників $\sum_{i=1}^{n} x_i$, кожне зі значень яких не перевищує модуль m ($x_i < m$). Тоді вводиться випадкове число $r < m$, про яке відомо

лише першому учаснику. Кожен учасник по черзі обчислює значення $\tilde{x}_i = x_i + \tilde{x}_{i-1}$, де $\tilde{x}_0 = r$, й передає його наступному. Коли n-й учасник завершить обчислення, то можна знайти загальну суму $y = \sum_{i=1}^{n} x_i = \tilde{x}_n - r$, при цьому приватна інформація суб'єктів шифрування не буде розкритою.

Такий підхід має ряд переваг: він готовий до комерційного використання, усуває компроміс між безпекою і корисністю даних (які характерні для анонімізації й диференційної приватності), а також результат є точним, а сторони не бачать приватних даних інших учасників. Однак, він є обчислювально (необхідно генерувати випадкові числа, виконувати операції над даними кожному учаснику) й комунікаційно (передача інформації між учасниками) витратним.

1.4. Підхід федеративного навчання

Стандартні підходи до створення систем штучного інтелекту вимагають централізації навчальних даних на одній машині. Однак у 2016 році дослідниками з організації Google було запропоновано децентралізований архітектурний підхід – федеративне навчання (Federated Learning) [11, 12].

Ідеєю федеративного навчання є навчання алгоритму штучного інтелекту на багатьох кінцевих пристроях чи серверах, що містять локальні набори даних, які в ході навчання залишаються на пристрої (не відбувається їх обмін між пристроями). Цей підхід відрізняється від традиційних технік централізованого машинного навчання, де всі зразки даних завантажуються на один сервер, а також більш класичних децентралізованих підходів, які передбачають, що локальні вибірки даних розподіляються однаково між пристроями.

Розглянемо приклад такого архітектурного підходу до навчання, який зображено на рис. 1.4 [13].

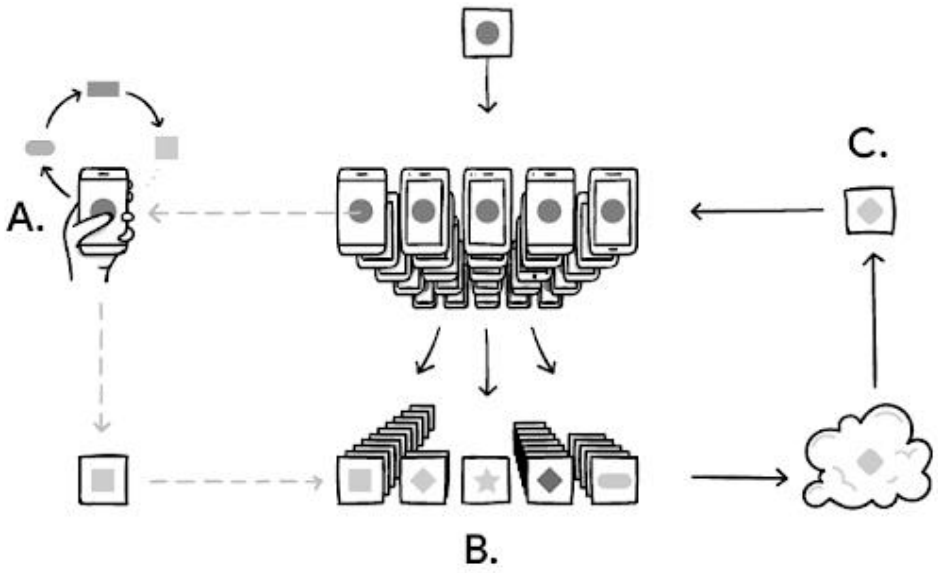


Рис. 1.4. Архітектурний підхід федеративного навчання (Federated Learning)

Нехай необхідно вирішити задачу класифікації різних геометричних фігур (коло, квадрат, ромб, овал тощо). За умови застосування федеративного навчання, поточна версія моделі зберігається на сервері, наприклад, у хмарі. Пристрій користувача завантажує почну версію моделі на свій пристрій (наприклад, телефон або планшет), й удосконалює її шляхом навчання локальному наборі даних (блок А з рис. 1.3), а потім підсумовує зміни ваг моделі й надсилає їх у хмару як невелике оновлення. Як тільки це оновлення, надіслане в зашифрованому вигляді, приходить до сервера, воно відразу усереднюється з оновленнями інших користувачів (блок В з рис. 1.3) й ваги спільної моделі покращуються (блок С з рис. 1.3). Далі ця процедура повторюється. При цьому усі дані, які були використані для навчання залишаються на пристрої користувача, а жодні оновлення в хмарі не зберігаються.

Розглянемо детальніше алгоритм усереднення оновлень на основі задачі обчислення кінцевих сум, який покриває не лише лінійну та логістичну регресію, а й більш складні алгоритми, зокрема нейронні мережі:

$$\min_{w \in R^d} f(w) \text{ де } f(w) \stackrel{def}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(w).$$

Для задачі машинного навчання $f_i(w) = l(w; x_i; y_i)$ – це втрата точності передбачення для прикладу $\{x_i, y_i\}$, зробленого моделлю з параметрами *w*. Припустимо, що *K* існує користувачів, які беруть участь в навчанні моделі, використовуючи локальні набори даних. Нехай $\{P_k\}_{k=1}^{K}$ позначає розділ індексів точок даних $\{1,\ldots,n\}$, оскільки P_k – набір точок даних, що зберігаються у користувача *k*, і визначають $n_k = |P_k|$. Тоді сформулюємо визначення функції втрат для спільної моделі як лінійну комбінацію функцій втрат [11, 12]:

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \stackrel{def}{=} \sum_{k=1}^{K} \frac{n_k}{n} \cdot \frac{1}{n_k} \sum_{i \in P_k} f_i(w).$$

Федеративне навчання дає можливість створювати кращі моделі, з меншою затримкою та з меншим енергоспоживання, гарантуючи конфіденційність. Також цей підхід, окрім надання оновлення до спільної моделі, дозволяє використовувати вдосконалену модель і на кінцевому пристрої з мінімальною затримкою.

Ідеальними задачами для федеративного навчання є задачі, для яких характерно наступне:

- навчання на реальних даних у реальному часі з мобільних пристроїв має явну перевагу над навчанням на довірених даних, які зазвичай доступні в центрі обробки даних;

- дані є чутливими до конфіденційності або мають великі розміри (порівняно з розмірами моделі), тому бажано не зберігати їх в центрі обробки даних, виключно з метою навчання моделей;
- для контрольованих завдань, мітки на даних для яких можна отримати із взаємодії користувача.

Федеративне навчання не може вирішити всі проблеми машинного навчання (наприклад, навчитися розпізнавати різні породи собак шляхом навчання на ретельно маркованих прикладах), а для багатьох інших моделей необхідні дані про навчання вже й так зберігаються в хмарних сховищах (наприклад, фільтр спаму для Gmail).

Зараз федеративне навчання застосовується для інтелектуальної клавіатури Google Gboard на мобільних телефонах з операційною системою Android [13]: коли клавіатура пропонує запит, телефон користувача локально зберігає інформацію про поточний контекст та про те, чи обрав користувач пропозицію. Потім процеси федеративного навчання, які використовують історію на пристрої, пропонують покращити наступну ітерацію моделі пропозицій користувацьких запитів.

Іншим прикладом застосування цього підходу є сфера охорони здоров'я [14]: французький стартап Owkin, де були розроблені моделі навчання біомедичних засобів, заснованих на алгоритмах збору неоднорідних даних (фармацевтичних компаній і медичних установ) шляхом застосування федеративного навчання з використанням технологій високої відстежуваності (технологія distributed ledgers, однією з форм якої є blockchain).

1.5. Порівняння методів захисту приватних наборів даних

Проведемо порівняльний аналіз методів захисту приватних наборів даних, використовуючи наступні п'ять критеріїв:

- складність;

- практичність;
- потреба у великій кількості даних для використання методу;
- надійність (чи значний рівень приховування даних);
- точність системи штучного інтелекту (на модифікованих даних);

Результати порівняння наведені в табл. 1.4.

Таблиця 1.4

Порівняння методів захисту приватних наборів даних

Метод	Складність	Практич-ність	Потреба у великій кількості даних	Надійність	Висока точність системи
Генерування синтетичних даних	+	+	+	+	+/–
Анонімізація даних	–	+	–	–	+
Диференційна приватність	+/–	+	+	+/–	+/–
Гомоморфне шифрування	+	–	–	+	+
Федеративне навчання	–	+/–	+	+	+

З проведеного аналізу, можна зробити висновок, що генерування синтетичних наборів даних є практичним і надійним методом, але досить складним і потребує великої кількості вхідних даних (умов, прикладів) для формування більш менш точної системи штучного інтелекту.

Анонімізація даних є достатньо простою, практичною, а також не потребує великих наборів даних, однак цей метод недостатньо надійний, що є головним критерієм для побудови моделей.

Диференційна приватність – це практичний метод, який потребує великих наборів даних, і залежно від кількості застосованого шуму може бути як дуже надійним і неточним, так і ненадійним, але дуже точним.

Гомоморфне шифрування є надійним і з його допомогою можна побудувати високоточні системи, але він є складним, зокрема обчислювально-витратним, і може бути застосованим до обмеженого класу задач.

Федеративне навчання є надійним та точним методом, за рахунок не поширення локальних навчальних даних, проте його можна застосувати лише за наявності щонайменше кількох незалежних користувачів, які мають достатню кількість навчальних даних.

1.6. Висновки

Проаналізовано переваги та недоліки існуючих методів захисту приватних наборів даних. Зокрема, були розглянуті методи генерування синтетичних даних, анонімізацію даних, диференційну приватність, гомоморфне шифрування та федеративне навчання. Розглянуті методи дозволяють вирішити деякі проблеми конфіденційності даних, але більшість з них знаходяться ще в стані досліджень і вони мають недоліки.

Отже, спираючись на проведений аналіз можна зробити висновок, що розроблення альтернативних методів захисту інформації, які дозволять мінімізувати розглянуті недоліки, є актуальною задачею.

2. МОДЕЛЬ ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ НЕЙРОННИХ МЕРЕЖ

2.1. Математичне підґрунтя для побудови моделі

Традиційно вважається, що методи штучного інтелекту, зокрема нейронні мережі, не варто застосовувати для формування методів захисту інформації в криптографії, адже таким системам складно виконувати навіть базові операції, такі як XOR, які використовуються в більшості криптографічних алгоритмів. Доцільним, вважається, лише використання таких методів для зламу алгоритмів шифрування. Проте з розширенням сфери використання GAN (generative adversarial networks – генеративні конкуруючі нейронні мережі), ситуація змінилась.

Модель шифрування з використанням нейронних мереж була запропонована представниками компанії Google Brain у 2016 році [15, 16]. Її зображено на рис. 2.1.

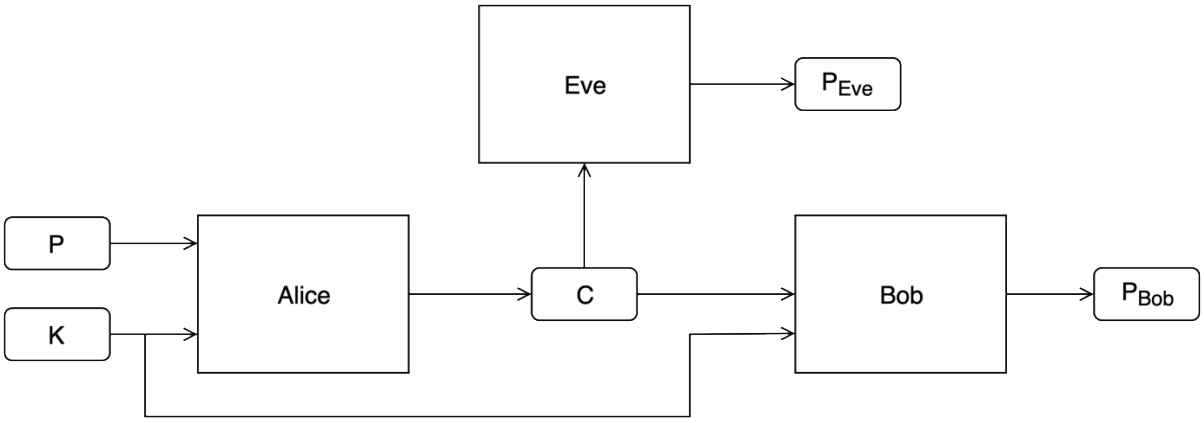


Рис. 2.1. Модель шифрування даних з використанням GAN

Ідея моделі полягає в поєднанні двох підходів:

1. Принципи симетричного шифрування.

2. Концепція генеративних конкуруючих нейронних мереж.

Розглянемо їх докладніше.

2.1.1. Симетричне шифрування

Будь-який шифр формально можна описати двома наступними перетвореннями [17]:

$$E_K(P) = C,$$

$$D_K(C) = P,$$

де P – відкритий текст (вхідний текст для шифрування),

C – зашифрований текст або шифротекст (результат шифрування),

K – ключ шифрування,

E_K – функція шифрування,

D_K – функція дешифрування.

Якщо для операцій шифрування й дешифрування використовується один і той же ключ, то він має бути приватним і такий шифр є симетричним. Якщо ж для розглянутих перетворень використовуються два різні ключі, то один з них є загальнодоступним, а інший – приватним, і відповідний шифр називається асиметричним.

Зважаючи на вищенаведене, поняття симетричного шифрування можна сформулювати наступним чином: це тип шифрування, в якому шифрування та дешифрування інформації здійснюється з використанням лише одного ключа (приватного). Якщо суб'єкти, обмінюються інформацією за допомогою симетричного шифрування, вони повинні спочатку якимось чином обрати єдиний ключ, який буде використаний у шифрі. Оскільки такий ключ є приватним, часто цей тип шифрування називають private key encryption (шифрування приватним ключем).

Розглянемо один з найдавніших і найпростіших прикладів симетричного шифрування – шифр Цезаря. Це моноалфавітний шифр заміни, який полягає в заміні літери відкритого тексту на літеру, що знаходиться в алфавіті на певну кількість позицій далі від неї. Формально такі перетворення можна сформулювати за допомогою наступних формул:

$$E(p) = (p + k) \bmod n,$$

$$D(c) = (c - k) \bmod n,$$

де p – літера алфавіту, з якого складається відкритий текст,

c – літера зашифрованого тексту,

k – ключ шифрування, який показує на скільки позицій в алфавіті відбувається зсув під час шифрування;

n – розмір алфавіту.

Таким чином, якщо використовується український алфавіт (n = 33 літери) й ключ шифрування k = 2, то відкритий текст P = «секретне повідомлення» буде зашифровано в шифротекст C = «ужмтжфпж сргйєронжппб». Знаючи ключ шифрування, це повідомлення дуже легко дешифрувати, використавши зсув літер в інший бік.

Попри те, що шифр Цезаря не застосовується на практиці, оскільки його можна легко зламати, він чудово ілюструє принципи симетричного шифрування.

Прикладами сучасних симетричних шифрів є AES (Advanced Encryption Standard), DES (Data Encryption Standard), IDEA (International Data Encryption Algorithm), RC5 (Rivest Cipher 5).

Перевагою симетричного шифрування є простота реалізації, яка є наслідком простих операцій, які лежать в основі шифру. Також побудова симетричних шифрів в загальному випадку менш обчислювально-витратна, ніж асиметричні, як наслідок, алгоритми симетричного шифрування є значно швидшими.

Недоліком таких шифрів є потреба забезпечення суб'єктів комунікації приватним ключем, який може бути перехопленим, тому його варто часто змінювати й передавати лише безпечними каналами передачі інформації. Іншою проблемою для симетричного шифрування є складність управління ключами шифрування (їх генерування, передача, зберігання й знищення), адже зі збільшенням суб'єктів кількість ключів квадратично зростає. Розглянуті недоліки зазвичай компенсують шляхом поєднання

асиметричного й симетричного шифрування: ключ передають за допомогою більш надійних асиметричних алгоритмів, а власне шифрування великої кількості даних здійснюють з використанням симетричних шифрів.

Ще однією важливою характеристикою шифрів, яка необхідна для розуміння моделі шифрування даних, є метод обробки відкритого тексту: блочне (паралельне шифрування блоків) чи потокове (послідовне шифрування байтів або декількох бітів). За умови блочного методу обробки, текст ділиться на частини певної довжини, які шифруються окремо, при цьому, якщо для останнього блоку не вистачає певної кількості символів, він зазвичай доповнюється нулями. Якщо ж шифр використовує потоковий метод обробки, то вхідний текст обробляється посимвольно.

2.1.2. Генеративні конкуруючі нейронні мережі

Головна ідея генеративних конкуруючих мереж, вже була розглянута в підрозділі 1.2, а зараз розглянемо структуру мереж генератора й дискримінатора більш детально.

Одним з найпоширеніших способів задання таких мереж є згорткові нейронні мережі (CNN – convolutional neural networks). Це клас глибинних нейронних мереж, які характеризуються прямим поширенням (інформація передається в одному напрямку) і наявністю щонайменше одного згорткового шару. Найчастіше такі мережі застосовують для роботи з зображеннями.

Архітектура таких нейронних мереж подібна до зв’язаності нейронів у людському мозку. Окремі нейрони мережі реагують на подразники в обмеженому полі зору, яке називається рецептивне поле. Множина таких полів накладається, щоб покрити все візуальне поле зображення.

Зображення в даній архітектурі представляється як множина пікселів, яка розгладжується й передається як багаторівневий перцептрон з метою класифікації.

Згорткова нейронна мережа складається з певної комбінації шарів серед яких [18]:

- згортковий (convolutional);
- агрегувальний (pooling);
- повністю з'єднані шари (fully connected layers);
- шар активації.

Розглянемо їх докладніше.

Згортковий шар (convolutional layer) – основний шар для побудови CNN архітектури. Параметри цього шару складаються з набору навчальних фільтрів (ядер), які мають невелике сприйнятливе поле, але поширюються на всю глибину вхідного об'єму даних. Під час прямого проходу кожен фільтр згортається по ширині та висоті вхідного поля, обчислюючи добуток між записами фільтра та входом, і створюючи двовимірну карту активації цього фільтру. Це дозволяє нейронній мережі вивчити фільтри, які активуються, коли виявляють певний тип функції в певному просторовому положенні на вході. Принцип роботи такого шару представлено на рис. 2.2.

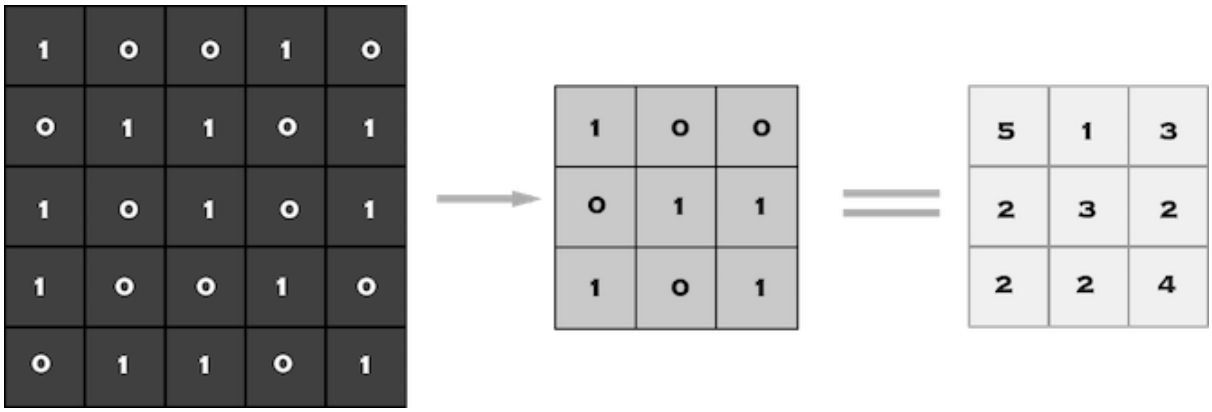


Рис. 2.2. Принцип роботи згорткового шару

Скалярний результат кожного згорткового шару потрапляє на нелінійну функцію активації (шар активації). Зважаючи на це, шар активації часто логічно об'єднують зі згортковим шаром. Найчастіше застосовується функція зрізаних лінійних вузлів (ReLU): $f(x) = \max(0, x)$. Ця функція активації дозволяє прискорити навчання й зменшити обчислювальні витрати (через її простоту), порівняно з функціями гіперболічного тангенса $f(x) = \tanh(x)$, $f(x) = |\tanh(x)|$ та сигмоїда $f(x) = \left(1 + e^{-x}\right)^{-1}$, які використовувалися до її відкриття.

Агрегувальний шар (pooling layer) – шар нейронної мережі, який є формою нелінійного зниження дискретизації. Існує декілька функцій для здійснення такого агрегування, найпоширенішою з яких є агрегування шляхом максимізації (max pooling) [19]. Приклад обробки даних таким шаром, зображено на рис. 2.3.

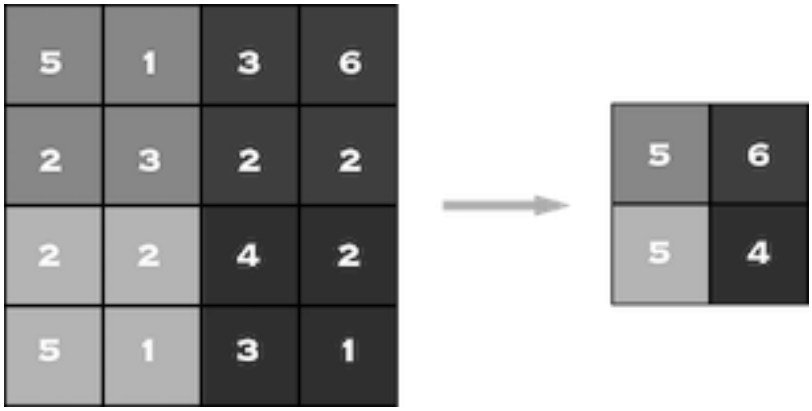


Рис. 2.3. Приклад обробки даних агрегувальним шаром

Вхідна матриця в цьому прикладі розподіляється на 4 блоки в кожному з яких знаходиться максимальне значення. Матриця максимальних значень і буде виходом агрегувального шару.

Повністю з'єднані шари (fully connected layers) – набір шарів, в яких кожен нейрон (параметр) одного шару з'єднаний з кожним нейроном (параметром) в іншому шарі. Оскільки часово-просторова

складність зменшується завдяки згортковому й агрегувальному шару, то можна побудувати повністю з'єднану мережу, щоб класифікувати зображення. Приклад повністю з'єднаних шарів зображено на рис. 2.4.

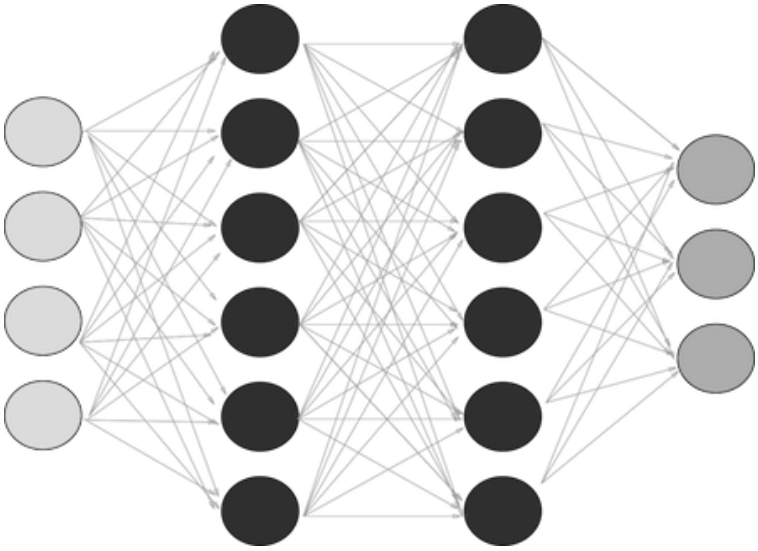


Рис. 2.4. Повністю з'єднані шари згорткової мережі

Розглянувши окремо складові згорткової нейронної мережі, тепер можна сформулювати приклад повного її вигляду, який зображено на рис. 2.5.

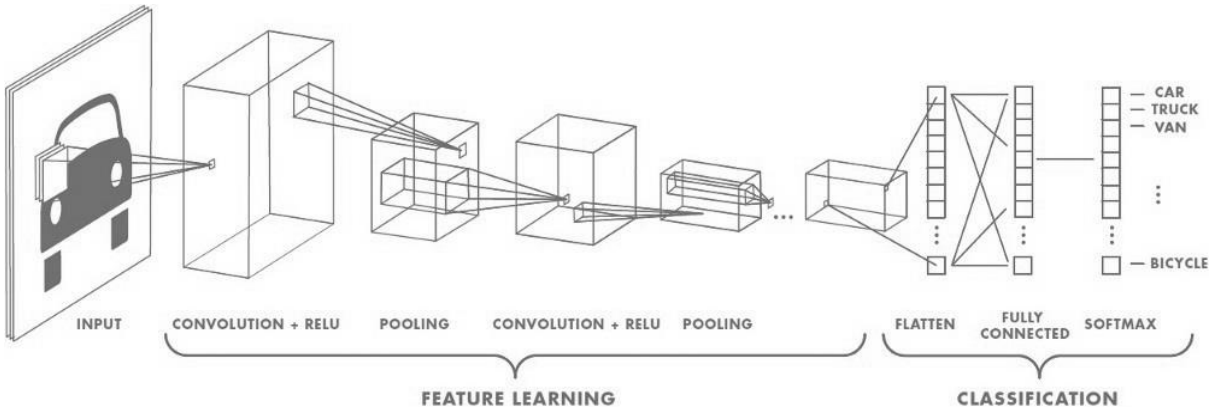


Рис. 2.5. Приклад згорткової нейронної мережі

Проаналізувавши згорткову нейронну мережу, що наведена на рисунку, можна зробити висновок, що спочатку зображення проходить

через набір згорткових, активаційних й агрегувальних шарів (кількість таких шарів залежить від складності зображення), де складність зображення зменшується, без втрат характерних ознак зображення. Далі дані згаджуються й проходять через повністю з'єднані шари, які завершуються функцією активації softmax: $P\left(y=j\middle|\mathbf{x}\right)=\frac{e^{\mathbf{x}^T\mathbf{w}_j}}{\sum_{i=1}^{K}e^{\mathbf{x}^T\mathbf{w}_k}}$, де $\mathbf{x}$ – вхідний вектор, $\mathbf{w}_i$ – вектори ваги моделі, а прогнозована ймовірність мітки $y = j$.

Зважаючи на те, що згорткові нейронні мережі вирішують задачу класифікації, моделі генератора і дискимінатора генеративної конкуруючої мережі можна побудувати з їх допомогою.

2.2. Складові моделі шифрування

Зображена на рис. 2.1, модель шифрування складається з 3-х нейронних мереж: «Alice», «Bob», «Eve». Розглянемо їх докладніше.

1. Мережа «Alice». Її завдання винайти надійний метод шифрування, який має забезпечити безпечний обмін повідомлень з мережею «Bob». Вхідними параметрами для мережі є початковий текст (*P*) та ключ (*K*), а вихідним – зашифрований текст (*C*).
2. Мережа «Eve». Перехопивши зашифрований текст (*C*), мережа намагається дешифрувати текст з метою отримання оригінальної інформації.
3. Мережа «Bob». Завданням цієї мережі є винайдення методу дешифрування, який з зашифрованого тексту (*C*) та ключа шифрування (*K*) має відтворити початковий текст (*P*).

Згідно з дослідженням [15, 20], при реалізації цієї моделі було обрано архітектуру «змішати та перетворити», яку було реалізовано за допомогою шарів згорткової мережі. Першим шаром цієї архітектури був

повністю з'єднаний шар згорткової мережі, де кількість виходів дорівнює кількості входів. Простий текст і ключові біти подаються в цей шар. Оскільки кожен вихідний біт може бути лінійною комбінацією всіх вхідних бітів, цей шар дозволяє (але не вимагає) змішування між бітами ключа і відкритого тексту (зокрема, цей шар може переставляти біти). Наступним шаром є послідовність згорткових шарів, останній з яких дає вихід (шифротекст) з розміром, що дорівнює розміру для відкритого тексту. Згорткові шари навчаються застосовувати певну функцію до груп бітів, змішаних попереднім шаром, без апріорного уточнення, якою має бути ця функція.

У цій моделі було використано протилежний порядок шарів згорткової мережі, яку використовують в програмах для обробки зображень. Нейронні мережі, розроблені для таких додатків, часто використовують згортки, щоб скористатися просторовими можливостями розташування пікселів. Для конкуруючої нейронної криптографії розташування біт зробили властивістю, яку нейронні мережі визначають під час навчання.

Таким чином, мережа «Alice», архітектура якої зображена на рис. 2.6, комбінує два N-бітні входи (ключ і відкритий текст), використовуючи вектор $2N \times 1$ для представлення бітових значень. Цей вектор обробляється через $2N \times 2N$ повністю з'єднаний шар, а потім надсилається через послідовність чотирьох 1-D згорткових шарів (фільтрами для яких є відповідно [4, 1, 2], [2, 2, 4], [1, 4, 4], і [1, 4, 1], з кроками 1, 2, 1, 1). У якості функції активації було використано сигмоїдну нелінійну функцію після кожного шару, окрім заключного, для якого застосовано гіперболічний тангенс. Мережа «Bob» має ідентичну структуру. Мережа «Eve» приймає лише шифротекст як вхідні дані, тому її перший шар змінений на $N \times 2N$ повністю з'єднаний шар.

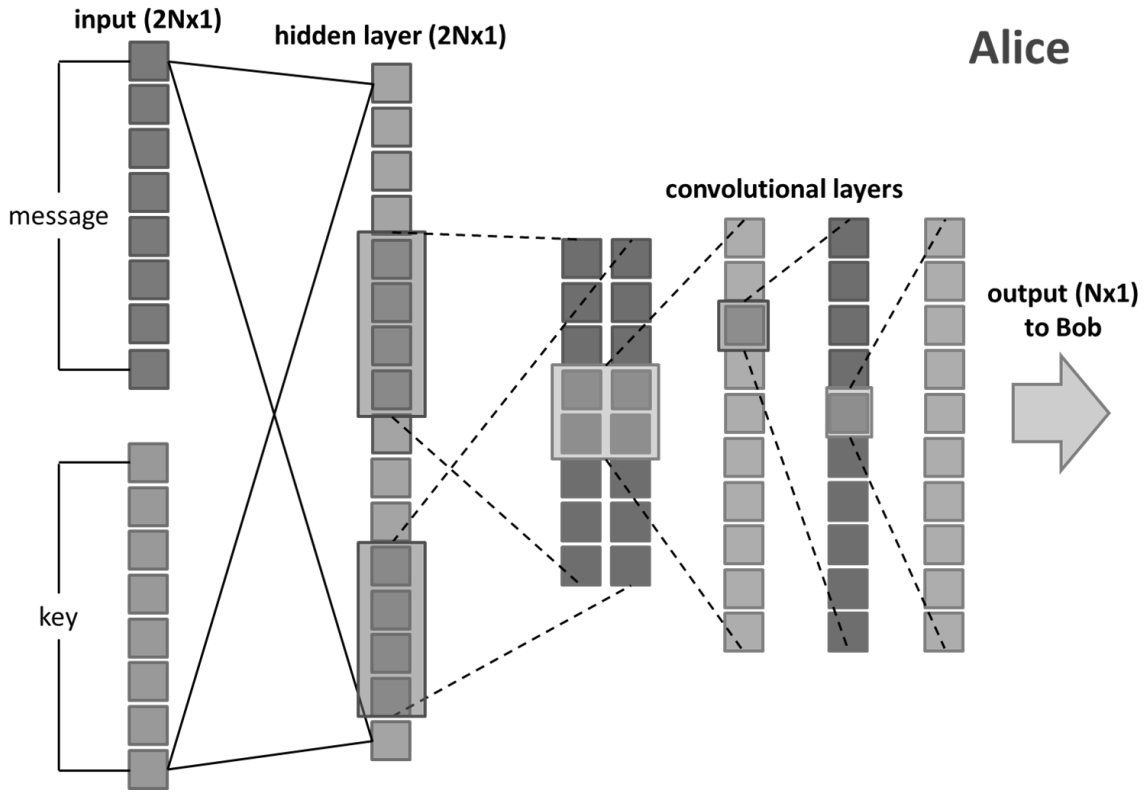


Рис. 2.6. Архітектура мережі «Alice»

Використовуючи генеративні методики конкуруючих мереж, мережі «Alice» і «Bob» спільно навчаються, щоб безпечно обмінюватись повідомленнями, мета їх навчання – перемогти мережу «Eve». Кожна мережа попередньо не знає жодних криптографічних алгоритмів та методик, які можна було б використати для досягнення своєї мети. Варто зауважити, що згідно з принципами GAN мережі «Alice» і «Bob» намагаються перемогти кращу версію мережі «Eve», а не фіксовану її версію.

Згідно з розглянутою моделлю, необхідно використовувати дві різні функції втрат навчання [21-23]:

1. Для мережі «Eve» функція втрат дорівнює відстані Хеммінга (d) між бітовим вектор відкритого тексту (P) і результатом дешифрування тексту мережею «Eve» (P_{Eve}). Цю функцію втрат яку також можна визначити у вигляді норми l_1, що застосовується

до результату бітової операції XOR між зазначеними векторами. Математично це можна сформулювати наступним чином:

$$L_{Eve} = d\left(P, P_{Eve}\right) = \sum_{j=1}^{N} \left| P_j - \left(P_{Eve}\right)_j \right| = \left\| P \oplus P_{Eve} \right\|_1 .$$

2. Функцією втрат для мереж «Alice» і «Bob» також є абсолютна помилка дешифрування (відстань Хеммінга між вектором початкового тексту (P) і результатом дешифрування мережею «Bob» (P_{Bob})), проте вона коригується додатковим параметром (($N/2 - L_{Eve}$)^2 / ($N/2$)^2, де N – довжина початкового тексту в бітах), який показує наскільки ефективно мережа «Eve» дешифровує зашифрований текст. Таким чином, розглянуту функцію втрат можна подати у вигляді формули:

$$L_{Bob} = \left\| P \oplus P_{Bob} \right\|_1 + \frac{\left(\frac{N}{2} - L_{Eve}\right)^2}{\left(\frac{N}{2}\right)^2} .$$

Гіперпараметри моделі шифрування були визначені наступним чином [15, 20]:

- розмір міні партій даних (mini-batch size) становить від 256 до 4096 бітів;
- використовується оптимізатор Адама з коефіцієнтом швидкості навчання 0,0008;
- навчання відбувається наступним чином: протягом однієї міні партії даних мережа «Eve» є фіксованою, а мережі «Alice» і «Bob» навчаються, а потім на дві міні партії даних мережі міняються місцями (відбувається навчання мережі «Eve», а мережі «Alice» і «Bob» є фіксованими).

Результати навчання моделі шифрування, що базована на конкуруючій нейронній криптографії представлені на рис. 2.7 [15].

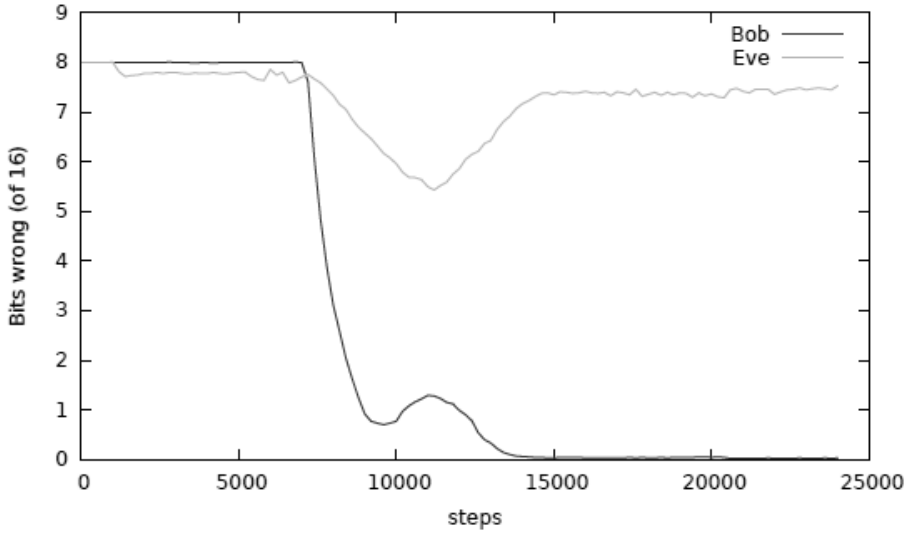


Рис. 2.7. Результати навчання моделі шифрування

Проаналізувавши рисунок, можна зробити висновок, що необхідно здійснити близько 8000 навчальних кроків, щоб мережі «Bob» і «Eve» починали відтворювати відновлювати початкове повідомлення. Необхідно ще приблизно 2000 кроків, щоб мережі «Alice» та «Bob» виявили це й ускладнили шифрування, таким чином, щоб рівень помилок дешифрування мережею «Eve» знову піднявся. Іншими словами, саме в цей момент мережі навчились захищати комунікацію, щоб уникнути нападу, продовжуючи підвищувати свою ефективність.

2.3. Висновки

Проаналізовано модель шифрування даних з використанням генеративних конкуруючих нейронних мереж, яка здійснює шифрування наборів біт. В основі моделі лежить поєднання двох підходів: принципів симетричного шифрування й концепції генеративних конкуруючих нейронних мереж.

В основі нейронних мереж, що використовуються в моделі шифрування лежать шари згорткової мережі, розміщені в зворотному

порядку, що дозволяє їм навчатись без знань жодних криптографічних алгоритмів та методик.

Також було розглянуто функції втрат нейронних мереж і гіперпараметри, що були використані при побудові даної моделі.

3. ЗАПРОПОНОВАНИЙ МЕТОД ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ НЕЙРОННИХ МЕРЕЖ

3.1. Метод шифрування даних

Метод шифрування набору даних, що представлені у вигляді зображень, можна сформулювати у вигляді наступних кроків:

1. Попередня обробка даних, шляхом зменшення розмірності вхідних зображень.
2. Застосування модифікованої моделі шифрування, загальна ідея якої розглянута у розділі 2, до кожного елемента набору даних.
3. Формування нового набору даних з зашифрованих елементів з ключем, що був використаний для шифрування.
4. Класифікація зображень на зашифрованих даних.

Розглянемо їх докладніше.

На етапі попередньої попередньої обробки даних, застосовуємо до зображень набір згорткових та агрегувальних шарів, що дозволить зменшити розмірність вхідних даних (кількість пікселів).

Попередньо здійснюється навчання модифікованої моделі шифрування, що базується на використанні двовимірних згорткових шарів, та дозволяє шифрувати матриці пікселів зображень. Докладніше модель буде розглянуто нижче.

На другому етапі до кожного зображення застосовується модифікована модель шифрування даних з однаковим ключем шифрування.

Далі з зашифрованих даних формується новий набір даних, важливим елементом для якого є приватний ключ шифрування. Процес формування набору даних представлено на рис. 3.1.

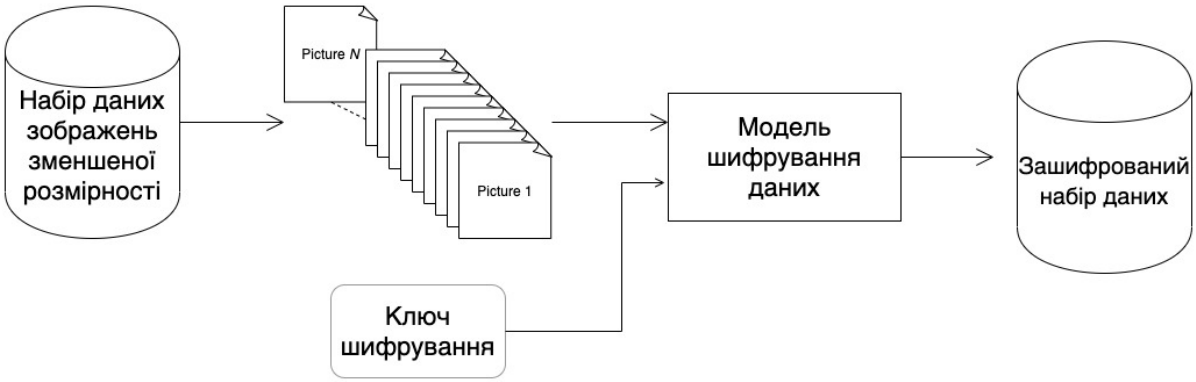


Рис. 3.1. Процес формування нового набору даних

На останньому етапі відбувається класифікація зашифрованих зображень. Незалежно, від обраного методу класифікації Її можна виконувати двома способами:

1. На основі зашифрованих зображеннь без ключа шифрування.

2. Використовуючи зашифровані зображення і ключ шифрування.

У першому випадку достатньо просто використати метод класифікації, в той час як, для другого способу необхідно побудувати додатковий повністю з'єднаний згортковий шар на основі ваг мережі «Bob», яка, під час навчання моделі шифрування даних, навчилась з використанням ключа шифрування відтворювати зображення зменшеної розмірності.

3.2. Модифікована модель шифрування даних

Класична модель шифрування даних використовується для шифрування інформації представленої у вигляді набору біт. Проте, такий спосіб представлення даних є не завжди доцільним. Зокрема, для візуальних даних, наприклад зображень, більш природнім представленням інформації є множина пікселів.

Структура генеруючої мережі «Alice» у модифікованій моделі:

- вхідна інформація: матриця пікселів зображення зменшеної розмірності (матриця N на N) та ключ шифрування (матриця N на N).
- прихований шар: повноз'єднальний шар, N на $2*N$,
- згорткові шари: три двовимірні згорткові шари (Conv2D), у якості функції активації пропонується використовувати сигмоїдну нелінійну функцію після кожного шару;
- вихідна інформація: зашифрована матриця пікселів зображення (матриця N на N).

Структура мережі «Bob» така ж як і мережі «Alice», а структура мережі «Eve» відрізняється лише першим шаром.

Запропонована модифікована модель шифрування даних відрізняється від існуючої архітектурою генеруючої й дискримінуючої мереж. Пропонується використовувати двовимірні згорткові нейронні мережі, замість одновимірних. Це дозволить спростити використання моделі для даних, що представлені у вигляді зображень.

3.3. Вибір метрик для оцінки запропонованих методу й модифікованої моделі

Одним з основних етапів побудови методів, алгоритмів та моделей в системах аналізу даних і штучного інтелекту є оцінка їх ефективності. Адаптивність моделі визначається тим, наскільки добре вона робить передбачення на досі не відомих їй даних (небачених нею даних). Відсутність належної оцінки AI-моделі з використанням різних показників може призвести до проблеми, коли побудована модель здійснює не досить правильні прогнози на невідомих (користувацьких) даних. У таких випадках, моделі не навчаються, а запам'ятовують правильні відповіді,

внаслідок чого вони не можуть досить добре узагальнити їх на нові для них дані.

Існує багато метрик оцінки систем штучного інтелекту, що застосовуються залежно від задачі, яка вирішується за допомогою машинного навчання. Зважаючи на це, існують різні показники для завдач класифікації, регресії, ранжування, кластеризації тощо. Однак, деякі метрики, наприклад, точність, застосовуються для декількох завдань.

Зважаючи на те, що запропонований метод шифрування даних пропонується застосовувати для вирішення задачі класифікації зашифрованих зображень, розглянемо докладніше метрики оцінки ефективності моделей класифікації даних. Спочатку сформулюємо метрики для задачі бінарної класифікації, яка полягає у визначенні того, чи належить екземпляр даних до позитивного класу (наприклад, спам – це позитивний клас для задачі фільтрації спаму в електронній. пошті), а потім узагальнимо їх для багатокласової класифікації.

Перш за все, визначимо матрицю помилок (confusion matrix) або матрицю невідповідності (як її ще називають). Така матриця є основою для визначення метрик оцінки моделі. Її структура представлена в табл. 3.1.

Таблиця 3.1

Матриця помилок в задачах машинного навчання

		Передбачення	
		Позитивний клас	Негативний клас
Реальний результат	Позитивний клас	True Positive (*TP*)	False Negative (*FN*)
	Негативний клас	False Positive (*FP*)	True Negative (*TN*)

З матриці помилок визначають такі властивості моделі:

- True Positive (*TP*) – кількість передбачень моделі, коли позитивний клас було визначено правильно;
- True Negative (*TN*) – кількість передбачень моделі, коли негативний клас було визначено правильно;
- False Positive (*FP*) – кількість передбачень моделі, коли позитивний клас було визначено неправильно;
- False Negative (*FN*) – кількість передбачень моделі, коли негативний клас було визначено неправильно.

Точність моделі до певного значення (*Accuracy*) в контексті класифікаційних моделей можна визначити як відношення правильно класифікованих зразків $N_{correct}$ до загальної кількості зразків N_{total} [24]:

$$Accuracy = \frac{N_{correct}}{N_{total}}.$$

Для бінарної класифікації точність до певного значення (*Accuracy*) може бути визначена наступним чином:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{N_{total}}.$$

При побудові систем намагаються зробити їх високоточними, однак, не завжди точності достатньо для забезпечення ефективності на невідомих даних. Зокрема, лише критерій точності моделі не варто застосовувати до наборів даних з незбалансованими класами. У цьому випадку, доцільніше використати інші метрики, наприклад, точність позитивних передбачень або прецизійність (*Precision*) та повноту (*Recall*).

У задачі класифікації точність позитивних передбачень або прецизійність (*Precision*) класу визначається співвідношенням кількості правильних позитивних результатів (*TP*) і загальної кількості елементів, що мають мітку позитивного класу, тобто

$$Precision = \frac{TP}{TP + FP}.$$

Висока точність показує, що модель передбачила значно більш до релевантних (доречних) результатів, ніж нерелевантних.

У контексті задачі класифікації повнота (*Recall*) – це співвідношення правильних позитивних результатів (*TP*) і загальної кількісті елементів, які фактично належать до позитивного класу, іншими словами – частота правильних рішень моделі, яка визначається за формулою [25]:

$$Recall = \frac{TP}{TP + FN}.$$

Високий рівень повноти показує, що модель передбачила більшість релевантних результатів.

Часто використовують метрику міра F_1, яка поєднує розглянуті метрики точність позитивних передбачень (*Precision* та *Recall*) й обчислюється як їх середнє гармонійне:

$$F_1 = \frac{2 \cdot (Precision \cdot Recall)}{Precision + Recall}.$$

Міра F_1 зменшує вплив екстремальних значень у наборі даних.

Ще однією метрикою, що використовується для оцінки ефективності бінарної класифікації є ROC-крива. Це графік, що показує ефективність класифікаційної моделі при всіх порогових параметрах класифікації. Крива складається з двох параметрів:

- True Positive Rate (*TPR*) – рівень правильних позитивних результатів, який є синонімом повноти й визначається як:

$$TruePositiveRate(TPR) = \frac{TP}{TP + FN}.$$

- False Positive Rate (*FPR*) – рівень неправильних позитивних результатів, який визначається наступним чином:

$$FalsePositiveRate(FPR) = \frac{FP}{FP + TN}.$$

Приклад ROC-простору зображено на рис. 3.2.

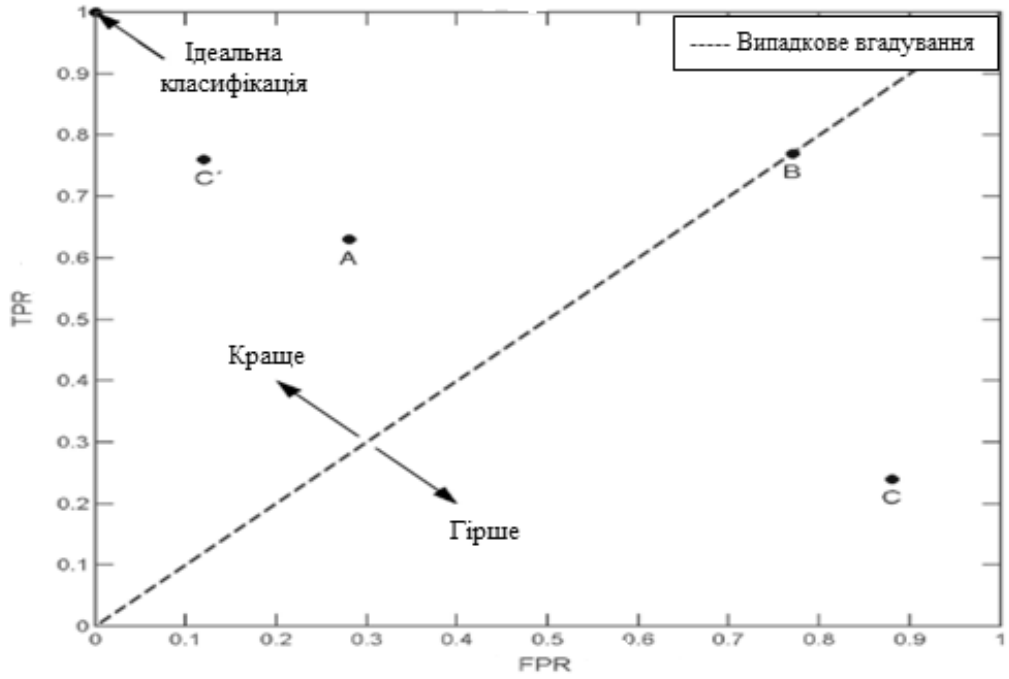


Рис. 3.2. Приклад ROC-простору

Згідно з графіком точка (0;0) відповідає стану класифікатору, який ніколи не робить передбачень приналежності до позитивного класу; точка (0;1) відображає модель, яка завжди класифікує зразки як позитивний клас. Ці дві точки, а також точка *B* лежать на прямій випадкового вгадування, відповідно класифікатор, що показує такі результати не ефективно здійснює класифікацію (вона еквівалентна простому вгадуванню класу). Найбільш ефективний класифікатор має значення метрики в точці (0;1), найближче до якої зображена точка *C'*. Варто зауважити, що ROC-кривої не чутливі до змін розподілу класів (до їх збалансованості), тому при зміні кількості позитивних і негативних зразків вигляд кривої не зміниться. Кількісною характеристикою такої кривої є міра *AUC* – площа під кривою, яка визначає сукупний показник ефективності для всіх можливих порогів класифікації.

Узагальнимо розглянуті метрики бінарної класифікації, шляхом введення матриці помилок для багатокласової, класифікації яка представлена у табл. 3.2.

Таблиця 3.2

Матриця помилок для багатокласової класифікації

		Передбачення			
		Клас 1	Клас 2	…	Клас N
Реальний результат	Клас 1	X_{11}	X_{12}	…	X_{1N}
	Клас 2	X_{21}	X_{22}	…	X_{2N}
	…	…	…	…	…
	Клас N	X_{N1}	X_{N2}	…	X_{NN}

Метрики оцінки ефективності вирішення багатокласової класифікації для i-го класу можна сформулювати наступним чином:

- $TP_i = X_{ii}$;
- $TN_i = \sum\limits_{\substack{j=1 \\ j \neq i}}^{N} \sum\limits_{\substack{k=1 \\ k \neq i}}^{N} X_{jk}$;
- $FP_i = \sum\limits_{\substack{j=1 \\ j \neq i}}^{N} X_{ji}$;
- $FN_i = \sum\limits_{\substack{j=1 \\ j \neq i}}^{N} X_{ij}$;
- $\left(Precision\right)_i = \dfrac{TP}{TP + FP_i} = \dfrac{\sum\limits_{j=1}^{N} X_{jj}}{\sum\limits_{j=1}^{N} X_{jj} + \sum\limits_{\substack{j=1 \\ j \neq i}}^{N} X_{ji}}$;
- $\left(Recall\right)_i = \dfrac{\sum\limits_{j=1}^{N} X_{jj}}{\sum\limits_{j=1}^{N} X_{jj} + \sum\limits_{\substack{j=1 \\ j \neq i}}^{N} \sum\limits_{\substack{k=1 \\ k \neq i}}^{N} X_{jk}}$.

Для обчислення узагальнених метрик багатокласової класифікації достатньо знайти середнє арифметичне значення відповідних метрик кожного класу.

Головною метою запропонованого методу є захист приватних даних, при якому зашифровані дані можуть бути використані при вирішенні задачі класифікації за допомогою штучного інтелекту. Для перевірки роботи методу було обрано збалансований набір даних MNIST. Зважаючи на це, головним критерієм для оцінки якості побудованої системи було обрано точність (*Accuracy*) класифікації зашифрованих і оригінальних даних, що здійснюється за допомогою різних класифікаційних алгоритмів.

Задачею модифікованої моделі є спрощення шифрування зображень, тому для оцінки моделі будуть використані похибка дешифрування даних, швидкість шифрування та тривалість навчання моделі шифрування (кількість ітерацій навчання).

3.4. Висновки

Запропоновано метод шифрування наборів даних, що дозволяє використовувати приватні дані в публічних системах штучного інтелекту й аналізу даних шляхом зменшення їх розмірності і шифрування з приватним.

Модифіковано модель шифрування даних, шляхом використання двовимірних згорткових нейронних мереж, що дозволяє використовувати модель для шифрування зображень.

Визначено метрики для оцінки запропонованого методу шифрування даних й модифікованої моделі. Для методу шифрування даних ключовою метрикою є точність класифікації зашифрованих даних. Оцінкою модифікованої моделі буде кількість ітерацій навчання та швидкість шифрування вхідних даних.

4. РОЗРОБЛЕННЯ ПРОГРАМНОЇ СИСТЕМИ ДЛЯ ДОСЛІДЖЕННЯ МЕТОДІВ ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ НЕЙРОННИХ МЕРЕЖ

4.1. Обґрунтування вибору технологій для вирішення поставленої задачі

Для розроблення програмної системи, що реалізовує запропоновані метод та модифікацію моделі шифрування даних, що представлені у вигляді зображень, необхідно здійснити вибір наступних технологій:

- мова програмування;
- технології для глибинного навчання;
- засоби візуалізації.

Проведемо докладний аналіз існуючих технологій та визначимо, які з них доцільно використовувати для вирішення поставленої задачі.

4.1.1. Вибір мови програмування

Для побудови систем штучного інтелекту можна використовувати різні мови програмування, зокрема Python, R, C/C++, Scala, Java, JavaScript тощо. Проаналізуємо їх переваги й недоліки.

Основною мовою для побудови систем штучного інтелекту вважається Python, оскільки ця мова має простий синтаксис, широкий вибір бібліотек, зокрема для роботи з даними (їх перетворень та візуалізації), а також більшість бібліотек цієї мови є крос-платформними, що дозволяє виконувати текст програми на пристроях з різними операційними системами [26-28]. Крім цього, мова Python масштабована й підтримує функціональний, об’єктно-орієнтований та процедурний стиль написання програм. Також спільнота розробників, що використовують цю мову є досить великою, що спрощує пошук методів вирішення проблем, що часто зустрічаються. Головний недолік мови, який є ціною за простоту

й різні рівні абстракції, – час виконання програм, який більший порівняно з мовою C++.

Мова R – це статистична мова, що була створена і використовується для прогнозної аналітики, статистичного аналізу й візуалізації даних [27]. Перевагами мови R є безкоштовність використання (на відміну, від комерційної мови Matlab, що використовується зі схожою метою), наявність великої кількості методів для роботи з даними та можливість виконувати текст програм на різних операційних системах. Недоліками цієї мови є невелика кількість засобів роботи з алгоритмами машинного навчання; виконання програм здійснюється повільніше, ніж мовою Python (зокрема під час аналізу широкомасштабних даних); а також складний синтаксис, що підвищує «поріг входження» в неї.

Іншою мовою, що може бути застосована для побудови систем штучного інтелекту є C++. Ця мова є найшвидшою для таких систем (швидке виконання коду, кращий рівень контролю ефективністю), тому її використовують в проєктах, де час дуже важливий і необхідний правильний розподіл ресурсів [26]. Зважаючи на це, мова C++ зазвичай використовується для оптимізації ресурсно-витратних систем. Недоліками розглянутої мови є складність синтаксису мови, розроблення та підтримки системи; а також відсутність збирача сміття (garbage collector), що покладає всю відповідальність за роботу з пам’яттю на розробника.

При роботі з великими даними мовою програмування доцільно обирати мову Scala, яка має високий рівень підтримки багатозадачності, що дозволяє обробляти великі масиви даних [27]. Зокрема, ця мова має набір бібліотек, що характеризуються високою продуктивністю під час аналізу даних, серед яких Saddle, Scalalab та Breeze; а також вона підтримує фреймворк Hadoop для розподіленої обробки даних. Однак, кількість інструментів мови Scala для машинного навчання значно менша, ніж у мов Python та R.

У зв’язку зі значним розвитком штучного інтелекту протягом останніх років, багато мов програмування (Java, JavaScript, C# тощо) додають підтримку засобів для інтелектуальної роботи з даними. Однак, станом на зараз вони поступаються розглянутим вище мовам.

Спираючись на проведений аналіз, мовою для розроблення програмного забезпечення було обрано Python, адже вона має простий синтаксис, великий набір засобів для побудови систем штучного інтелекту, а також є найбільш використовуваною для таких систем.

4.1.2. Вибір технологій для глибинного навчання

Для побудови моделей глибинного навчання зараз існує багато технологій, серед яких Tensorflow, PyTorch, Keras, MXNet [29].

Платформа Tensorflow, яка була розроблена компанією Google Brain у 2015 році [30], це популярний обчислювальний фреймворк для створення моделей машинного навчання, який підтримує різні набори інструментів для побудови моделей на різних рівнях абстракції. Головними перевагами фреймворку є масштабованість, великі можливості для налагодження моделей, візуалізація обчислень, висока продуктивність, дозволяє використовувати різні апаратних можливостей комп’ютера (CPU, GPU). Платформа Tensorflow підтримує різні мови програмування, серед яких Python, R, C++ та інші. Її недоліками є високий «поріг входження» (складний синтаксис, вимагає фундаментальних знань з математичного аналізу й лінійної алгебри, а також розуміння методів машинного навчання); нижча швидкість обчислень порівняно з конкурентами.

Keras – високорівневий фреймворк для глибинного навчання з відкритим кодом. Keras може працювати як надбудова над такими платформами: TensorFlow, Theano, Microsoft Cognitive Toolkit, R або PlaidML [31]. Цей фреймворк має нескладний синтаксис та дозволяє реалізовувати моделі з використанням таких блоків як: шари нейронної

мережі, цільові функції, функції активації, метрики тощо [31, 32]. Наприклад, за допомогою класу Sequential (tf.keras.Sequential) можна, можна побудувати моделі з різною послідовністю й кількістю згорткових шарів, як показано в лістингу 4.1.

Лістинг 4.1. Приклад побудови нейронної мережі за допомогою класу Sequential

```
      model = Sequential()
      model.add(Conv2D(28, kernel_size=(3, 3),
input_shape=input_shape))
      model.add(MaxPooling2D(pool_size=(2, 2)))
      model.add(Flatten())  # Flattening the 2D arrays for fully
connected layers
      model.add(Dense(128, activation=tf.nn.relu))
      model.add(Dropout(0.2))
      model.add(Dense(10, activation=tf.nn.softmax))
      model.compile(optimizer='adam',
                    loss='sparse_categorical_crossentropy',
              metrics='accuracy')
```

Фреймворк Keras надає можливості роботи як з текстом, так і з зображеннями. Також він підтримує згорткові та рекурентні нейронні мережі і дозволяє зберігати ваги натренованих мереж. Недоліком фреймворку Keras є менша гнучкість при побудові моделі, адже він перебуває на вищому рівні абстракції, ніж TensorFlow.

Ще однією технологією для глибинного навчання є бібліотека PyTorch, яка на відміну від TensorFlow, дозволяє вносити зміни в архітектуру моделі в процесі навчання [29]. Ця бібліотека має простий синтаксис, підтримує паралелізм та розподілену модель навчання. Недоліками цієї бібліотеки є не достатня кількість інтерфейсів моніторингу та візуалізації, менша спільнота користувачів та відсутність гнучкої системи для обслуговування моделей машинного навчання, що розроблені з метою використання у виробництві.

Інструмент глибинного навчання MXNet призначений для використання моделей на різних пристроях. Ця бібліотека підтримує велику кількість мов програмування, серед яких C++, Python, R, Julia, JavaScript, Scala, Go [29]. Основна перевага бібліотеки в тому, що вона дозволяє виконувати ефективні паралельні обчислення з використанням багатьох графічних процесорів. Зокрема, ця технологія використовується у веб-сервісах Amazon. Бібліотека MXNet має невелику спільноту користувачів, однак, її документація є досить детальною, що сприяє простоті у її використанні.

Зважаючи на проведений аналіз, для розроблення системи було обрано платформу TensorFlow та її надбудову – фреймворк Keras, адже ці технології мають достатній рівень абстракції для побудови системи, що реалізує запропонований метод та модифікацію моделі, а саме можливість побудови нейронної мережі, що складається з набору шарів, використання різних метрик оцінки моделі та можливість збереження/завантаження ваг моделі.

4.1.3. Вибір засобів візуалізації

Не менш важливою задачею під час розроблення системи, є візуалізація отриманих результатів та зображень, метод шифрування яких було запропоновано. У мові програмування Python існує багато бібліотек, що вирішують таку задачу. Дві найбільших з них – бібліотеки Matplotlib та Seaborn.

Для розробки програмної системи було обрано бібліотеку Matplotlib, адже вона дозволяє будувати високоякісні графіки й діаграми, є кросплатформною й поєднується з бібліотекою Numpy, що використовується для роботи з чисельним представленням даних; в той час як бібліотека Seaborn показує кращі результати для табличних даних, завдяки інтеграції з бібліотекою Pandas.

4.2. Архітектура системи

Розроблена програмна система складається з двох модулів:

- модуль шифрування даних;
- модуль класифікації.

Модуль шифрування даних дозволяє здійснювати шифрування даних як запропонованим методом, так і класичною моделлю шифрування.

Модуль класифікації даних реалізовано таким чином, що задачу класифікації можна вирішувати використовуючи оригінальні та зашифровані дані (з ключем і без ключа).

На рис. 4.1 представлено діаграму класів розробленої програмної системи.

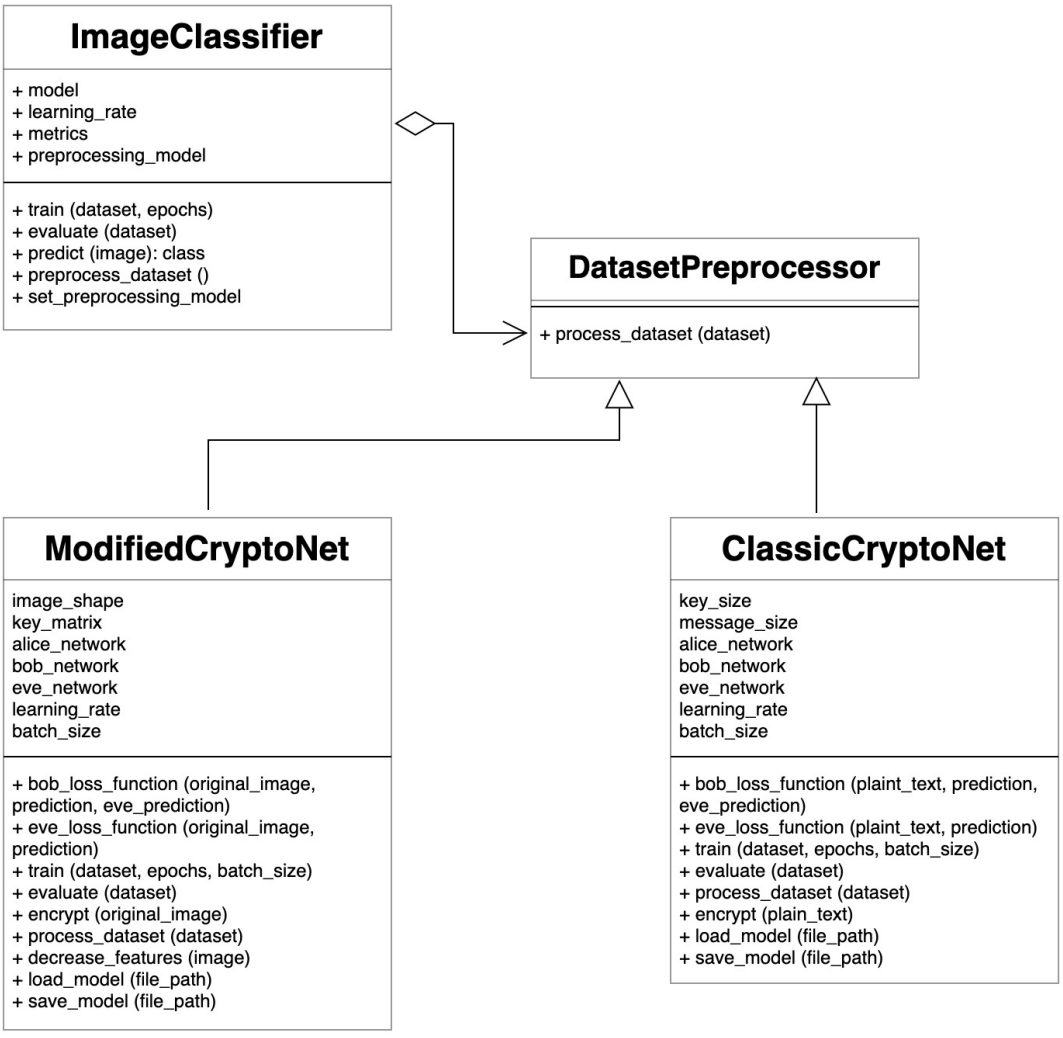


Рис. 4.1. Діаграма класів розробленої програмної системи

Згідно з діаграмою класів, розроблена система складається з наступних класів: DatasetPreprocessor, ClassicCryptoNet, ModifiedCryptoNet та ImageClassifier. Перші три класи входять до модулю шифрування даних, а останній – до модулю класифікації даних.

Клас DatasetPreprocessor – клас з єдиним методом process_dataset, в якому здійснюється попередня обробка набору даних. Реалізація цього методу в класі відсутня за замовчуванням (виконується pass оператор, під час виконання якого нічого не відбувається, але такий оператор є необхідним за синтаксисом мови Python). У класах-наслідниках цього класу відбувається шифрування в контексті запропонованого методу.

Клас ClassicCryptoNet – реалізація класичної моделі шифрування, що була докладно розглянута в розділі 2.

Клас ModifiedCryptoNet – реалізація модифікованої моделі, яка була запропонована в підрозділі 3.2.

Варто зазначити, що оскільки навчання генеративних конкуруючих нейронних мереж – досить тривалий процес, то з метою збереження результатів навчання в класах ClassicCryptoNet, ModifiedCryptoNet було реалізовано метод save_model, що дозволяє зберігати ваги моделі, та метод load_model, завдяки якому можна завантажити ваги моделі в систему й відновити її стан.

Клас ImageClassifier – клас, що здійснює класифікацію зображень, попереднім етапом якої є виклик методу process_dataset з класу DatasetPreprocessor, що дозволяє виконувати класифікацію як зашифрованих, так і оригінальних даних.

Отже, для розроблення програмної системи було використано шаблон проєктування «Стратегія» [33, 34], завдяки чому була здійснена інкапсуляція алгоритму попередньої обробки вхідного набору даних, що дозволяє замінювати конкретну реалізацію в екземплярі класу класифікатор без застосування наслідування.

4.3. Аналіз експериментальних результатів

Експериментальні дослідження проводились на комп'ютері MacBook Pro на операційній системі macOS Catalina з наступними характеристиками: CPU 2.3 ГГц Quad-Core Intel Core i5, оперативна пам'ять 8 Гб. Алгоритми реалізовані мовою програмування Python.

Для дослідження було обрано набір даних MNIST. Це набір монохромних зразків (зображень у відтінках сірого) рукописного написання арабських цифр (0-9). Він складається з 60000 зображень навчального набору та з 10000 зображень для тестування. Кожне зображення з набору даних має розмірність 28 на 28 пікселів. Приклад екземпляру з набору даних зображено на рис. 4.2.

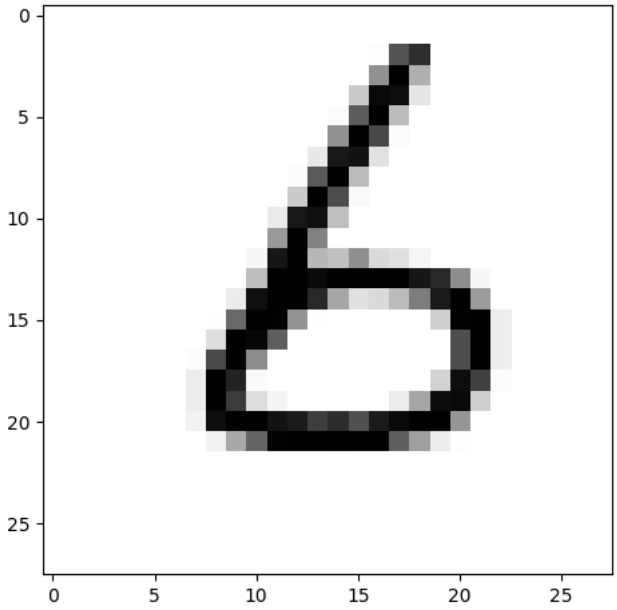


Рис. 4.2. Приклад екземпляру даних з набору даних MNIST

Обраний набір даних є збалансованим і складається з 10 класів, кількість елементів яких представлена у табл. 4.1.

Таблиця 4.1

Збалансованість класів набору даних MNIST

	Кількість елементів класів									
Набір даних	«0»	«1»	«2»	«3»	«4»	«5»	«6»	«7»	«8»	«9»
Для навчання	5923	6742	5958	6131	5842	5421	5918	6265	5851	5949
Для тестування	1001	1127	991	1032	980	863	1014	1069	945	978

Відповідно до найкращих практик розробки систем штучного інтелекту [32], набір даних необхідно розділяти щонайменше на дві незалежні частини: набір для навчання та набір для тестування. Окрім цього, вважається доцільним, застосування набору даних для затвердження (validation set) – підмножина набору даних, яка використовується для об’єктивної оцінки моделі на етапі навчання, що дозволяє регулювати гіперпараметри моделі під час навчання. Цю підмножини даних обирають з набору даних для навчання й такі дані не використовуються для навчання моделі. Перевірка моделі на наборі даних для затвердження здійснюється в кінці кожної епохи навчання моделі.

Зважаючи на це, набір даних було розділено наступним чином:

- набір навчальних даних для моделей шифрування – 5000 зображень з навчального набору MNIST (з них 10% – набір даних для затвердження (validation set));
- навчальний набір даних для вирішення задачі класифікації – інші 55000 зображень з навчального набору MNIST (з них 20% – набір даних для затвердження (validation set));
- тестовий набір даних для вирішення задачі класифікації – 10000 зображень з навчального набору MNIST;

4.3.1. Модифікація моделі шифрування даних

Для досліджень модифікованої моделі шифрування даних було обрано наступні гіперпараметри:

- розмірність зображення 28 на 28 пікселів;
- коефіцієнт швидкості навчання (learning rate) – 0,0015;
- метод оптимізації – оптимізатор Адам (метод адаптивної оцінки моментів), який, згідно з останніми дослідженнями, показує найкращі результати при вирішенні задачі оптимізації.
- розмір партії даних (batch size) – 150 зображень.
- кількість епох навчання – 30.

Експериментальні результати тривалості навчання модифікованої моделі зображено на рис. 4.3.

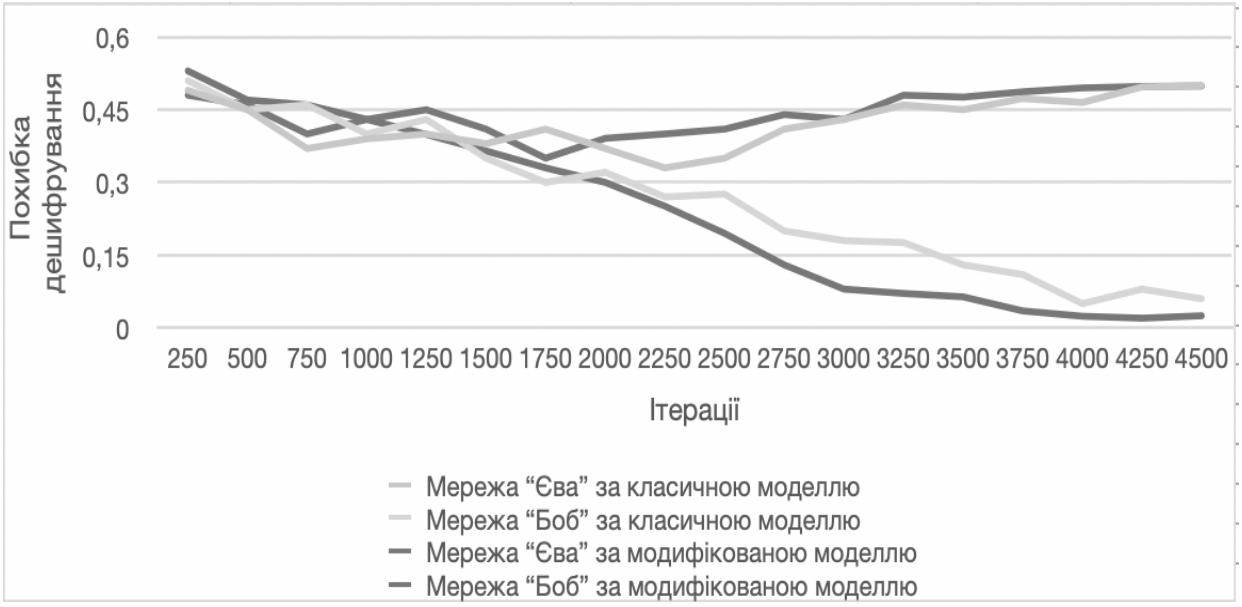


Рис. 4.3. Експериментальні результати тривалості навчання модифікованої моделі шифрування даних

Відповідно до графіку, на початкових ітераціях (1-1000) нейронні мережі «Eve» та «Bob» класичної та модифікованої моделей не можуть дешифрувати зображення. Протягом ітерацій 1000-1750 ці мережі

починають зменшувати похибку дешифрування на 10-15 %, що свідчить про те, що вони починають розуміти принципи шифрування, які застосовує мережа «Alice», при цьому результат дешифрування з ключем і без нього відрізняється несуттєво. Під час наступної 1000 ітерацій вплив наявності ключа шифрування стає більш суттєвим, відповідно мережа «Bob» починає успішно дешифрувати повідомлення, а помилка мережі «Eve» поступово збільшується до 50 %, що фактично відповідає випадковому вгадуванню інформації.

Модифікована модель приблизно у 1,4 рази швидше досягає необхідного рівня похибки дешифрування мережею «Bob» (менше 0,1): на 2850 ітерації, на відміну від класичної моделі, де такий самий рівень похибки на ітерації 3950. Також необхідний рівень дешифрування мережею «Eve» (більше 0,49) досягається на ітерації 3250 модифікованою моделлю шифрування, а класичною – на ітерації 4250. Зважаючи на це, класичну модель можна використовувати для шифрування, починаючи з ітерації 4250, а модифіковану – з 3250 ітерації.

Отже, модифікована модель шифрування даних, що представлені у вигляді зображень шифрування зображень, дозволяє пришвидшити шифрування цих даних у 1,5 рази.

4.3.2. Навчання на зашифрованих запропонованим методом даних

Результати точності вирішення задачі класифікації оригінальних та зашифрованих даних представлені у на рис. 4.4.

Згідно з отриманими експериментальними результатами, класифікація зображень, що здійснена з використанням алгоритму на основі згорткових мереж (CNN), оригінальних даних має на 4 % кращу точність, ніж класифікація зашифрованих даних з відомим ключем шифрування. Якщо ж така класифікація здійснена з використанням лінійного класифікатора, то точність на зашифрованих даних з відомим ключем лише на 2 % менша.

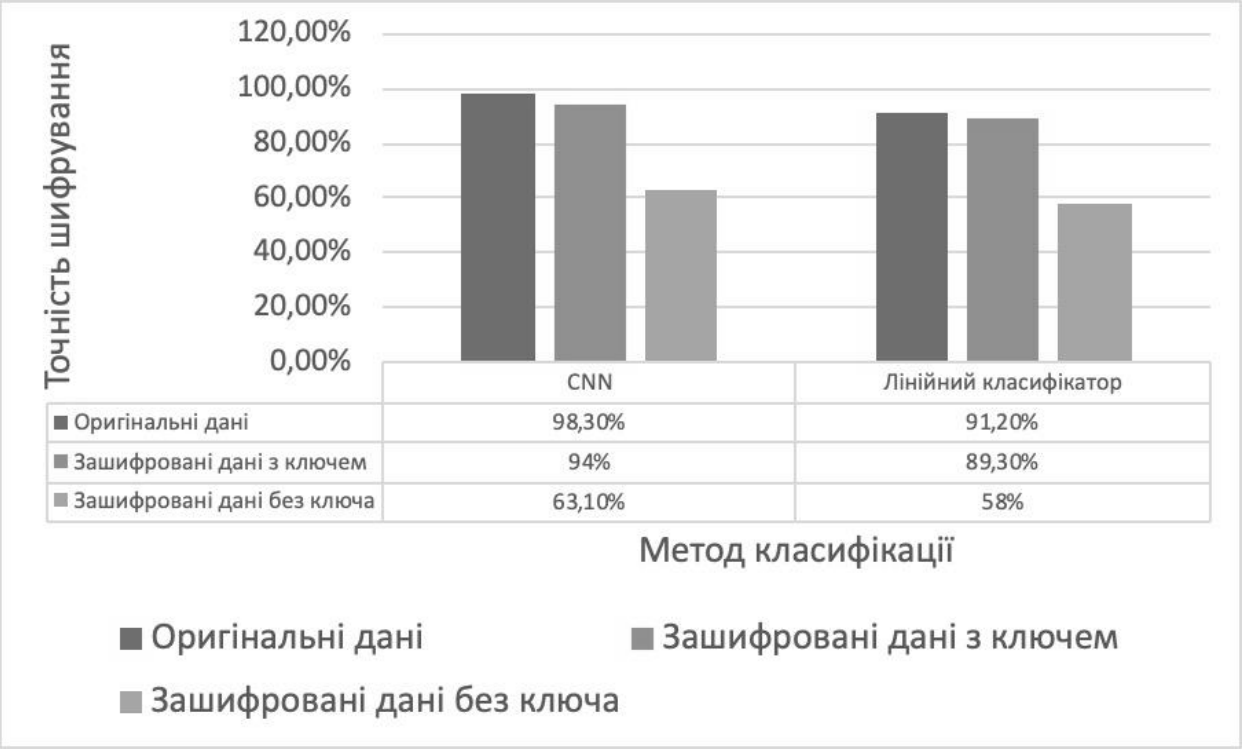


Рис. 4.4. Експериментальні результати точності класифікації оригінальних і зашифрованих даних

При вирішенні класифікації зашифрованих даних без ключа, результати свідчать про те, що такі дані є недостатньо корисними для навчання, адже точність класифікації зображень у такому випадку не перевищує 64 %.

4.4. Висновки

1. Розроблено програмну систему, яка реалізує метод шифрування даних, і дозволяє здійснювати класифікацію оригінальних та зашифрованих даних.
2. Розроблено архітектуру програмного забезпечення, особливістю якої є інкапсуляція алгоритму попередньої обробки вхідного набору даних, що дозволяє на основі шаблону «Стратегія» замінювати конкретну реалізацію в екземплярі класу класифікатор без застосування наслідування.

3. Запропоновано метод шифрування наборів даних, що дозволяє використовувати приватні дані в публічних системах штучного інтелекту й аналізу даних. Зокрема, при вирішенні задачі класифікації зображень, точність отриманих результатів на зашифрованих даних з відомим ключем шифрування була лише на 2-4 % нижчою (залежно від обраного методу класифікації), ніж точність класифікації оригінальних даних.
4. Модифіковано модель шифрування даних, шляхом використання двовимірних згорткових нейронних мереж, що дозволяє пришвидшити у 1,5 рази шифрування зображень, порівняно з класичною моделлю, яка призначена шифрування бітів.

ВИСНОВКИ

Отже, зважаючи на проведене дослідження, виконане в даній магістерській дисертації, можна зробити наступні висновки:

1. Проаналізовано переваги та недоліки існуючих методів захисту приватних наборів даних (були розглянуті методи генерування синтетичних даних, анонімізацію даних, диференційну приватність, гомоморфне шифрування та федеративне навчання);
2. Проаналізовано модель шифрування даних з використанням генеративних конкуруючих нейронних мереж, яка здійснює шифрування наборів біт. Зокрема, було розглянуто функції втрат нейронних мереж, що в неї входять і гіперпараметри, що використовуються при побудові даної моделі;
3. Запропоновано метод шифрування наборів даних, що дозволяє використовувати приватні дані в публічних системах штучного інтелекту й аналізу даних. Зокрема, при вирішенні задачі класифікації зображень, точність отриманих результатів на зашифрованих даних з відомим ключем шифрування була лише на 2-4 % нижчою (залежно від обраного методу класифікації), ніж точність класифікації оригінальних даних;
4. Модифіковано модель шифрування даних, шляхом використання двовимірних згорткових нейронних мереж, що дозволяє пришвидшити у 1,5 рази шифрування зображень, порівняно з класичною моделлю, яка призначена шифрування бітів;
5. Розроблено програмну систему, яка реалізує метод шифрування даних, і дозволяє здійснювати класифікацію оригінальних та зашифрованих даних;
6. Розроблено архітектуру програмного забезпечення, особливістю якої є інкапсуляція алгоритму попередньої обробки вхідного набору даних, що дозволяє на основі шаблону «Стратегія»

замінювати конкретну реалізацію в екземплярі класу класифікатор без застосування наслідування.

Подальшими напрямками роботи є емпірична оцінка точності навчання на зашифрованих приватних даних для задач кластеризації та регресії.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Thaine P. Perfectly Privacy-Preserving AI [Електронний ресурс] / Patricia Thaine. — 2020. — Режим доступу: https://towardsdatascience.com/perfectly-privacy-preserving-ai-c14698f322f5.
2. Lauter K. Faculty Summit 2017: Private AI [Електронний ресурс] / Kristin Lauter // Microsoft Research. — 2017. — Режим доступу: https://www.microsoft.com/en-us/research/wp-content/uploads/2017/07/Private_AI_Kristin_Lauter.pdf.
3. Nikolenko S. I. Synthetic Data for Deep Learning [Електронний ресурс] / Sergey I. Nikolenko. — 2019. — Режим доступу: https://arxiv.org/pdf/1909.11512.pdf.
4. Generative Adversarial Nets [Text] / [I. J. Goodfellow, J. Pouget-Abadie, M. Mirza та ін.]. // Advances in neural information processing systems. — 2014. — Vol. 2 — P. 2672–2680.
5. Generative Adversarial Networks [Електронний ресурс] — Режим доступу: https://developers.google.com/machine-learning/gan.
6. Data Anonymization Techniques [Електронний ресурс]. — 2019. — Режим доступу: https://www.solarwindsmsp.com/blog/data-anonymization-overview.
7. Guide to basic data anonymisation techniques [Електронний ресурс] // Personal Data Protection Commission Singapore (PDPC). — 2018. — Режим доступу: https://iapp.org/media/pdf/resource_center/Guide_to_Anonymisation.pdf.
8. Dwork C. The Algorithmic Foundations of Differential Privacy [Text] / C. Dwork, A. Roth. // Foundations and Trends® in Theoretical Computer Science. — 2014. — Vol. 9, №3-4. — С. 211–407. — DOI 10.1561/0400000042.
9. Minelli M. Fully homomorphic encryption for machine learning [Text] / Michele Minelli., 2018. — 157 p.
10. Lindell Y. Secure Multiparty Computation (MPC) [Електронний ресурс] / Yehuda Lindell — Режим доступу: https://eprint.iacr.org/2020/300.pdf.

11. Communication-efficient learning of deep networks from decentralized data. [Text] / [H. Brendan McMahan, E. Moore, D. Ramage та ін.]. — 2016.
12. Konečný J. Federated Optimization: Distributed Optimization Beyond the Datacenter [Електронний ресурс] / J. Konečný, B. McMahan, D. Ramage. — 2015. — Режим доступу: https://arxiv.org/pdf/1511.03575.pdf.
13. Brendan McMahan H. Federated Learning: Collaborative Machine Learning without Centralized Training Data [Електронний ресурс] / H. Brendan McMahan, D. Ramage. — 2017. — Режим доступу: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.
14. Kuchler H. Pharma groups combine to promote drug discovery with AI [Електронний ресурс] / Hannah Kuchler // Financial Times. — 2019. — Режим доступу: https://www.ft.com/content/ef7be832-86d0-11e9-a028-86cea8523dc2.
15. Abadi M. Learning to Protect Communications with Adversarial Neural Cryptography [Електронний ресурс] / M. Abadi, D. G. Andersen. — 2016. — Режим доступу: https://arxiv.org/abs/1610.06918.
16. Rodriguez J. Adversarial Neural Cryptography can Solve the Biggest Friction Point in Modern AI [Електронний ресурс] / Jesus Rodriguez. — 2018. — Режим доступу: https://towardsdatascience.com/adversarial-neural-cryptography-can-solve-the-biggest-friction-point-in-modern-ai-cc13b337f969.
17. Smart N. Cryptography: An Introduction / Nigel Smart. — London, St. Louis: Mcgraw-Hill College, 2004. — 433 p. — ISBN 978-0-07-709987-9, DOI 10.5555/1206247.
18. CS231n Convolutional Neural Networks for Visual Recognition [Електронний ресурс] — Режим доступу: https://cs231n.github.io/convolutional-networks/.
19. Saha S. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way [Електронний ресурс] / Sumit Saha. — 2018. — Режим доступу: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

20. Adversarial Neural Cryptography [Електронний ресурс]. — 2018. — Режим доступу: https://mathybit.github.io/adversarial-neural-crypto/#ai-based-encryption-poc.
21. Северін А. І. Метод шифрування даних з використанням нейронних мереж [Текст] / М. В. Онай, А. І. Северін. // Прикладна математика та комп'ютинг. ПМК, 2019 : дванадцята наук. конф. магістрантів та аспірантів, Київ, 13-15 лист. 2019 р. : зб. тез доп. — К.: Просвіта. — 2019. — С. 95–98 — ISBN 978-617-7010-16-5.
22. Triastcyn A. Generating Differentially Private Datasets Using GANs [Електронний ресурс] / A. Triastcyn, B. Faltings. — 2018. — Режим доступу: https://openreview.net/pdf?id=rJv4XWZA-.
23. Learning Perfectly Secure Cryptography to Protect Communications with Adversarial Neural Cryptography [Text] / [M. Coutinho, R. D. Albuquerque, F. Borges та ін.]. // Sensors (Basel). — 2018.
24. Singh B. Evaluation Metrics for Machine Learning Models [Електронний ресурс] / Bhajandeep Singh. — 2019. — Режим доступу: https://heartbeat.fritz.ai/evaluation-metrics-for-machine-learning-models-d42138496366.
25. Провост Ф. Data Science для бізнесу. Як збирати, аналізувати і використовувати дані / Ф. Провост, Т. Фоусетт: пер. з англ. — Київ: Наш формат, 2019. — 400 с. — ISBN 978-617-7730-03-2.
26. Selawsky J. Top five programming languages for AI and machine learning you should learn this year [Електронний ресурс] / John Selawsky. — 2019. — Режим доступу: https://www.itproportal.com/features/top-five-programming-languages-for-ai-and-machine-learning-you-should-learn-this-year/.
27. Beklemysheva A. Why Use Python for AI and Machine Learning? [Електронний ресурс] / Angela Beklemysheva — Режим доступу: https://steelkiwi.com/blog/python-for-ai-and-machine-learning/.

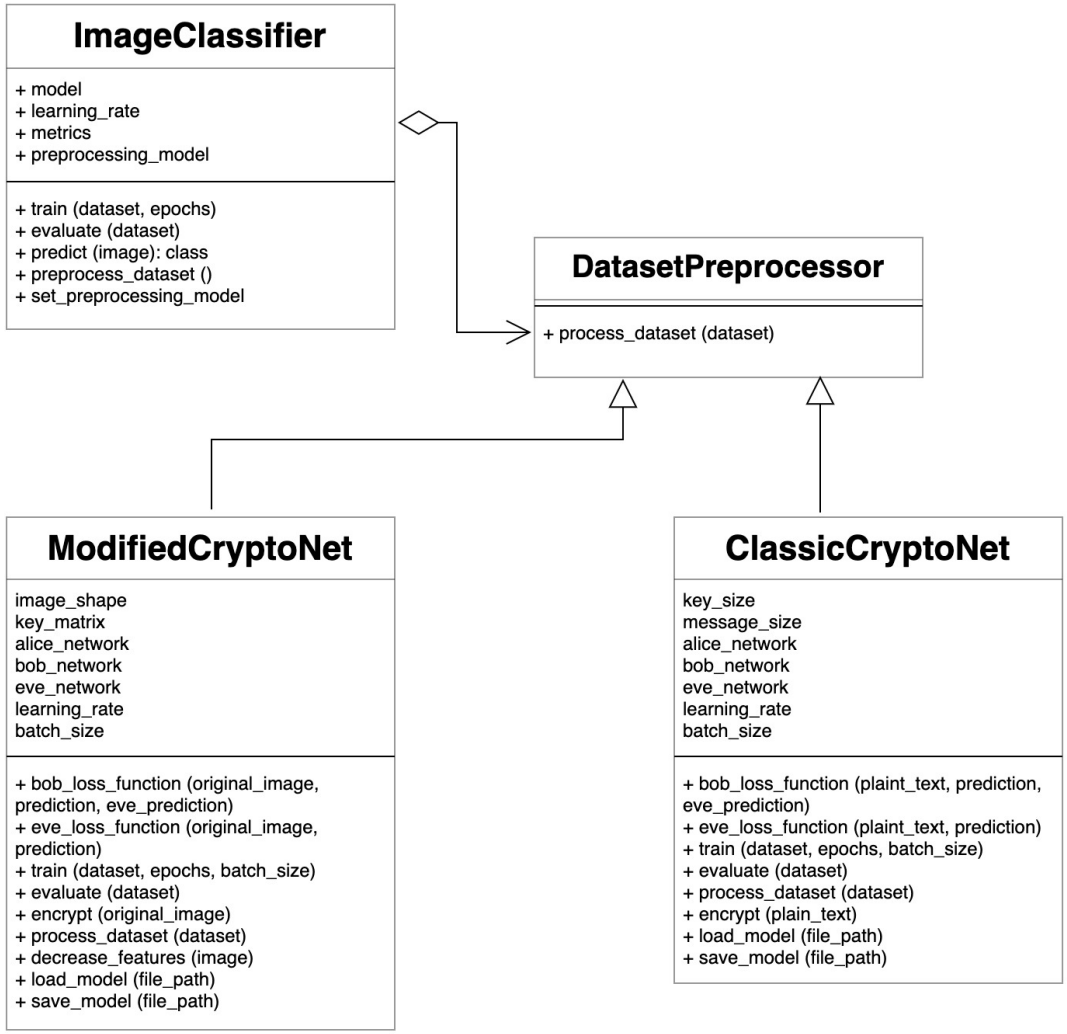
28. Sharma A. Top 9 Python Libraries for Machine Learning in 2020 [Електронний ресурс] / Aditya Sharma. — 2020. — Режим доступу: https://www.upgrad.com/blog/top-python-libraries-for-machine-learning/.
29. Kharkovyna O. Top 10 Best Deep Learning Frameworks in 2019 [Електронний ресурс] / Oleksii Kharkovyna. — 2019. — Режим доступу: https://towardsdatascience.com/top-10-best-deep-learning-frameworks-in-2019-5ccb90ea6de.
30. Tutorials | TensorFlow Core [Електронний ресурс] — Режим доступу: https://www.tensorflow.org/tutorials.
31. Keras API reference [Електронний ресурс] — Режим доступу: https://keras.io/api/.
32. Machine Learning Guides [Електронний ресурс] — Режим доступу: https://developers.google.com/machine-learning/guides.
33. Мартин, Р. Чистый код. Создание, анализ и рефакторинг [Текст] / Р. Мартин. : пер. с англ. — СПб.: Питер, 2010. — 464 с. — ISBN 978-5-49807-381-1.
34. Python Design Patterns Tutorial [Електронний ресурс] — Режим доступу: https://www.tutorialspoint.com/python_design_patterns/index.htm.

ДОДАТКИ

Додаток 1

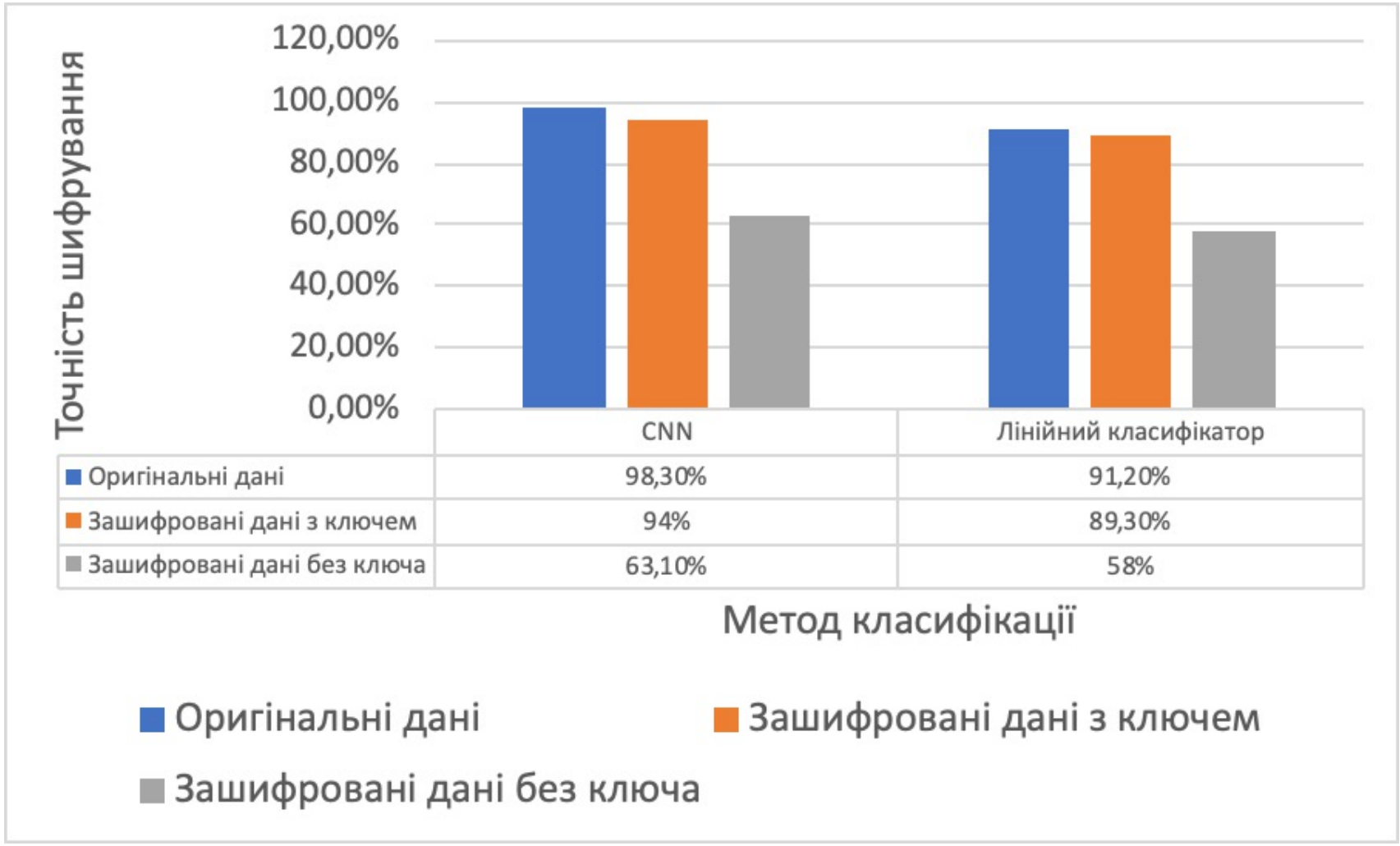
Копії графічних матеріалів

ДІАГРАМА КЛАСІВ РОЗРОБЛЕНОЇ ПРОГРАМНОЇ СИСТЕМИ



Северін А.І., група КП-81мн

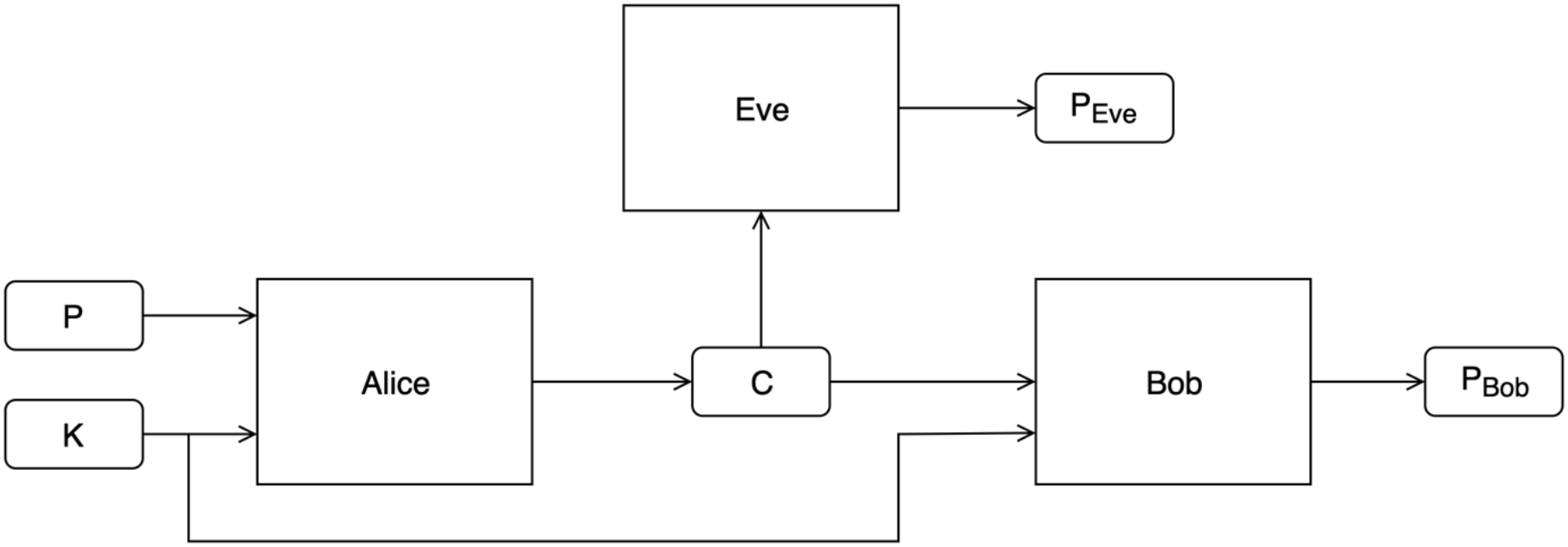
ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ ТОЧНОСТІ ВИРІШЕННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ОРИГІНАЛЬНИХ ТА ЗАШИФРОВАНИХ ДАНИХ



	CNN	Лінійний класифікатор
Оригінальні дані	98,30%	91,20%
Зашифровані дані з ключем	94%	89,30%
Зашифровані дані без ключа	63,10%	58%

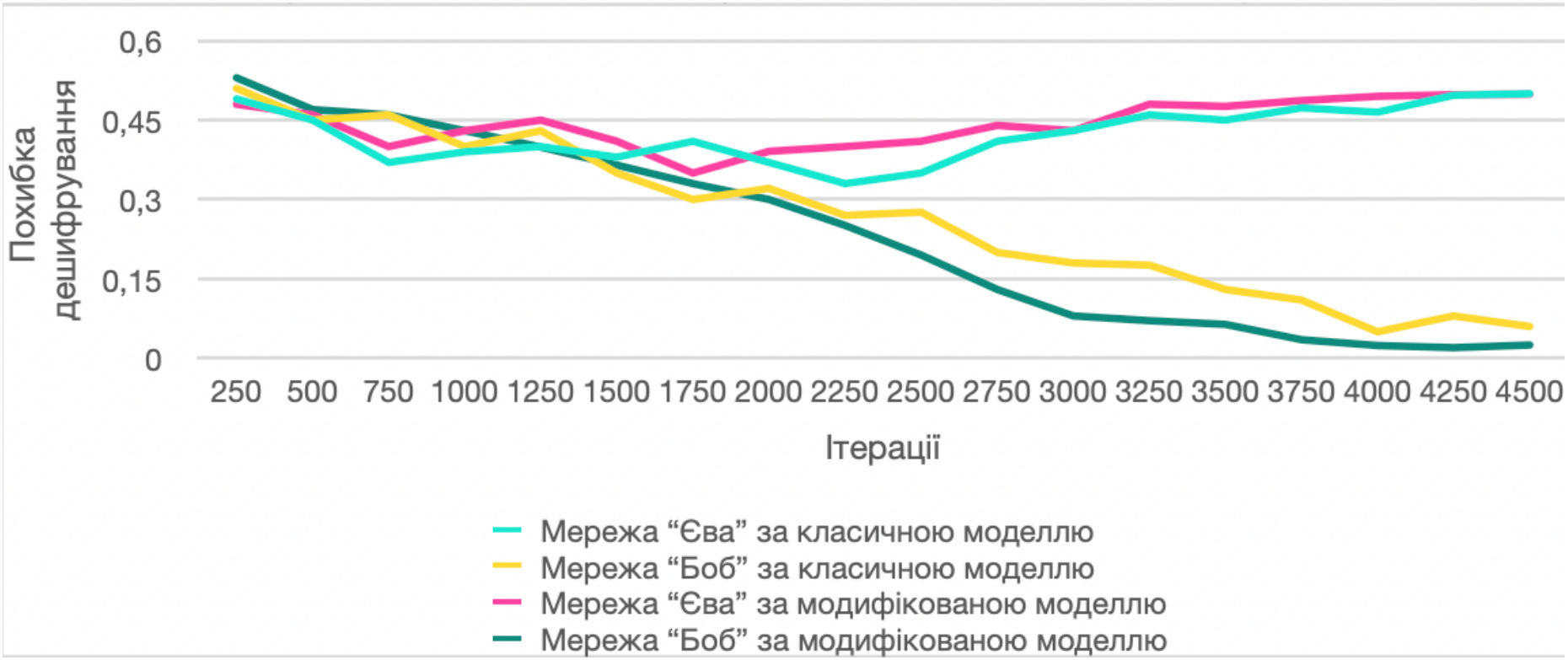
Северін А.І., група КП-81мн

МОДЕЛЬ ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ ГЕНЕРАТИВНИХ КОНКУРУЮЧИХ НЕЙРОННИХ МЕРЕЖ



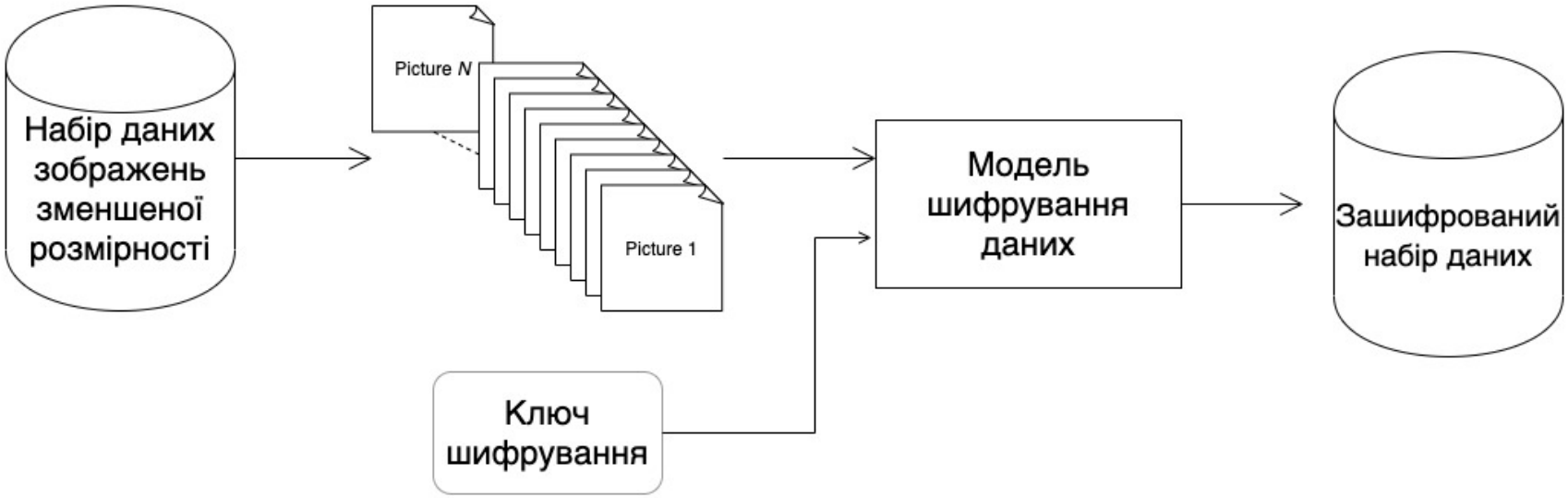
Северін А.І., група КП-81мн

ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ ТРИВАЛОСТІ НАВЧАННЯ МОДИФІКОВАНОЇ МОДЕЛІ ШИФРУВАННЯ ДАНИХ



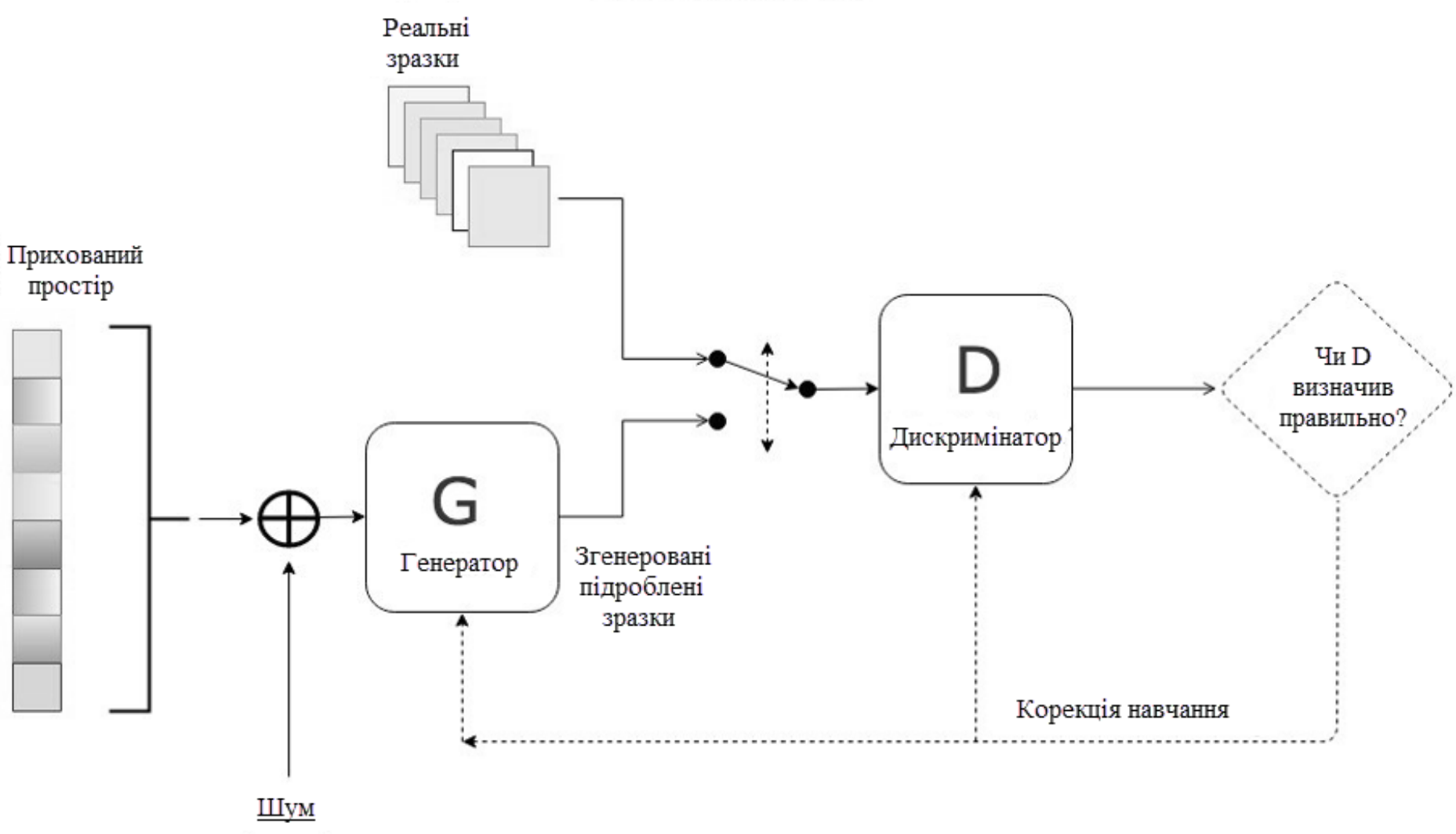
Северін А.І., група КП-81мн

СХЕМА ЕТАПУ ФОРМУВАННЯ НОВОГО НАБОРУ ДАНИХ ЗАПРОПОНОВАНИМ МЕТОДОМ



Северін А.І., група КП-81мн

СХЕМА РОБОТИ ГЕНЕРАТИВНОЇ КОНКУРУЮЧОЇ МЕРЕЖІ



Северін А.І., група КП-81мн

Додаток 2

Фрагменти тексту програми

main.py

```
import tensorflow as tf
import matplotlib.pyplot as plt

import ImageClassifier
from EncryptionModule import ClassicCryptoNet, ModifiedCryptoNet


def check_dataset_balance(dataset):
    x = dataset[0]
    y = dataset[1]

    dictionary = dict()

    for i in range(y.shape[0]):
        class_element = y[i - 1]
        if class_element in dictionary:
            dictionary[class_element] += 1
        else:
            dictionary.update({class_element: 1})

    print(dictionary)


train_dataset, test_dataset = tf.keras.datasets.mnist.load_data()
print('train_dataset')
check_dataset_balance(train_dataset)
print('test_dataset')
check_dataset_balance(test_dataset)
img_rows = 28
img_cols = 28



image_classifier = ImageClassifier.ImageClassifier()
image_classifier.train(train_dataset)
image_classifier.evaluate(test_dataset)

image_index = 4444
image_to_predict = test_dataset[0][image_index]
plt.imshow(image_to_predict.reshape(28, 28), cmap='Greys')
plt.show()
pred = image_classifier.predict(image_to_predict.reshape(1, 28, 28, 1))
print(pred.argmax())


print('Classic Model')
image_classifier = ImageClassifier.ImageClassifier()
image_classifier.set_preprocessing_model(ClassicCryptoNet)
image_classifier.train(train_dataset)
image_classifier.evaluate(test_dataset)

image_index = 4444
image_to_predict = test_dataset[0][image_index]
plt.imshow(image_to_predict.reshape(28, 28), cmap='Greys')
plt.show()
pred = image_classifier.predict(image_to_predict.reshape(1, 28, 28, 1))
print(pred.argmax())


print('Modified Model')
image_classifier = ImageClassifier.ImageClassifier()
image_classifier.set_preprocessing_model(ModifiedCryptoNet)
```

```
image_classifier.train(train_dataset)
image_classifier.evaluate(test_dataset)

image_index = 4444
image_to_predict = test_dataset[0][image_index]
plt.imshow(image_to_predict.reshape(28, 28), cmap='Greys')
plt.show()
pred = image_classifier.predict(image_to_predict.reshape(1, 28, 28, 1))
print(pred.argmax())
```

ImageClassifier.py

```
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D


# from EncryptionModule import DatasetPreprocessor


class ImageClassifier:
    model = None
    learning_rate = 0.015
    metrics = ['accuracy']
    preprocessing_model = None  # DatasetPreprocessor()

    input_shape = None

    def __init__(self, input_shape=(28, 28, 1)):
        self.input_shape = input_shape
        self.model = self.__build_model()

    def train(self, dataset, epochs=10):
        x_train = dataset[0]
        y_train = dataset[1]

        # Making sure that the values are float
        x_train = x_train.reshape(x_train.shape[0],
                                  self.input_shape[0],
                                  self.input_shape[1],
                                  self.input_shape[2])
        x_train = x_train.astype('float32')
        # Normalizing the RGB codes
        x_train /= 255

        self.model.fit(x=x_train, y=y_train, epochs=epochs)

    def evaluate(self, dataset):
        x_test = dataset[0]
        y_test = dataset[1]

        # Making sure that the values are float
        x_test = x_test.reshape(x_test.shape[0],
                                self.input_shape[0],
                                self.input_shape[1],
                                self.input_shape[2])
        x_test = x_test.astype('float32')
        # Normalizing the RGB codes
        x_test /= 255

        self.model.evaluate(x_test, y_test)
```

```
    def predict(self, image):
        return self.model.predict(image.reshape(1,
                                                self.input_shape[0],
                                                self.input_shape[1],
                                                self.input_shape[2]))

    def preprocess_dataset(self, dataset):
        self.preprocessing_model.process_dataset(dataset)

    def set_preprocessing_model(self, preprocessing_model):
        self.preprocessing_model = preprocessing_model

    def __build_model(self):
        # Creating a Sequential Model and adding the layers
        model = Sequential()
        model.add(Conv2D(28, kernel_size=(3, 3),
input_shape=self.input_shape))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Flatten())  # Flattening the 2D arrays for fully
connected layers
        model.add(Dense(128, activation=tf.nn.relu))
        model.add(Dropout(0.2))
        model.add(Dense(10, activation=tf.nn.softmax))
        model.compile(optimizer='adam',
                      loss='sparse_categorical_crossentropy',
                      metrics=self.metrics)
        return model
```

EncryptionModule.py

```
import tensorflow as tf
import numpy as np


class DatasetPreprocessor:
    def __init__(self):
        pass

    def process_dataset(self, dataset):
        pass


class ClassicCryptoNet(DatasetPreprocessor):
    key_size = 16
    message_size = 16
    learning_rate = 0.0015
    batch_size = 5

    alice_network = None
    bob_network = None
    eve_network = None

    key = None

    sess = None
    alice_saver = None
    bob_saver = None
    eve_saver = None

    eve_opt = None
    bob_opt = None
```

```
    def __init__(self):
        DatasetPreprocessor.__init__(self)

        self.alice_network = self.__build_model('Alice',
alice_input_message, self.key)
        self.bob_network = self.__build_model('Bob', self.alice_network,
self.key)
        self.eve_network = self.__build_model('Eve', self.alice_network)


        alice_vars = tf.get_collection(tf.GraphKeys.GLOBAL_VARIABLES,
scope='Alice')
        bob_vars = tf.get_collection(tf.GraphKeys.GLOBAL_VARIABLES,
scope='Bob')
        eve_vars = tf.get_collection(tf.GraphKeys.GLOBAL_VARIABLES,
scope='Eve')
        self.alice_saver = tf.train.Saver(alice_vars)
        self.bob_saver = tf.train.Saver(bob_vars)
        self.eve_saver = tf.train.Saver(eve_vars)

        self.eve_opt =
tf.train.AdamOptimizer(learning_rate=self.learning_rate, beta1=0.9,
epsilon=1e-08)
            .minimize(self.eve_loss_function, var_list=eve_vars])
        self.bob_opt =
tf.train.AdamOptimizer(learning_rate=self.learning_rate, beta1=0.9,
epsilon=1e-08)
            .minimize(self.bob_loss_function, var_list=[alice_vars +
bob_vars])
        self.sess = tf.Session()
        init = tf.global_variables_initializer()
        self.sess.run(init)


    def bob_loss_function(self, plain_text, prediction, eve_prediction):
        return self.eve_loss_function(plain_text,prediction) +\
               tf.reduce_sum(tf.square(float(self.message_size) / 2.0 -

self.eve_loss_function(plain_text,eve_prediction)) /
                             ((self.message_size / 2) ** 2))

    def eve_loss_function(self, plain_text, prediction):
        return (1 / self.batch_size) * tf.reduce_sum(tf.abs(prediction -
plain_text))

    def train(self, dataset, epochs, batch_size):
        messages = dataset[0]
        keys = build_key_matrix(self.message_size, self.key_size)
        steps_per_epoch = 15
        ITERS_PER_ACTOR = 2
        EVE_MULTIPLIER = 1

        # Training begins
        for i in range(epochs):

            for j in range(steps_per_epoch):

                # get batch dataset to train
                batch_messages = messages[j * batch_size: (j + 1) *
batch_size]
                batch_keys = keys[j * batch_size: (j + 1) * batch_size]

                # Train Alice and Bob
                for _ in range(ITERS_PER_ACTOR):
```

```
                    temp = self.sess.run(
                        [self.bob_opt, self.alice_loss_function(),
self.bob_loss_function, self.bob_network],
                                         feed_dict={alice_input_message:
batch_messages, key: batch_keys})

                    temp_alice_bob_loss = temp[1]
                    temp_eve_evs_loss = temp[2]
                    temp_bob_msg = temp[3]

                # train Eve
                for _ in range(ITERS_PER_ACTOR + EVE_MULTIPLIER):
                    temp = self.sess.run([eve_opt, self.eve_loss_function,
self.eve_network],
                                         feed_dict={alice_input_message:
batch_messages, key: batch_keys})

                    temp_eve_loss = temp[1]
                    temp_eve_msg = temp[2]

            # save after every 5 epochs
            if i % 5 == 0 and i != 0:
                self.save_model()

            # output bit error and loss after every epoch
            if i % 1 == 0:
                print('  epochs: ', i, '  bob bit error: ',
temp_alice_bob_loss, ' + ', temp_eve_evs_loss,
                      '   & eve bit error:', temp_eve_loss)

    def evaluate(self, dataset):
        pass

    def process_dataset(self, dataset):
        encrypted_dataset = []
        for element in dataset:
            encrypted_dataset.append(self.alice_network(element))
        pass

    def encrypt(self, plain_text):
        return self.alice_network(plain_text, self.key)

    def load_model(self, filepath="./weights/"):
        self.alice_saver.restore(self.sess, filepath +
"alice_weights/model.ckpt")
        self.bob_saver.restore(self.sess, filepath +
"bob_weights/model.ckpt")
        self.eve_saver.restore(self.sess, filepath +
"eve_weights/model.ckpt")

    def save_model(self, filepath="./weights/"):
        self.alice_saver.save(self.sess, filepath +
"alice_weights/model.ckpt")
        self.bob_saver.save(self.sess, filepath + "bob_weights/model.ckpt")
        self.eve_saver.save(self.sess, filepath + "eve_weights/model.ckpt")

    def __build_model(self, collection, message, key=None):
        if key is not None:
            combined_message = tf.concat(axis=1,
                                         values=[message, key])
        else:
            combined_message = message

        with tf.variable_scope(collection):
```

```
            fc = tf.layers.dense(combined_message, self.message_size +
self.key_size, activation=tf.nn.relu)
            fc = tf.expand_dims(fc, 2)

            conv1 = tf.layers.conv1d(fc, filters=2, kernel_size=4,
strides=1, padding='SAME', activation=tf.nn.sigmoid)
            conv2 = tf.layers.conv1d(conv1, filters=4, kernel_size=2,
strides=2, padding='VALID',
                                     activation=tf.nn.sigmoid)
            conv3 = tf.layers.conv1d(conv2, filters=4, kernel_size=1,
strides=1, padding='SAME',
                                     activation=tf.nn.sigmoid)

            # output
            conv4 = tf.layers.conv1d(conv3, filters=1, kernel_size=1,
strides=1, padding='SAME', activation=tf.nn.tanh)

            out = tf.squeeze(conv4, 2)
        return out
```

Додаток 3

Копія презентації

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМП’ЮТЕРНИХ СИСТЕМ

АЛГОРИТМІЧНО-ПРОГРАМНИЙ МЕТОД ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ НЕЙРОННИХ МЕРЕЖ

Виконав: Северін Андрій Іванович

Науковий керівник:
Доцент кафедри ПЗКС, к.т.н., доцент Онай Микола Володимирович

Київ – 2020

АКТУАЛЬНІСТЬ ДОСЛІДЖЕННЯ

З кожним роком клас задач, де можна й доцільно застосовувати методи штучного інтелекту зростає.

Важливим елементом для побудови надійної і точної системи є наявність достатньої кількості даних для навчання, проте не всі дані можна використовувати для систем аналізу даних і штучного інтелекту, оскільки значна частина з них є приватними.

Пошук набору даних для AI-рішень є важливим завданням.

МЕТА ДОСЛІДЖЕННЯ

Розроблення алгоритмічно-програмного методу шифрування даних на основі генеративних конкуруючих нейронних мереж для використання зашифрованих приватних наборів даних в системах аналізу даних і штучного інтелекту.

ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Об’єкт дослідження: процеси шифрування, дешифрування та генерації інформації з використанням нейронних мереж.

Предмет дослідження: методи, способи та моделі захисту наборів даних з використанням нейронних мереж.

НАУКОВО-ПРАКТИЧНА ЗАДАЧА ВКЛЮЧАЄ НАСТУПНІ ЗАВДАННЯ:

1. Дослідження методів захисту приватних наборів даних.
2. Аналіз методів, способів, підходів та моделей шифрування та генерації наборів даних.
3. Розроблення та дослідження моделі шифрування даних з використанням нейронних мереж.
4. Розроблення алгоритмічно-програмного методу шифрування приватних наборів даних.
5. Вирішення задачі класифікації оригінальних та зашифрованих даних.
6. Аналіз отриманих результатів.

ТЕРМІНОЛОГІЯ

AI (Artificial intelligence) – штучний інтелект.

GAN (Конкуруючі нейронні мережі) – алгоритм машинного навчання без вчителя, побудований на комбінації двох нейронних мереж, одна з яких генерує зразки (генеративна), а інша намагається відрізнити правильні (оригінальні) зразки від неправильних.

P (plaint text) – вхідний текст для шифрування.

K (key) – ключ шифрування.

C (cypher text) – зашифрований текст.

Функція втрат (L, loss function) – міра того, наскільки прогноз моделі далекий від її мітки (очікуваного результату).

ГЕНЕРУВАННЯ СИНТЕТИЧНИХ ДАНИХ

Синтетичні дані – штучні дані, які згенеровані за певним алгоритмом на відміну від оригінальних даних, які базуються на реальній інформації.

Приклади генеративних моделей:

- прихована модель Маркова;
- Баєсівська мережа;
- агентне моделювання;
- варіаційний автокодувальник;
- генеративні конкуруючі нейронні мережі.

Приклад синтетичних даних

АНОНІМІЗАЦІЯ ДАНИХ

Вхідні дані

Користувач	Вік	E-mail	Інші атрибути	Кількість покупок
Олег	21	oleh@gmail.com	…	3
…	…	…	…	…
Руслан	34	ruslan@i.ua	…	8

Анонімізовані дані

Користувач	Вік	E-mail	Інші атрибути	Кількість покупок
UserN	20-25	****@gmail.com	…	3
…	…	…	….	…
UserK	> 30	******@i.ua	…	8

ДИФЕРЕНЦІЙНА ПРИВАТНІСТЬ

Диференційна приватність – метод захисту даних, який захищає конфіденційність користувача шляхом додавання випадкового шуму до даних.

Залежність цінності даних, від кількості доданого шуму

ГОМОМОРФНЕ ШИФРУВАННЯ

Гомоморфне шифрування – шифрування, яка дозволяє проводити обчислення на шифротексті, дешифрований результат яких буде таким самим, як і результат виконання операцій над відкритим текстом.

Функція f: між двома групами G та H є гомоморфною, якщо:

$$f(xy) = f(x)f(y) \qquad \text{для} \qquad \forall x, y \in G$$

Функція шифрування: $E(x) = 2x$.

Функція дешифрування: $D(x) = x / 2$.

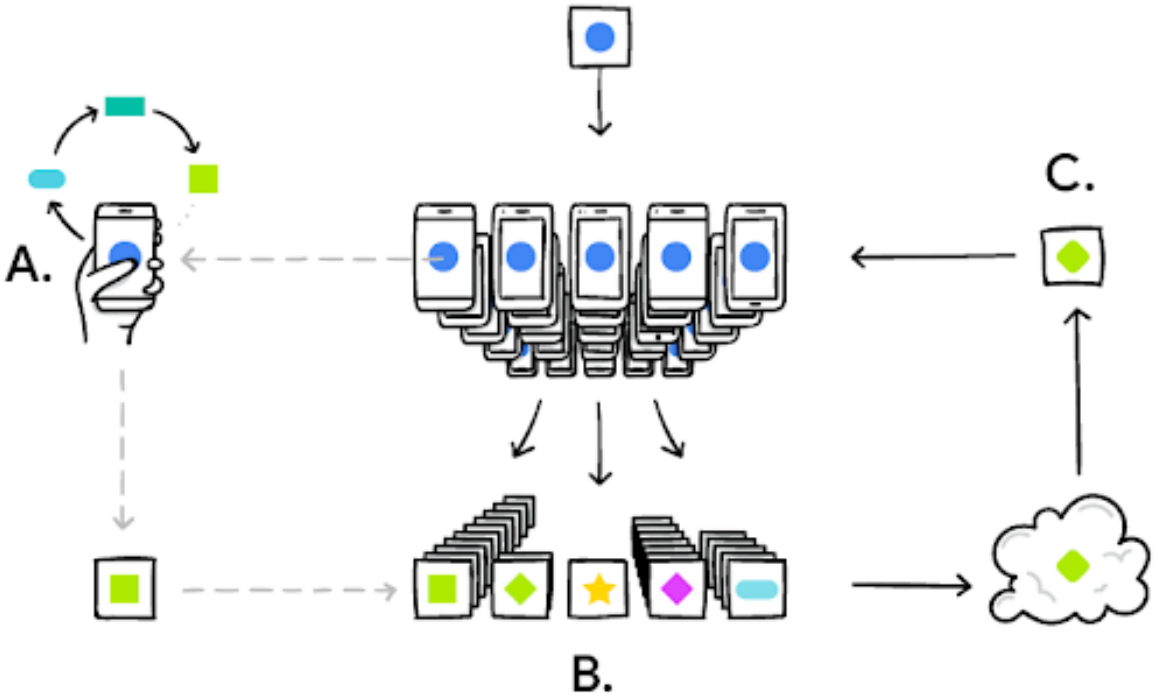
Відкритий текст: $x = x_1 + x_2 + x_3 = (2 + 5 - 8) = -1$

Зашифрований текст: $E(x) = E(x_1) + E(x_2) + E(x_3) = (4 + 10 - 16) = -2$

Дешифрований результат: $D(E(x)) = D(-2) = -1$

ФЕДЕРАТИВНЕ НАВЧАННЯ

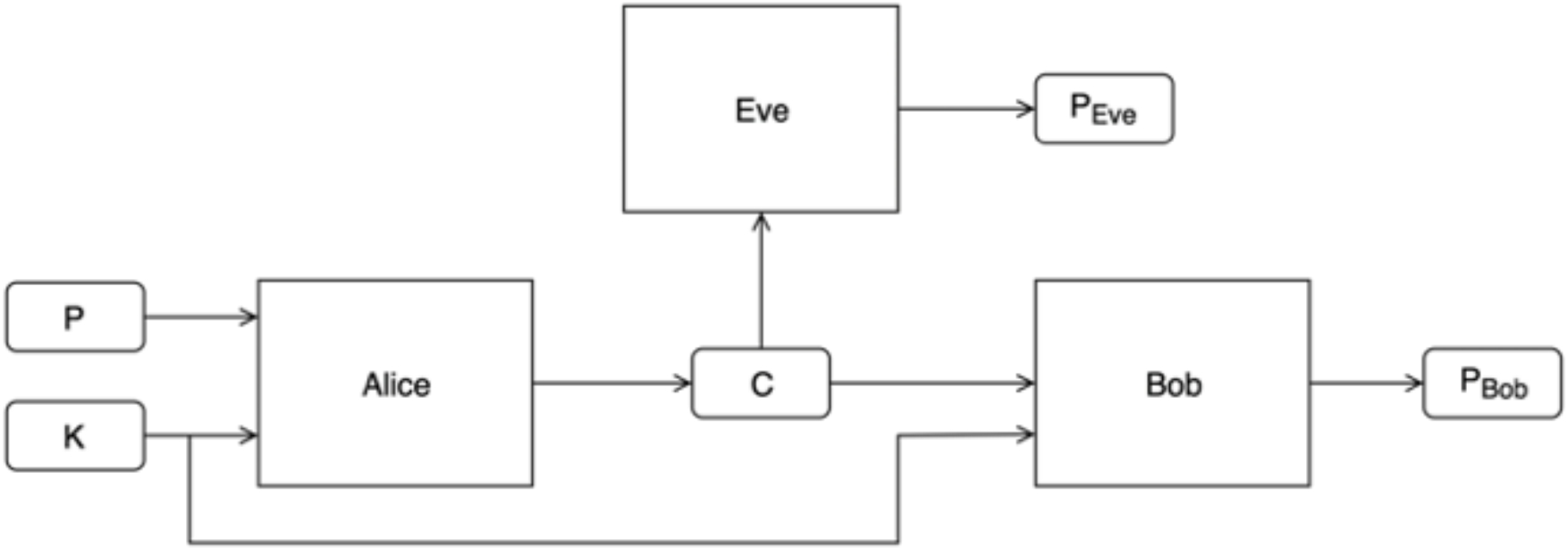
- блок A: завантаження моделі на пристрій і її удосконалення шляхом навчання на локальних даних;
- блок B: отримання й усереднення оновлень від локальних моделей сервером;
- блок C: удосконалення спільної моделі за допомогою усередненого оновлення ваг моделі;



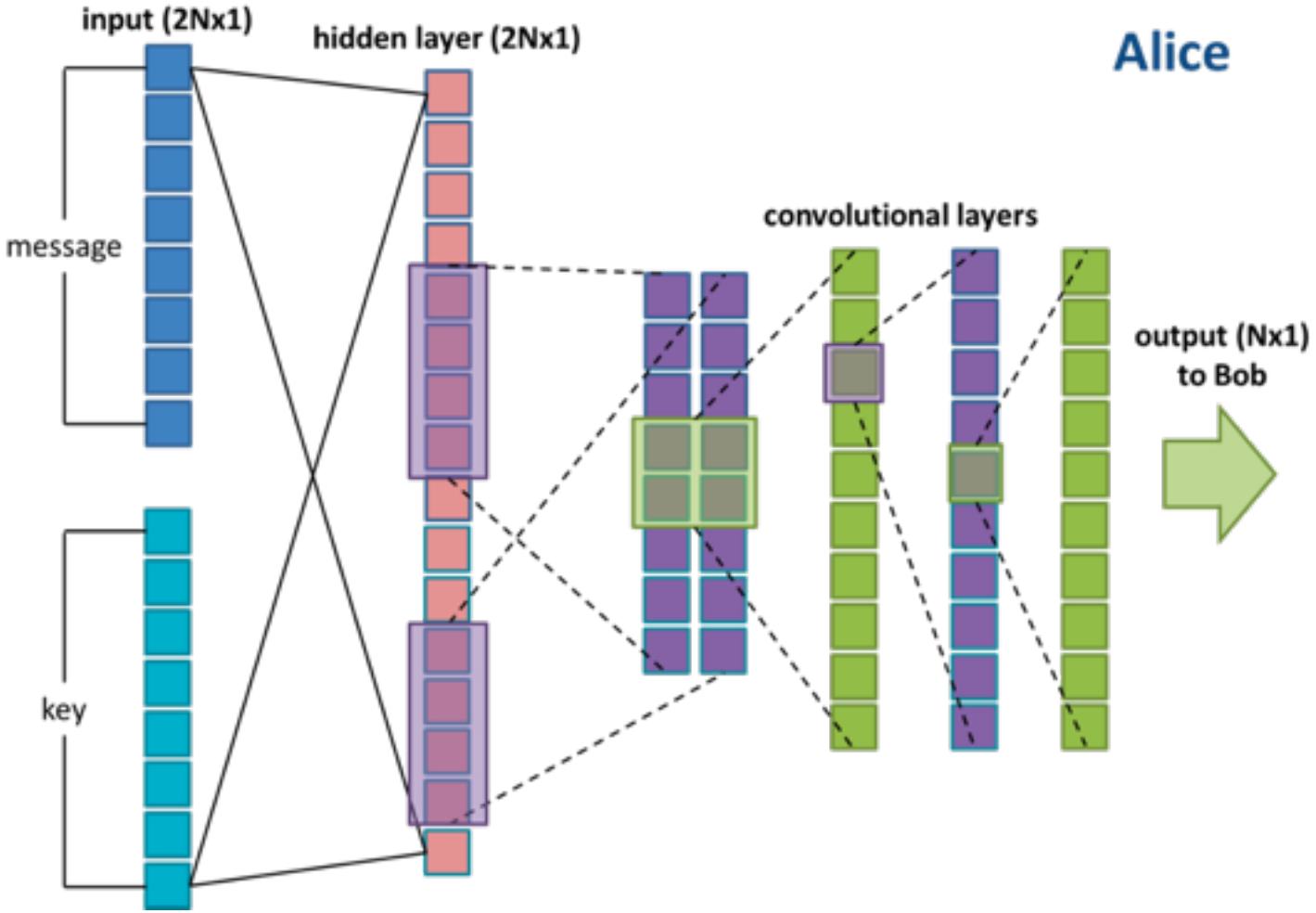
АНАЛІЗ МЕТОДІВ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

Метод	Складність	Практичність	Потреба у великій кількості даних	Надійність	Висока точність системи
Генерування синтетичних даних	+	+	+	+	+/ —
Анонімізація даних	—	+	—	—	+
Диференційна приватність	+/ —	+	+	+/ —	+/ —
Гомоморфне шифрування	+	—	—	+	+
Федеративне навчання	—	+/ —	+	+	+

МОДЕЛЬ ШИФРУВАННЯ ДАНИХ З ВИКОРИСТАННЯМ GAN



СТРУКТУРА НЕЙРОННОЇ МЕРЕЖІ “АЛІСА”



ФУНКЦІЇ ВТРАТ

- *Функція втрат для мережі «Eve»*. Відстань Хеммінга (d) між P і P_{Eve}:

$$L_{Eve} = d\left(P, P_{Eve}\right) = \sum_{j=1}^{N} \left| P_j - \left(P_{Eve}\right)_j \right| = \left\| P \oplus P_{Eve} \right\|_1 \qquad (1)$$

- *Функція втрат для мереж «Alice» і «Bob».* Абсолютна помилка дешифрування (відстань Хеммінга між вектором P і P_{Bob}), що коригується додатковим параметром $((N/2 - L_{Eve})^2 / (N/2)^2$, який показує наскільки ефективно мережа «Eve» дешифрує зашифрований текст.

$$L_{Bob} = \left\| P \oplus P_{Bob} \right\|_1 + \frac{\left(\frac{N}{2} - L_{Eve}\right)^2}{\left(\frac{N}{2}\right)^2} \qquad (2)$$

Позначення: P – бітовий вектор відкритого тексту довжиною N.

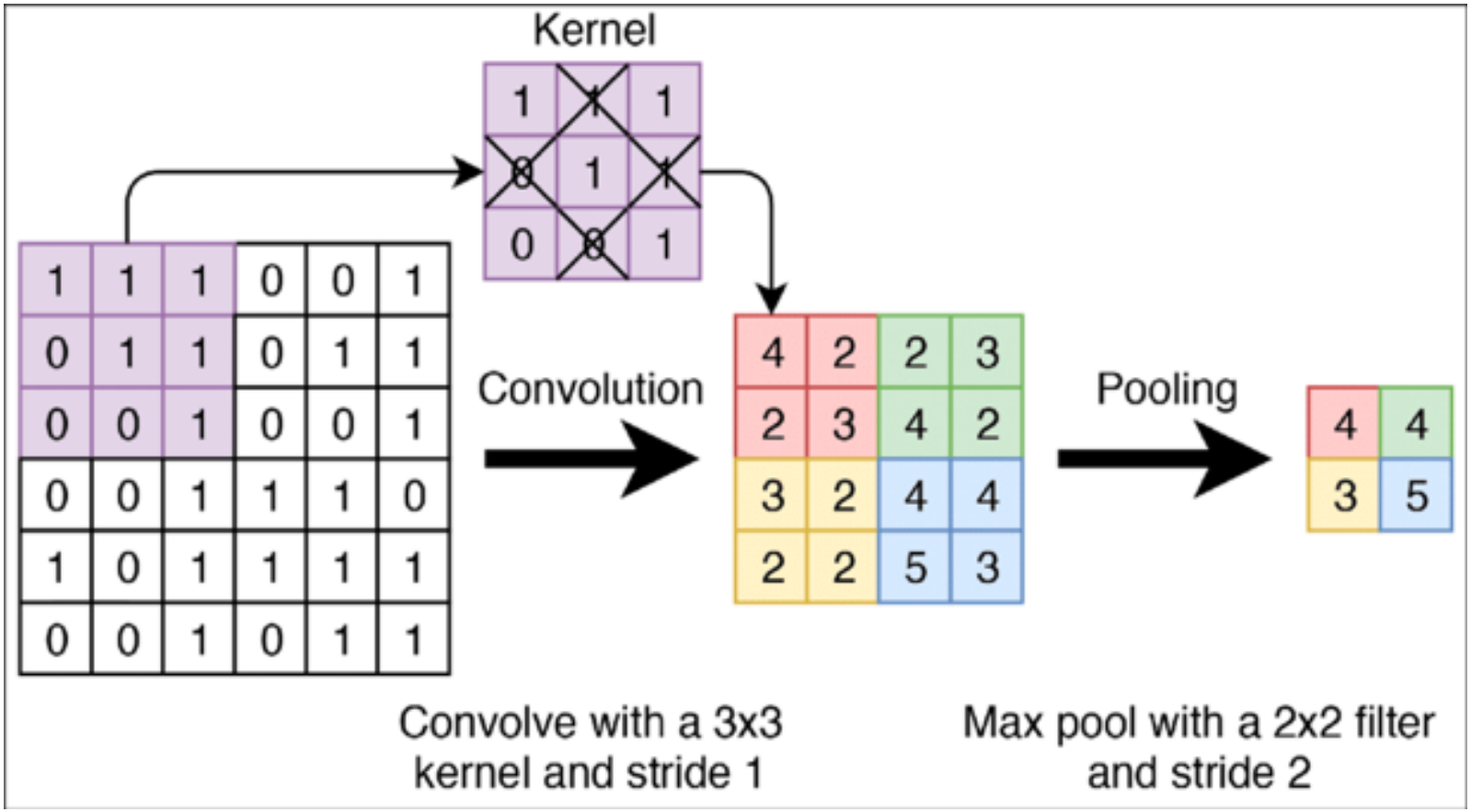
P_{Eve} , P_{Bob} – результат дешифрування тексту мережами «Eve» та «Bob».

L_{Eve} , L_{Bob} – функції втрат мереж «Eve» та «Bob».

ЗАПРОПОНОВАНИЙ МЕТОД ШИФРУВАННЯ НАБОРІВ ДАНИХ

1. Попередня обробка даних, шляхом зменшення розмірності вхідних зображень.
2. Застосування модифікованої моделі шифрування до кожного елемента набору даних.
3. Формування нового набору даних на основі зашифрованих елементів та ключа, що був використаний для шифрування.
4. Класифікація зображень зашифрованих даних.

ПОПЕРЕДНЯ ОБРОБКА ДАНИХ



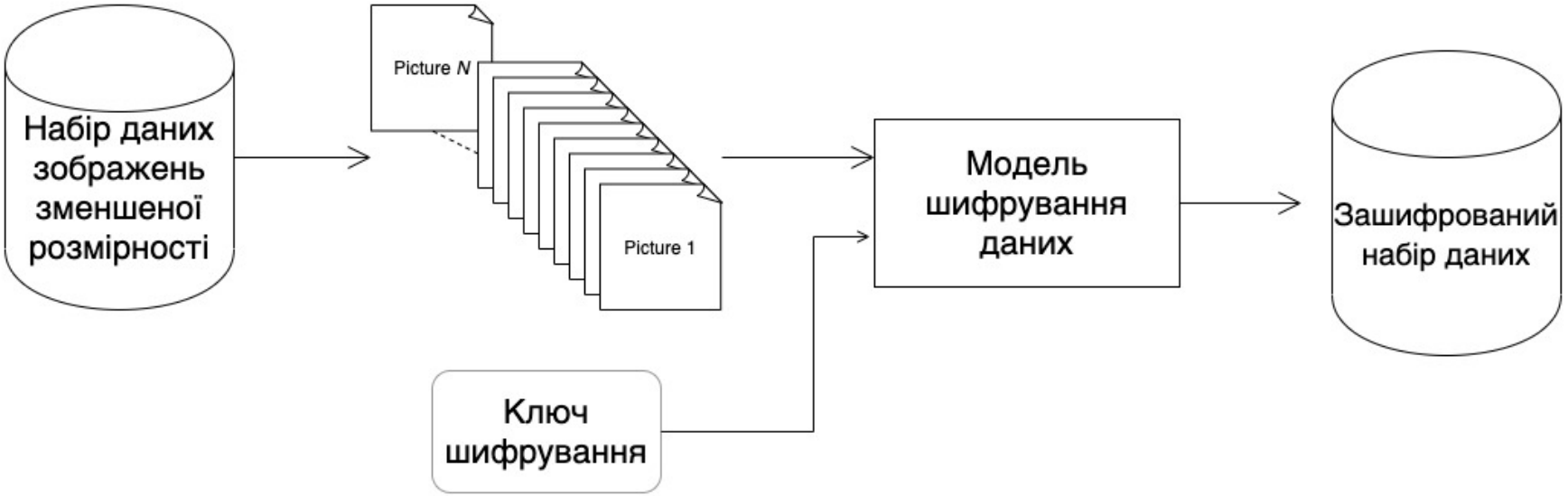
МОДИФІКАЦІЯ МОДЕЛІ ШИФРУВАННЯ ДАНИХ

- Вхідна інформація: матриця пікселів зображення зменшеної розмірності.

- Вхідний шар: об’єднання матриці пікселів з матрицею ключем
- Прихований шар: повноз’єднальний шар.
- Згорткові шари: 3-и двовимірні згорткові шари (Conv2D).

- Вихідна інформація: зашифрована матриця пікселів.

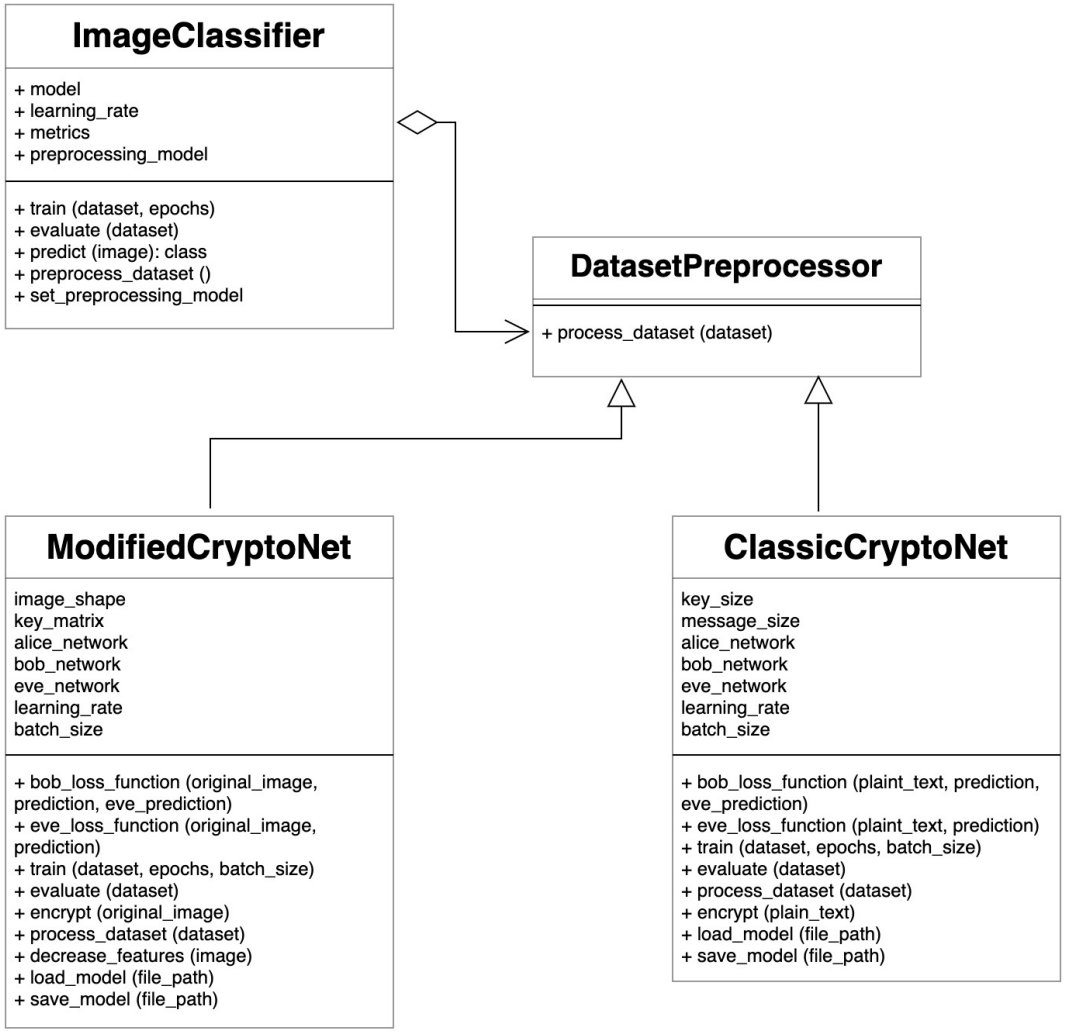
ФОРМУВАННЯ НОВОГО НАБОРУ ДАНИХ



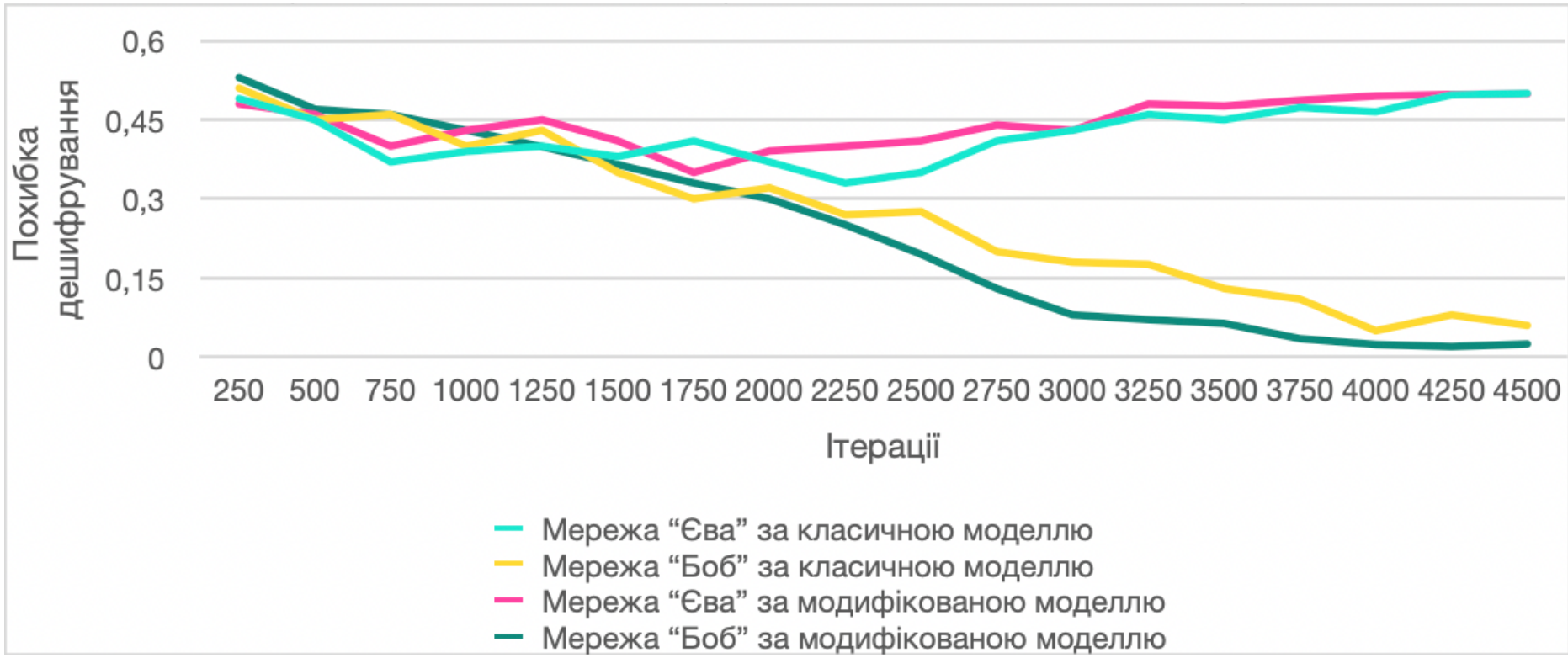
КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ ЗАШИФРОВАНИХ ДАНИХ

ЗАСОБИ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

ДІАГРАМА КЛАСІВ РОЗРОБЛЕНОЇ СИСТЕМИ



ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ ТРИВАЛОСТІ НАВЧАННЯ МОДИФІКОВАНОЇ МОДЕЛІ ШИФРУВАННЯ ДАНИХ



ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ ТОЧНОСТІ ВИКОРИСТАННЯ МЕТОДУ ШИФРУВАННЯ ДАНИХ ДЛЯ ЗАДАЧІ КЛАСИФІКАЦІЇ

Метод класифікації / Вхідні дані	CNN	Лінійний класифікатор
Оригінальні дані	98,3%	91,2%
Зашифровані дані з ключем	94%	89,3%
Зашифровані дані без ключа	63,1%	58%

НАУКОВА НОВИЗНА

1. Запропоновано метод шифрування наборів даних, що дозволяє використовувати приватні дані в системах штучного інтелекту й аналізу даних: при вирішенні задачі класифікації зображень, точність отриманих результатів на зашифрованих даних з відомим ключем шифрування лише на 2-4 % нижча, ніж точність класифікації оригінальних даних.

2. Модифіковано модель шифрування даних, шляхом використання двовимірних згорткових нейронних мереж, що дозволяє в 1,5 рази прискорити шифрування зображень, порівняно з класичною моделлю, яка призначена для шифрування окремих біт.

АПРОБУВАННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ

1. XII наукова конференція магістрантів та аспірантів «Прикладна математика та комп’ютинг» ПМК-2019 (Київ, 13-15 листопада 2019 р.).

2. Наукова стаття у міжнародному науковому журналі «Радіоелектроніка, інформатика, управління» 2020 (випуск №2), що входить до наукометричної бази даних Web of Science.

ВИСНОВКИ

1. Проаналізовано переваги та недоліки існуючих методів захисту приватних наборів даних.

2. Проаналізовано модель шифрування даних з використанням генеративних конкуруючих нейронних мереж, яка здійснює шифрування наборів біт. Розглянуто функції втрат нейронних мереж, що в неї входять і гіперпараметри, що використовуються при побудові даної моделі;

3. Запропоновано метод шифрування наборів даних, що дозволяє використовувати приватні дані в системах штучного інтелекту й аналізу даних. Зокрема, при вирішенні задачі класифікації зображень, точність отриманих результатів на зашифрованих даних з відомим ключем шифрування була лише на 2-4 % нижчою, ніж точність класифікації оригінальних даних.

ВИСНОВКИ (продовження)

4. Модифіковано модель шифрування даних, шляхом використання двовимірних згорткових нейронних мереж, що дозволяє прискорити в 1,5 рази шифрування зображень, порівняно з класичною моделлю, яка призначена для шифрування окремих біт.

5. Розроблено програмну систему, яка реалізує метод шифрування даних, і дозволяє здійснювати класифікацію оригінальних та зашифрованих даних.

ВИСНОВКИ (продовження)

6. Розроблено архітектуру програмного забезпечення, особливістю якої є інкапсуляція алгоритму попередньої обробки вхідного набору даних, що дозволяє на основі шаблону «Стратегія» замінювати конкретну реалізацію в екземплярі класу класифікатор без застосування наслідування.

7. Подальшими напрямками роботи є емпірична оцінка точності навчання на зашифрованих приватних даних для задач кластеризації та регресії.

Дякую за увагу!